ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10096221

FED-3DA: A DYNAMIC AND PERSONALIZED FEDERATED LEARNING FRAMEWORK

Hui Wang[1*], *Jie Sun*[1], *Tianyu Wo*[2], *Xudong Liu*[1]

[1] *SKLSDE, School of Computer Science and Engineering, Beihang University*, Beijing, China
[2] *SKLSDE, School of Software, Beihang University*, Beijing, China

ABSTRACT

In federated learning, the non-IID data generated from heterogeneous clients may reduce the global model efficiency. Previous studies use personalization as a common approach to adapt the global model to these clients (called the local model). However, client's data distribution may change dynamically with its location or environment, which can degrade the performance of the local model, leading to a new Dynamic Personalized Federated Learning (DPFL) problem. This paper proposes a novel approach to reduce the impact of the dynamic distribution on the local model based on meta-learning and distribution distance measurement named Fed-3DA. It calculates the distribution distance periodically to perceive the distribution change on the client and adjust the local model preferences from a global meta-model through the distribution representation. Our experiments on public datasets show that Fed-3DA can effectively reduce the performance fluctuation of the local model in DPFL scenarios.

Index Terms— Dynamic personalized federated learning, non-IID data, Meta-learning, Distribution distance

1. INTRODUCTION

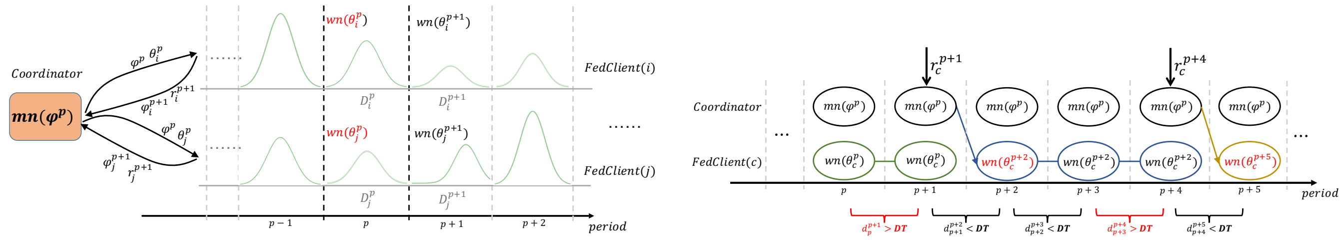
Federated Learning (FL) [1] is a distributed machine learning paradigm that implements cooperative learning among devices while protecting data privacy. FL typically involves one coordinator and multiple client devices. The global model training is an iterative process managed by the coordinator. At each training iteration, the coordinator sends the current state of the global model to the distributed devices. The devices then train the global model on their local data through stochastic gradient descent and generate local parameters, which are finally aggregated into the global model by the coordinator.

The IID sampling of the training data is important to ensure that the stochastic gradient is an unbiased estimate of the full gradient. Nonetheless, in most scenarios, the data is likely to be non-IID across devices. To address this problem, related works (e.g., [2, 3]) propose personalization-based approaches that allow each device to use a local model with personalized information for inference. However, these approaches cannot adapt to dynamic scenarios, where the data distribution of each device may experience random drift due to changes in location or environment. More specifically, since the personalized working models are strongly correlated to the training data, their performance will decline dramatically when the data distribution changes during inference. We demonstrate this phenomenon through experiments in section 3.2.

Towards this problem, a straightforward idea is to update the personalized working models regularly to adapt to the data distribution change. However, this is also a challenging task. The reason is twofold. First, in most cases, labeling the data during model inference is expensive (e.g., it may need user interactions), and we can not assume enough data for the model update. Second, even though the data can be labeled in real-time, updating the model is computationally complex for existing approaches since it generally means training the personalized working models from scratch with the new data.

To solve the DPFL problem, we propose a new approach named **Fed**erated **D**ynamic **D**ata **D**istribution **A**daptive method (Fed-3DA). Fed-3DA can automatically detect the data distribution drift during the inference stage without relying on real-time labels and update the working models efficiently without the need for re-training. To achieve the first goal, Fed-3DA divides the timeline into fixed-length periods, and generates a low-dimensional feature vector (i.e., distribution representation vector) to represent the data distributions of each client in any period. Fed-3DA periodically calculates the data distribution distance of any two consecutive periods through optimal transport strategy [4] in order to detect the distribution drift. Towards the second goal, we use a global meta-model to establish the relationship between distribution representation vectors and working models at the training stage and share the distribution information among clients. Therefore, when the client data distribution drifts, Fed-3DA can update the corresponding working model by submitting the new distribution representation vector to the meta-model.

Contribution. (1) We define the DPFL problem and propose Fed-3DA to improve the performance of the local model. In addition to the *Accuracy*, the *MPV* and *LAT* (see section 3.1) are also proposed to evaluate the stability of the local model. **(2)** The experiments based on public datasets show that Fed-3DA can effectively reduce the performance jitter of the local model in DPFL scenarios.



(a) Training of models *meta-network* (mn) and *work-network* (wn) (b) Update of model *work-network* (wn) weight

Fig. 1. *Overview of proposed Fed-3DA.*

2. METHOD

2.1. Problem Formulation

In DPFL, client c is equipped with its own data distribution $\mathcal{P}_c^p$ in period p. The data of client c in period p is denoted as D_c^p, where $D_c^p = \{(x_c^{p(i)}, y_c^{p(i)})\}_{i=1}^{S_c^p} \sim \mathcal{P}_c^p$, and S_c^p refers to the size of D_c^p. To simplify the problem, we assume all the clients have the same period size, i.e., $S_c^p \triangleq \{ S_{c_i}^{p_i} = S_{c_j}^{p_j} \mid c_i, c_j \in [C]; p_i, p_j \in [P]\}$. C and P represent the total number of clients and periods. Let $\mathcal{L}_c^p(x_c^p, y_c^p; \theta_c^p)$ denote the loss function of client c in period p, and $\mathcal{L}^p(\theta^p) = \frac{1}{C}\sum_{c=1}^{C} \mathcal{L}_c^p(x_c^p, y_c^p; \theta_c^p)$ denote the average loss of clients in period p. θ^p denotes all the local model parameters $\{\theta_c^p\}_{c=1}^{C}$ in period p. The goal of the DPFL problem is to optimize

$$\arg\min_{\Theta} \mathcal{L}(\Theta) = \arg\min_{\Theta} \frac{1}{P}\sum_{p=1}^{P} \mathcal{L}^p(\theta^p), \tag{1}$$

where Θ denotes the set of local model parameters, i.e., $\Theta = \{\theta_c^p\}_c^p, c \in [C], p \in [P]$.

2.2. Distribution Distance between Different Datasets

Data D_c^p consists of a set of entries composed of features and labels $(f,\ l)$. f and l represent the data feature vector that belong to the feature space $\mathcal{FS}$ and the label vector that belong to the label space $\mathcal{LS}$.

$$D_c^p = \left\{ \left(f_c^{p(i)},\ l_c^{p(i)} \right) \in \mathcal{FS} \times \mathcal{LS} \right\}_{i=1}^{PS},$$

where *PS* represents the period size. Given two feature-label pairs datasets D_1=(f_1, l_1) and D_2=(f_2, l_2), according to [5], the distance of them can be calculated with:

$$d_{ot} = \left(d_f\left(f_1, f_2\right)^p + d_l\left(l_1, l_2\right)^p\right)^{1/p},\ p \geq 1\ .$$

d_{ot} contains the feature distance d_f and label distance d_l. Moreover, d_l can be converted into d_f based on the definition:

$$d_l\left(l_1, l_2\right) = d_f\left(\tfrac{1}{N_1}\sum_{f\in\mathcal{U}_D(l_1)} f,\ \tfrac{1}{N_2}\sum_{f\in\mathcal{U}_D(l_2)} f\right),$$

where $\mathcal{U}_D(l^*) = \{f|(f,\ l^*) \in D\}$, $N_1 = |\mathcal{U}_D(l_1)|$, $N_2 = |\mathcal{U}_D(l_2)|$. On this basis, we can define the distribution distance between the data of two periods $D_c^m = \{(f_c^{m(i)}, l_c^{m(i)})\}_{i=1}^{PS}$ and $D_c^n = \{(f_c^{n(i)}, l_c^{n(i)})\}_{i=1}^{PS}$:

$$distance\left(D_c^m,\ D_c^n\right) = d_f\left(f_c^m, f_c^n\right) + d_l\left(l_c^m, l_c^n\right), \tag{2}$$

where $d_f(f_c^m, f_c^n) = \sum_{i=1}^{PS}(f_c^{m(i)} - f_c^{n(i)})^2$. Since the samples are unlabeled, we ignore the d_l term in the inference.

2.3. Federated-3DA

Fed-3DA deploys a global meta-model (*meta-network*) on the coordinator and a work model (*work-network*) on each client, both of which are deep neural networks. The *meta-network* receives the distribution representation from each client and uses it to produce the weights for the corresponding *work-network*. On the other side, *work-network* is responsible for processing client data. We divide the training stage into continuous periods and vary the data distribution among different periods. The data in period p is denoted as D^p, which is generated by sampling from the available dataset. To estimate the data distribution in each period, we generate a feature vector r to represent the distribution of D^p. Vector r consists of the low-dimensional features extracted from the samples in D^p using t-SNE algorithm [6], as shown in **figure 2**.

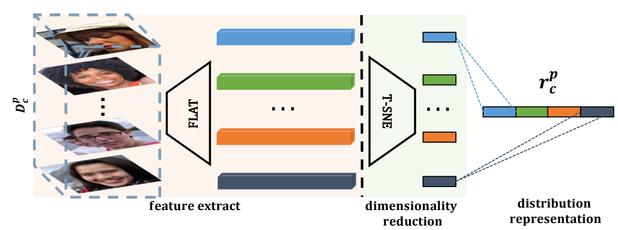


Fig. 2. *Distribution representation of data D_c^p.*

We restrict the distribution of D^p to $\mathcal{P}_{[iid]}$, $\mathcal{P}_{[0.5]}$, $\mathcal{P}_{[0.7]}$, $\mathcal{P}_{[0.8]}$, $\mathcal{P}_{[0.9]}$, $\mathcal{P}_{[0.4,0.4]}$, $\mathcal{P}_{[0.5,0.5]}$, and $\mathcal{P}_{[0.1,0.8]}$. $\mathcal{P}_{[iid]}$ denotes the D^p generated by random sampling from the training or test dataset. $\mathcal{P}_{[0.5]}$ indicates that 50% of the samples in D^p have the same label, and the rest are random. $\mathcal{P}_{[0.1,0.8]}$ indicates that 10% of the samples in D^p have label *A* and 80% have label *B*, while *A* and *B* are randomly selected. The sampling of D^p is shown in **figure 3**.

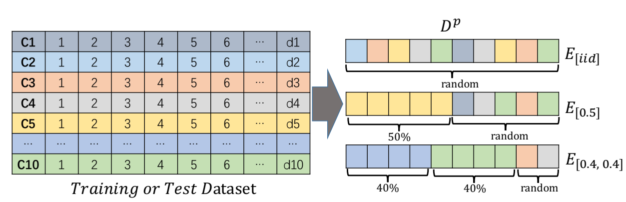


Fig. 3. *The sampling of data D^p.*

In each training period, we randomize the distribution

$\mathcal{P}_c^p$ and generate data D_c^p for the client. The *meta-network* receives the distribution representation r_c^p from client and outputs the weights θ_c^p for the *work-network* (step 4 in **algorithm 1**). Fed-3DA uses the parameter update rule $\Delta\varphi = \alpha\Delta\theta\mathcal{L}_c$ to link the training of *meta-network* and *work-network* on the client side (step 6 in **algorithm 2**). $\Delta\theta$ is the change in local work model parameters after several client training rounds, which benefits the convergence of the *meta-network* [7]. The coordinator optimizes the *meta-network* by synchronously aggregating the gradients $\Delta\varphi$ from the clients (step 6 in **algorithm 1**). After multiple training rounds, we can build a mapping between the distribution representations and their corresponding personalized work models on *meta-network*. The training of Fed-3DA is shown in **figure 1(a)**. In the inference stage, when the amount of data processed reaches period size, the client calculates the distribution distance $d_{p-1}^p = distance(D_c^p, D_c^{p-1})$ with the previous period. We set a hyper-parameter DT as the distribution distance threshold for us to identify the distribution drift. If $d_{p-1}^p >$ DT, then the local data distribution is considered as changed. The client then uploads the current distribution representation r_c^p to the *meta-network* and downloads the updated *work-network* weights θ_c^{p+1} for the next period (see **figure 1(b)**).

Algorithm 1 *Fed-3DA (meta-network)*

Input: *FE*: federated training rounds, *C*: client number. **Output**: *meta-network* weights φ, *work-network* weights θ.
Initialize the *meta-network* with φ^1, random clients' data $\{D_c^1\}_{c=1}^C$.

1: generate the distribution representation $\{r_c^1\}_{c=1}^C$ for $\{D_c^1\}_{c=1}^C$
2: **for** $fe = 1, ...,$ *FE* **do**
3:     **for** $c = 1, ...,$ *C* **do**
4:         $\theta_c^{fe} =$ *meta-network*(φ^{fe}, r_c^{fe})
5:         $\varphi_c^{fe+1}, r_c^{fe+1} \leftarrow \mathbf{FedClientTrain}(\varphi^{fe}, \theta_c^{fe})$
6:     $\varphi^{fe+1} \leftarrow \frac{1}{C}\sum_{c=1}^C \varphi_c^{fe+1}$
7: **while** receive r_c^p from client c **do**
8:     return $\theta_c^{p+1} =$ *meta-network*(φ^{fe}, r_c^p)

2.4. Theoretical Analysis

In Fed-3DA, a multi-task learning is formed among the clients [8]. Let $f(\cdot;\varphi^p)$ denote the *meta-network* parameterized by φ^p, $w(\cdot;\theta_c^p)$ denote the *work-network* parameterized by θ_c^p. Given the distribution representation r_c^p, the *meta-network* outputs the weights θ_c^p for the *work-network*, i.e., θ^p=$f(r_1^p...r_c^p;\ \varphi^p)$, θ^p=$\{\theta_c^p\}_{c=1}^C$. **Equation (1)** can be adjusted to obtain the optimal expression of the DPFL problem:

$$\arg\min_{r_1^p,...,r_c^p,\varphi^p} \frac{1}{P}\sum_{p=1}^{P}\mathcal{L}^p(f(r_1^p,...,r_c^p;\ \varphi^p)) \tag{3}$$

We denote by $\widehat{ER}(\varphi, r)$ the empirical loss of the *meta-network* over all the P periods $\widehat{ER}(\varphi, r) = \frac{1}{PC}\sum_{p=1}^P\sum_{c=1}^C \mathcal{L}_c^p(x_c^p, y_c^p; f(r_c^p;\varphi^p))$ and by $ER(\varphi, r)$ the expected loss $ER(\varphi, r) = \frac{1}{PC}\sum_{p=1}^P\sum_{c=1}^C \mathbb{E}_{(x,y)\sim\mathcal{P}_c^p}[\mathcal{L}_c^p(x, y; f(r_c^p;\varphi^p))]$.

Algorithm 2 *Fed-3DA (work-network)*

Input: *LE*: client training rounds, *DT*: distribution distance threshold, α: federated learning rate, η: client learning rate, *PS*: period size, φ^{fe}: copy of *meta-network* weights, θ_c^{fe}: *work-network* weights, $\mathcal{L}_c$: loss function. **Output**: φ^{fe+1}, distribution representation r_c^{fe+1} and $\hat{y}_c^{p(i)}$.

1: **FedClientTrain**$(\varphi^{fe}, \theta_c^{fe})$**:**
2: set $\hat{\theta} = \theta_c^{fe}$
3: **for** $le = 1, ...,$ *LE* **do**
4:     **for b** in batchs(D_c^{fe}) **do**
5:         $\theta_c^{fe} \leftarrow \theta_c^{fe} - \eta\mathcal{L}_c$
6: $\varphi^{fe+1} \leftarrow \varphi^{fe} - \alpha(\hat{\theta} - \theta_c^{fe})\mathcal{L}_c$
7: random $\mathcal{P}_c^{fe+1}$ and generate $D_c^{fe+1} = \{(x_c^i, y_c^i)\}_{i=1}^{PS}$
8: generate the distribution representation r_c^{fe+1} for D_c^{fe+1}
9: return φ^{fe+1}, r_c^{fe+1}
10: **while** receive $(x_c^{p(i)},\ *)$ **do**
11:     **if** i>*PS* & $distance(D_c^p, D_c^{p-1})$>*DT* **then**
12:         generate the distribution representation r_c^p for D_c^p
13:         $\theta_c^{p+1} =$ *meta-network*(φ^{fe}, r_c^p) and p++
14:         $y_c^{p(i)} =$ *work-network*$(\theta_c^{p+1}, x_c^{p(i)})$ and i++
15:     **else**
16:         $y_c^{p(i)} =$ *work-network*$(\theta_c^p, x_c^{p(i)})$ and i++
17:     return $\hat{y}_c^{p(i)}$

Theorem 1. *Let the space of the meta-network parameter and clients' distribution representation be bounded in a minimum enclosing ball of radius R_e, and the dimension of them are N and E. Let Lip_l, Lip_φ and Lip_r be the Lipschitz constant of functions $\mathcal{L}_c^p(x, y)$, $f(*;\varphi)$ and $f(r;*)$. For all p, ε, δ with $p \in [P]$ and $0 <\varepsilon$, $\delta< 1$, if the period size S satisfies $S \geq \mathbb{O}(\frac{CE+N}{C\varepsilon^2}(\log R_e Lip_l(Lip_\varphi + Lip_r) - \log(\varepsilon\delta)))$, we have with probability at least 1-δ, any φ, r will satisfy $|\widehat{ER}(\varphi, r) - ER(\varphi, r)| \leq \varepsilon$.*

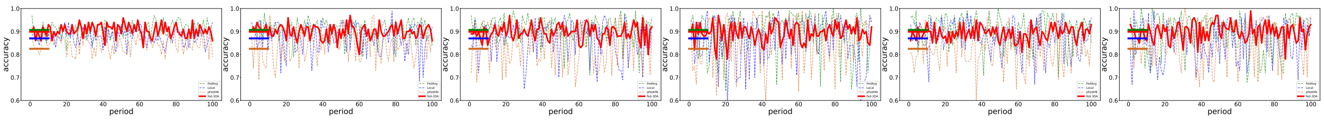
Proof. The detailed proof is omitted due to space limitations. The full proof can be found in appendix [9]. □

3. EXPERIMENTS

Dataset and Metric: We evaluate Fed-3DA using three image classification datasets: CIFAR10, CIFAR100 [10], and Fashion-MNIST [11]. We report the *Accuracy*, *MPV*, and *LAT*, of which the latter two are defined in **equations (4)** and **(5)**. **Baseline:** (a) Local, local training on each client without sharing the gradient. (b) FedAvg [1], a global model trained through the gradient sharing between the clients. (c) pFedHN [12], a personalized FL approach based on *Hypernetwork* that outperforms *Per-FedAvg* [13], *FedPer* [14], *pFedMe* [15].

3.1. MPV and LAT Metrics

We define **"multi-period variance" (MPV)** to measure the fluctuation of the client test accuracy. MPV_n^i represents the



(a) $\mathcal{P}_{[iid]}$ (b) $\mathcal{P}_{[0.5]}$ (c) $\mathcal{P}_{[0.7]}$ (d) $\mathcal{P}_{[0.9]}$ (e) $\mathcal{P}_{[0.5,0.5]}$ (f) $\mathcal{P}_{[0.1,0.8]}$

Fig. 4. *Accuracy evaluated on the Fashion-MNIST*. The validation accuracy is marked with a short line (*FedAvg*: 90.7%, *Fed-3DA*: 89.9%, *Local*: 87.1%, *pFedHN*: 82.5%). The distribution distance threshold (DT) comes from the maximum between the heterogeneous datasets. For more details, see appendix [9].

clients' test accuracy fluctuation in the i^{th} n period. *VACC* denotes the model validation accuracy. $TACC_c^p$ represents the test accuracy of client c in period p.

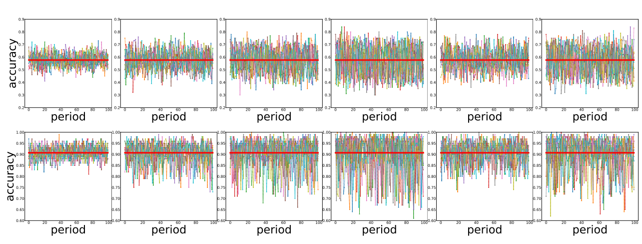
$$MPV_n^i = \frac{1}{n} \sum_{p=(i-1)n+1}^{in} \frac{1}{C} \sum_{c=1}^{C} (TACC_c^p - VACC)^2 \tag{4}$$

We define **"accuracy threshold" (AT)** as 90% of the model validation accuracy and calculate the proportion of clients, whose test accuracy is lower than the threshold, in all periods (denoted as **LAT**). LAT_n^i represents the proportion of the clients that meet the above definition in the i^{th} n period. *VACC* denotes the model validation accuracy.

$$\begin{gathered} LAT_n^i = \frac{1}{nC} \sum_{p=(i-1)n+1}^{in} f(p;\, i, n);\ where \\ f(p;\, i, n) = \begin{cases} 1, & 0 \leq \frac{(VACC - TACC_c^p)}{VACC} \leq AT \\ 0, & else \end{cases} \end{gathered} \tag{5}$$

3.2. Test Accuracy Fluctuation on non-IID Data

We optimize the global model based on the algorithm *FedAvg* [1], the validation accuracy obtained on the CIFAR10 and Fashion-MNIST are: 57.67% and 90.7%. We generate the D^p for testing by random sample from the test set. The test accuracy is shown in **figure 5**. Under the IID distribution (**subfigure a**), the accuracy fluctuates the least. Under the non-IID condition (**subfigure b-f**), the more extreme the distribution type, the greater the accuracy fluctuation.



(a) $\mathcal{P}_{[iid]}$ (b) $\mathcal{P}_{[0.5]}$ (c) $\mathcal{P}_{[0.7]}$ (d) $\mathcal{P}_{[0.9]}$ (e) $\mathcal{P}_{[0.40.4]}$ (f) $\mathcal{P}_{[0.10.8]}$

Fig. 5. *Test accuracy fluctuation*. The validation accuracy is marked with the red line (CIFAR10: first row, Fashion-MNIST: second row, ten clients, period size of 1K).

3.3. Model Adaptability under the Dynamic Distribution

We compare the Fed-3DA with other FL approaches, the results of *MPV* and *LAT* are presented in **table 1**. The test accuracy fluctuations on dataset Fashion-MNIST are shown in **figure 4**. The baselines: local, FedAvg and pFedHN, perform poorly on most tasks, showing the importance of the DPFL. Fed-3DA achieves significant 4.6%-20.1% and 5.3%-26.4% improvements over competing approaches in metrics *MPV* and *LAT* under the different distributions. These results demonstrate that Fed-3DA can effectively control the performance decline and reduce the model volatility.

Table 1. *MPV and LAT evaluation*.

Approach	CIFAR10		CIFAR100		Fashion-MNIST	
	MPV_{100}^1	LAT_{100}^1	MPV_{100}^1	LAT_{100}^1	MPV_{100}^1	LAT_{100}^1
Local$\vert_{[iid]}$	3.16 ± 0.52	20 ± 2	1.58 ± 0.63	31 ± 1	1.11 ± 0.57	**1 ± 1**
Local$\vert_{[0.5]}$	5.71 ± 1.91	27 ± 2	9.49 ± 6.44	49 ± 1	5.33 ± 2.26	9 ± 1
Local$\vert_{[0.7]}$	11.23 ± 3.74	32 ± 1	9.48 ± 1.41	54 ± 2	5.42 ± 0.30	**13 ± 1**
Local$\vert_{[0.8]}$	12.25 ± 1.95	33 ± 2	20.71 ± 10.40	55 ± 3	9.57 ± 5.78	16 ± 1
Local$\vert_{[0.9]}$	18.48 ± 7.29	33 ± 1	20.93 ± 10.74	57 ± 1	10.77 ± 2.07	**18 ± 2**
Local$\vert_{[0.4,0.4]}$	4.70 ± 2.10	31 ± 1	8.43 ± 2.29	49 ± 2	3.82 ± 0.94	10 ± 1
Local$\vert_{[0.5,0.5]}$	10.34 ± 5.42	34 ± 1	10.63 ± 3.36	49 ± 2	4.54 ± 2.73	15 ± 1
Local$\vert_{[0.1,0.8]}$	17.29 ± 3.65	34 ± 2	16.46 ± 8.75	57 ± 3	8.95 ± 3.48	**15 ± 1**
FedAvg$\vert_{[iid]}$	2.56 ± 0.74	17 ± 6	3.07 ± 1.69	33 ± 3	**0.78 ± 0.28**	**1 ± 1**
FedAvg$\vert_{[0.5]}$	6.19 ± 1.67	27 ± 5	9.73 ± 0.89	45 ± 2	3.74 ± 2.68	7 ± 1
FedAvg$\vert_{[0.7]}$	10.97 ± 3.82	31 ± 7	9.30 ± 3.23	48 ± 1	3.42 ± 0.62	14 ± 1
FedAvg$\vert_{[0.8]}$	15.00 ± 1.31	33 ± 5	13.91 ± 7.89	52 ± 3	3.21 ± 1.46	**14 ± 1**
FedAvg$\vert_{[0.9]}$	18.13 ± 3.77	33 ± 6	20.87 ± 3.23	52 ± 2	9.32 ± 4.73	19 ± 2
FedAvg$\vert_{[0.4,0.4]}$	9.59 ± 1.11	26 ± 5	7.00 ± 1.12	45 ± 2	2.64 ± 0.79	**6 ± 1**
FedAvg$\vert_{[0.5,0.5]}$	9.80 ± 2.99	29 ± 5	12.40 ± 5.21	47 ± 2	5.83 ± 0.66	12 ± 1
FedAvg$\vert_{[0.1,0.8]}$	13.97 ± 3.72	32 ± 6	15.61 ± 6.49	50 ± 2	9.23 ± 7.18	19 ± 1
pFedHN$\vert_{[iid]}$	3.50 ± 1.94	30 ± 3	1.66 ± 0.17	39 ± 1	4.08 ± 0.39	8 ± 2
pFedHN$\vert_{[0.5]}$	18.01 ± 3.01	60 ± 1	**6.62 ± 4.13**	69 ± 1	8.50 ± 3.21	16 ± 2
pFedHN$\vert_{[0.7]}$	31.67 ± 16.85	65 ± 2	10.78 ± 1.61	77 ± 1	8.83 ± 2.75	21 ± 1
pFedHN$\vert_{[0.8]}$	50.77 ± 19.15	65 ± 2	37.97 ± 10.02	78 ± 1	32.40 ± 10.21	20 ± 2
pFedHN$\vert_{[0.9]}$	69.14 ± 28.95	65 ± 3	49.23 ± 12.70	78 ± 1	26.00 ± 6.75	25 ± 2
pFedHN$\vert_{[0.4,0.4]}$	26.16 ± 5.42	51 ± 1	6.61 ± 2.31	68 ± 2	8.30 ± 2.54	16 ± 1
pFedHN$\vert_{[0.5,0.5]}$	39.83 ± 2.86	51 ± 3	20.67 ± 2.24	70 ± 2	15.01 ± 2.31	19 ± 1
pFedHN$\vert_{[0.1,0.8]}$	77.64 ± 50.21	64 ± 1	35.55 ± 31.95	74 ± 1	20.21 ± 5.47	22 ± 1
Fed-3DA$\vert_{[iid]}$	**2.05 ± 0.20**	**5 ± 1**	**1.51 ± 0.62**	**20 ± 1**	**0.78 ± 0.19**	**1 ± 1**
Fed-3DA$\vert_{[0.5]}$	**5.56 ± 0.80**	**16 ± 1**	6.84 ± 1.31	**36 ± 1**	**2.13 ± 0.48**	**7 ± 1**
Fed-3DA$\vert_{[0.7]}$	**8.01 ± 0.41**	**21 ± 1**	**8.82 ± 4.43**	**41 ± 3**	**2.83 ± 1.63**	**13 ± 1**
Fed-3DA$\vert_{[0.8]}$	**12.10 ± 4.86**	**25 ± 2**	**13.51 ± 2.48**	**43 ± 2**	**3.11 ± 1.29**	**14 ± 1**
Fed-3DA$\vert_{[0.9]}$	**18.07 ± 9.64**	**27 ± 1**	**20.34 ± 3.49**	**45 ± 2**	**6.77 ± 4.39**	**18 ± 1**
Fed-3DA$\vert_{[0.4,0.4]}$	**4.41 ± 1.26**	**17 ± 1**	**6.53 ± 5.12**	**36 ± 1**	**2.20 ± 1.07**	**6 ± 2**
Fed-3DA$\vert_{[0.5,0.5]}$	**9.49 ± 1.12**	**21 ± 1**	**10.46 ± 8.69**	**37 ± 2**	**3.94 ± 0.72**	**11 ± 1**
Fed-3DA$\vert_{[0.1,0.8]}$	**12.67 ± 6.05**	**25 ± 1**	**15.46 ± 5.72**	**44 ± 1**	**4.01 ± 3.63**	**15 ± 1**

4. CONCLUSION

We formulate the DPFL problem and propose a new method named Fed-3DA, which can automatically detect the data distribution drift during the inference stage without relying on real-time labels. Besides, Fed-3DA can update the working models efficiently during run-time. Based on experiments, we show that Fed-3DA outperforms the baseline approaches.

5. REFERENCES

[1] 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*. PMLR, 2017, pp. 1273–1282.

[2] Laura Rieger, Rasmus M Th Høegh, and Lars K Hansen, "Client adaptation improves federated learning with simulated non-iid clients," *arXiv preprint arXiv:2007.04806*, 2020.

[3] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang, "Personalized cross-silo federated learning on non-iid data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, vol. 35, pp. 7865–7873.

[4] Jason Altschuler, Jonathan Weed, and Philippe Rigollet, "Near-linear time approximation algorithms for optimal transport via sinkhorn iteration," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 1961–1971.

[5] David Alvarez Melis and Nicolo Fusi, "Geometric dataset distances via optimal transport," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[6] Laurens Van der Maaten and Geoffrey Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[7] Zhouyuan Huo, Qian Yang, Bin Gu, Lawrence Carin, and Heng Huang, "Faster on-device training using new federated momentum algorithm," *ArXiv*, vol. abs/2002.02090, 2020.

[8] Jonathan Baxter, "A model of inductive bias learning," *Journal of artificial intelligence research*, vol. 12, pp. 149–198, 2000.

[9] Anonymous, "Appendix," Website, 2022, `https://anonymous.4open.science/r/Fed-3DA-ICASSP2023-23C0/Appendix.pdf`.

[10] Alex Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[11] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017.

[12] Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik, "Personalized federated learning using hypernetworks," *arXiv preprint arXiv:2103.04628*, 2021.

[13] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar, "Personalized federated learning: A meta-learning approach," *arXiv preprint arXiv:2002.07948*, 2020.

[14] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary, "Federated learning with personalization layers," *arXiv preprint arXiv:1912.00818*, 2019.

[15] Canh T Dinh, Nguyen H Tran, and Tuan Dung Nguyen, "Personalized federated learning with moreau envelopes," *arXiv preprint arXiv:2006.08848*, 2020.