A Federated Learning Mechanism with Feature Drift for Feature Distribution Skew

1st Jihao Yang
School of Electronics and Information
Northwestern Polytechnical University
Xi'an, China
yangjihao@mail.nwpu.edu.cn

2nd Xinyang Deng*
School of Electronics and Information
Northwestern Polytechnical University
Xi'an, China
xinyang.deng@nwpu.edu.cn

3rd Laisen Nie
School of Electronics and Information
Northwestern Polytechnical University
Xi'an, China
nielaisen@nwpu.edu.cn

4th Wen Jiang
School of Electronics and Information
Northwestern Polytechnical University
Xi'an, China
jiangwen@nwpu.edu.cn

***Abstract*—Federated learning is a nascent distributed machine learning paradigm that enables multiple clients to collaborate in training a model for a specific task under the coordination of a central server, all while safeguarding the privacy of the user's local data. Nevertheless, the constraint that distributed datasets must remain within local nodes introduces data heterogeneity in federated learning training. In this paper, we focus on how to mitigate the damage caused by the data heterogeneity of feature distribution skew in federated learning models during training. To achieve this goal, we propose a feature drift-corrected federated learning algorithm. We design a feature drift variable derived from the local models of clients and the global model of the server. This variable is incorporated into the client's local loss function to rectify local model parameters. Additionally, we utilize the disparity between the global models before and after to regulate the local model. Validation experiments are conducted on multiple datasets exhibiting feature distribution skew. The implementation results demonstrate the efficacy of our approach in significantly enhancing the model performance of federated learning under feature distribution skew.**

***Index Terms*—Federated learning, Data heterogeneity, Feature distribution skew, Model correction.**

I. Introduction

Federated learning (FL) is a burgeoning machine learning paradigm designed to enable model training without compromising the confidentiality of local private data. As contemporary society places increasing emphasis on safeguarding personal and corporate private data, FL, distinguished by its privacy and security features, has garnered growing attention [1]. In 2017, Google introduced the pioneering FL algorithm, FedAvg [2], establishing a robust foundation for subsequent FL research.

The classical FL system consists of several local clients and a central server, with each client functioning as a node in this distributed system [3]. Communication is restricted solely to interactions between clients and the central server, precluding direct communication among clients. Each client possesses its own local private data. At the beginning of training, the server dispatches an initialized global model to each client. Subsequently, the client uses its local private data for training and uploads either the local model parameters or parameter updates to the server upon completion of the training, and the server executes an information fusion process (called aggregation). The aggregated global model is then transmitted back to the client, marking the commencement of the subsequent training phase [4]. Throughout this process, the client's data is not allowed to leave the local area to ensure the privacy and security of the client's local data.

Clients in FL systems exclusively train with their respective local data, and significant variations exist among different clients' local datasets stemming from factors such as user preferences, geographic locations, and device differences. It constitutes a non-independent and identically distributed (non-iid) global dataset. This data heterogeneity is the main reason for the poor performance of FL models. According to the study in [5], data heterogeneity is generally categorized into three types: label distribution skew, feature distribution skew, and quantity skew. In this paper, we focus on feature distribution skew.

Feature distribution skew is a type of data heterogeneity that widely exists in the real world, which is mainly characterized by the fact that local data from different clients share the same set of labels, but samples with identical labels exhibit distinct features due to various factors. For example, in the field of handwriting recognition, users writing the same words may produce varied features owing to differences in writing habits, stroke order, and other individual nuances [5]. In medical images, the use of diverse imaging machines and protocols across different hospitals results in substantial variations in scanned images of the same individual due to varying scanning intensities or contrasts. When multiple hospitals want to

This work was partially supported by the National Natural Science Foundation of China under Grant 62171378 and 62173272.

*Corresponding author

collaborate on training a medical image classification model, the local data of these hospitals cannot be trained centrally due to the protection of patients' private data by law or regulation, and thus an FL-based approach is used. However, such non-iid data with feature distribution skew can lead to a trained model with low classification accuracy, making it unusable.

To address the data heterogeneity issue associated with feature distribution skew, numerous researchers have proposed innovative solutions. FedFA [6] enhances the feature statistics of each sample in a probabilistic manner through a multivariate Gaussian distribution and synthesizes new statistics for the purpose of federated feature augmentation. HarmoFL [7] adopts a strategy of transforming medical images into the frequency domain and normalizing their amplitudes to simulate a standardized imaging setup, thereby alleviating feature bias in the medical domain. FedBN [8] makes it possible for locally trained parameters not to be assimilated in the model aggregation phase, thus degrading model performance, by preserving the Batch Normalization (BN) layer parameters that contain the local personalization of the client when the server performs model aggregation. However, while FedBN considers guaranteeing the local personalization of the client, it neglects the fact that data with similar features can be borrowed from each other to improve the learning ability of the local model for the samples.

Based on these insights, we propose an innovative **Fed**erated learning algorithm with **F**eature **D**rift to mitigate model performance degradation caused by feature distribution skew (FedFD). Inspired by the concept of client drift [9], we define a client-side feature drift variable encapsulating information about the local model parameters of other clients. This variable is employed to correct the local model for the purpose of explicitly learning features from other data. Furthermore, we use the difference between the two global models before and after to impose constraints on the local model. The main contributions of this paper are as follows:

- We define the feature drift variable by considering the model parameter discrepancies among clients and the global model parameters of the server. This enables us to glean data feature information embedded in the model parameters of other clients.
- Building upon client-side drift variables and disparities in global model parameters, we introduce a novel client-side local objective function. This function serves to correct and generate personalized local models, resulting in a substantial enhancement of overall model performance.
- Extensive experiments on multiple datasets exhibiting feature distribution skew demonstrate the effectiveness of our approach in alleviating the adverse impacts of data heterogeneity with feature distribution skew on model training.

The remainder of this paper is organized as follows. Section II describes our methodology. Experimental results and analysis are presented in Section III. Section IV summarizes our work.

II. Our Methodology

In this paper, we present an innovative federated learning algorithm, which corrects for feature drift, aiming to mitigate the performance degradation of federated learning models induced by feature distribution skew. Firstly, we define the feature drift variable and use it to correct the local loss function. Secondly, we fine-tune the local model parameters by iteratively learning the disparity between the global model parameters before and after. Finally, comparative experiments on multiple datasets validate the effectiveness of our proposed method.

A traditional FL system consists of N local clients and a central server. Client i ($i \in [N]$) performs local training using its private data $D_i = \{(\boldsymbol{x_i}, y_i)\}$, and uploads the trained model to the central server for aggregation. The server sends the aggregated global model down to all clients, thus starting the next round of training. The purpose of FL is to train a global model w^* among multiple clients collaboratively, while ensuring that each client's private data does not leave the local domain, i.e:

$$w^* = \arg\min_w F(w) = \sum_{i=1}^{N} p_i F_i(w), \tag{1}$$

where w is the global model parameter after aggregation by the central server, p_i is the aggregation weight of each client, $p_i > 0$ and $\sum_i p_i = 1$. $F(w)$ and $F_i(w)$ are the global and local objective functions, respectively. In general, $F_i(w)$ is defined as the classical cross-entropy loss function, i.e., $F_i(w) := \mathbb{E}_{\boldsymbol{x_i} \sim P_i}\left[\ell_i\left(w; \boldsymbol{x_i}, y_i\right)\right]$, where P_i is the local data distribution of client i. In general, we set $p_i = \frac{|D_i|}{n}$, where $n = \sum_i |D_i|$ is the total number of samples.

In FL, clients' local datasets are generally non-iid to each other, that is $P_i \neq P_j$ for different clients i and j. Rewriting $P_i(\boldsymbol{x}, y)$ as $P_i(y|\boldsymbol{x})P_i(\boldsymbol{x})$, feature distribution skew means that the marginal distributions $P_i(\boldsymbol{x})$ varies across clients, even if $P_i(y|\boldsymbol{x})$ is shared [5].

A. Architecture and Workflow of FedFD

The architecture and workflow of our approach are illustrated in Fig. 1. Each client's local dataset constitutes a feature distribution skew among them. In the training start phase, the central server sends initialized model parameters to each client. Subsequently, clients use their local private data to perform local training for several epochs and upload the trained local model to the central server. After receiving the local model from each client, the central server first calculates the feature drift variables for each client, then aggregates the local models to obtain the global model, and finally sends the updated global model and client drift variables to the corresponding clients. The above process is repeated until the model converges or the maximum number of communication rounds is reached.

B. Details of FedFD

Due to the existence of feature distribution skew, local data from distinct clients encompasses samples with identical labels

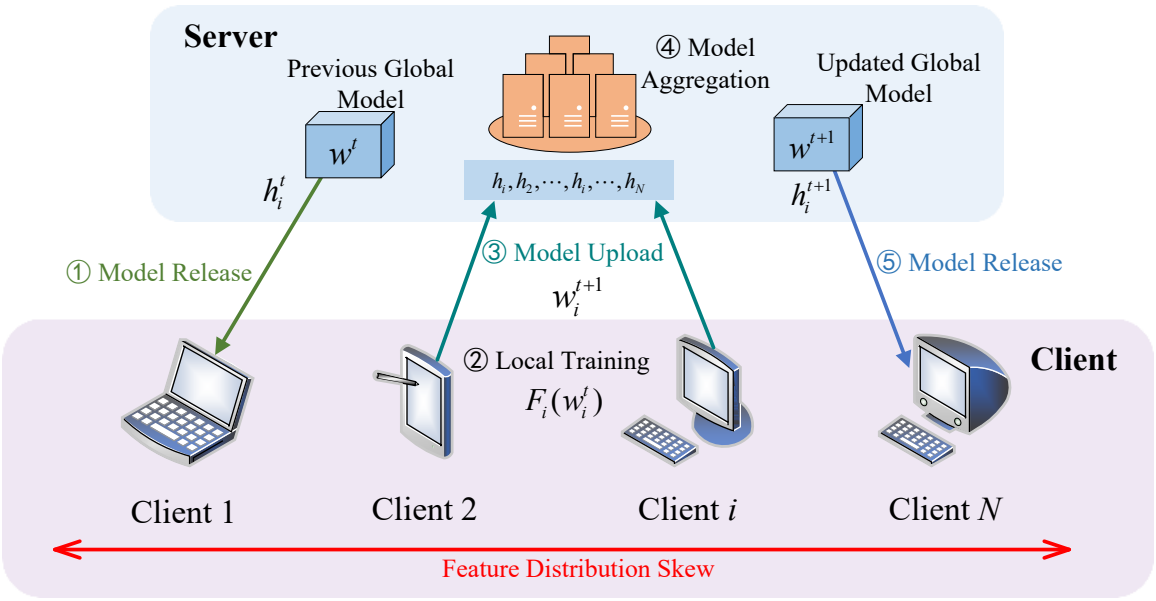


Fig. 1. Architecture and workflow of our FedFD.

but varying features. In order to mitigate the adverse effects of this heterogeneity, inspired by the concept of client drift [9], we define a feature drift variable h_i for each client in FedFD. We believe that the difference among the local models of other clients and the global model should be included in h_i, and therefore, h_i is defined as follows:

$$h_i = \sum_{j=1.j\neq i}^{N} c_j(w - w_j), \tag{2}$$

where c_j the weighting coefficient, $c_j = \frac{p_j}{\sum_{j=1,j\neq i}^{N} p_i}$, w_j is the parameter of client j's local model, and w is the parameter of the global model. From (2), it can be seen that h_i counts the model drifts of all other clients except client i, and contains the feature information of the local data of other clients in this communication round.

Moreover, considering that the global model parameters encapsulate training information from the entire dataset, we incorporate differences between global models as penalty terms to adjust the local model parameters. Consequently, the local objective function of FedFD comprises three components: the local empirical loss term, the feature drift correction term, and the global model penalty term. Equation (1) is rewritten as follows:

$$F_i(w) = L_{CE} + \frac{\alpha}{2}\|w - w_i - h_i\|^2 + \beta\|w_i - \Delta w\|^2, \tag{3}$$

where L_{CE} is the classical cross-entropy loss function, w_i is the local model parameter of client i, w is the global model parameter, and α and β are the weight coefficients of the feature drift correction term and the global model penalty term, respectively. In the t-th communication round, $\Delta w = w^t - w^{t-1}$. The details of FedFD are summarized in Algorithm 1, where $w_{i,l}^{t+1}$ is the model parameter of the l-th layer after client i performs local updates.

Algorithm 1 FedFD

Input: local dataset D_i, number of clients N, communication rounds T, number of local epochs K, learning rate η
Output: the final model w^T and $\{w_i^T | i \in [N]\}$

Server executes:
initialize w^0, h_i^0
for $t = 0, 1, \ldots, T-1$ **do**
  for each client i **in parallel do**
    compute feature drift variable h_i^t by (2)
    send the global model w^t and h_i^t to client i
    $w_i^{t+1} \leftarrow$ **LocalTraining** (i, w^t, h_i^t)
  end for
  for each client i and each layer l **do**
    if layer l is not BatchNorm **then**
      $w_l^{t+1} = \frac{1}{N}\sum_{i\in[N]} w_{i,l}^{t+1}$
    end if
  end for
end for
return w^T and $\{w_i^T | i \in [N]\}$

Client executes:
LocalTraining(i, w^t, h_i^t):
$w_i^t \leftarrow w^t$
for epoch $k = 1, 2, \ldots, K$ **do**
  for each batch $\boldsymbol{b}$ of D_i **do**
    $w_i^{t+1} \leftarrow w_i^t - \eta\nabla F_i(w_i^t; h_i^t, \boldsymbol{b})$
  end for
end for
return w_i^{t+1} **to the server**

As shown in Algorithm 1, the central server first initializes the global model parameters and feature drift variables (line 2). After receiving the local model parameters, the server calculates the feature drift variables h_i for each client using (2) and uses them for the next execution of local training for client i (lines 5 and 7). When performing global aggregation, FedFD uses the same aggregation method as FedBN [8] to preserve the independent individuality of each client and improve the performance of the client's local model. At the client level, FedFD is similar to classic FL algorithms such as FedAvg and FedProx, and performs only the most basic local deep learning training to reduce the computational overhead of the client. In terms of communication overhead, FedFD simply sends the feature drift variable h_i at the same time the server transmits the aggregated global model to the client.

C. Discussion

Aiming at the difficulty of the classical FL algorithm to handle the feature distribution skew in non-iid scenarios, our FedFD proposes an effective loss function to enhance the model's performance. First, we define the feature drift variable to characterize the feature differences among the local datasets of different clients in a single communication round. Due to the diversity in feature from local datasets, different clients exhibit significant disparities in the direction of the model's gradient descent during training. However, since these samples share identical labels, the overall direction of the model gradient descent remains similar, enabling mutual learning to enhance the performance of the local model. Drawing inspiration from the concept of information fusion, we use (2) to compute the feature drift variable of each client and incorporate it into the local loss function, which motivates the local model to learn the model parameters of other clients with similar datasets, thus improving the capability of discriminating the local samples.

Second, the direction of the global model parameters contains information about the model's training on the global dataset, so we correct the local model parameters by using the differences between the global models as a penalty term. The advantage of this approach is that the local model can acquire more comprehensive feature information from the global model parameters. As evident in Algorithm 1, the local training process on the client side is mainly adapted to the local objective function in (3), which improves the performance of the local model without increasing the computational overhead of the local device too much. For the server, the primary increase in computational cost compared to FedAvg arises from the computation of drift variables after receiving the local model parameters. However, when weighed against the notable improvement in model performance, the cost of adding a modest amount of server computation is deemed entirely acceptable.

III. Experiments

In this section, we first describe the experimental setup, and then conduct extensive experiments on multiple datasets to validate the effectiveness of our method for scenarios with feature distribution skew.

TABLE I
AVERAGE TEST ACCURACY (%) WITH DIFFERENT HYPERPARAMETER α AND β ON DIGIT5

α \ β	0.1	0.05	0.01	0.005	0.001
0.1	83.17	83.93	84.66	84.60	84.97
0.05	84.50	84.61	85.10	85.50	85.32
0.01	86.02	86.27	86.00	85.93	85.61
0.005	86.11	86.73	86.34	86.17	85.52
0.001	86.99	**87.61**	86.06	85.74	85.34

TABLE II
AVERAGE TEST ACCURACY (%) WITH DIFFERENT HYPERPARAMETER α AND β ON OFFICE-CALTECH10

α \ β	0.1	0.05	0.01	0.005	0.001
0.1	67.74	70.04	71.59	71.70	72.32
0.05	68.04	70.19	71.10	73.12	72.67
0.01	69.41	71.19	72.65	72.88	73.78
0.005	71.49	71.91	72.83	**73.94**	73.14
0.001	73.18	73.12	73.36	72.39	72.83

TABLE III
AVERAGE TEST ACCURACY (%) WITH DIFFERENT HYPERPARAMETER α AND β ON DOMAINNET

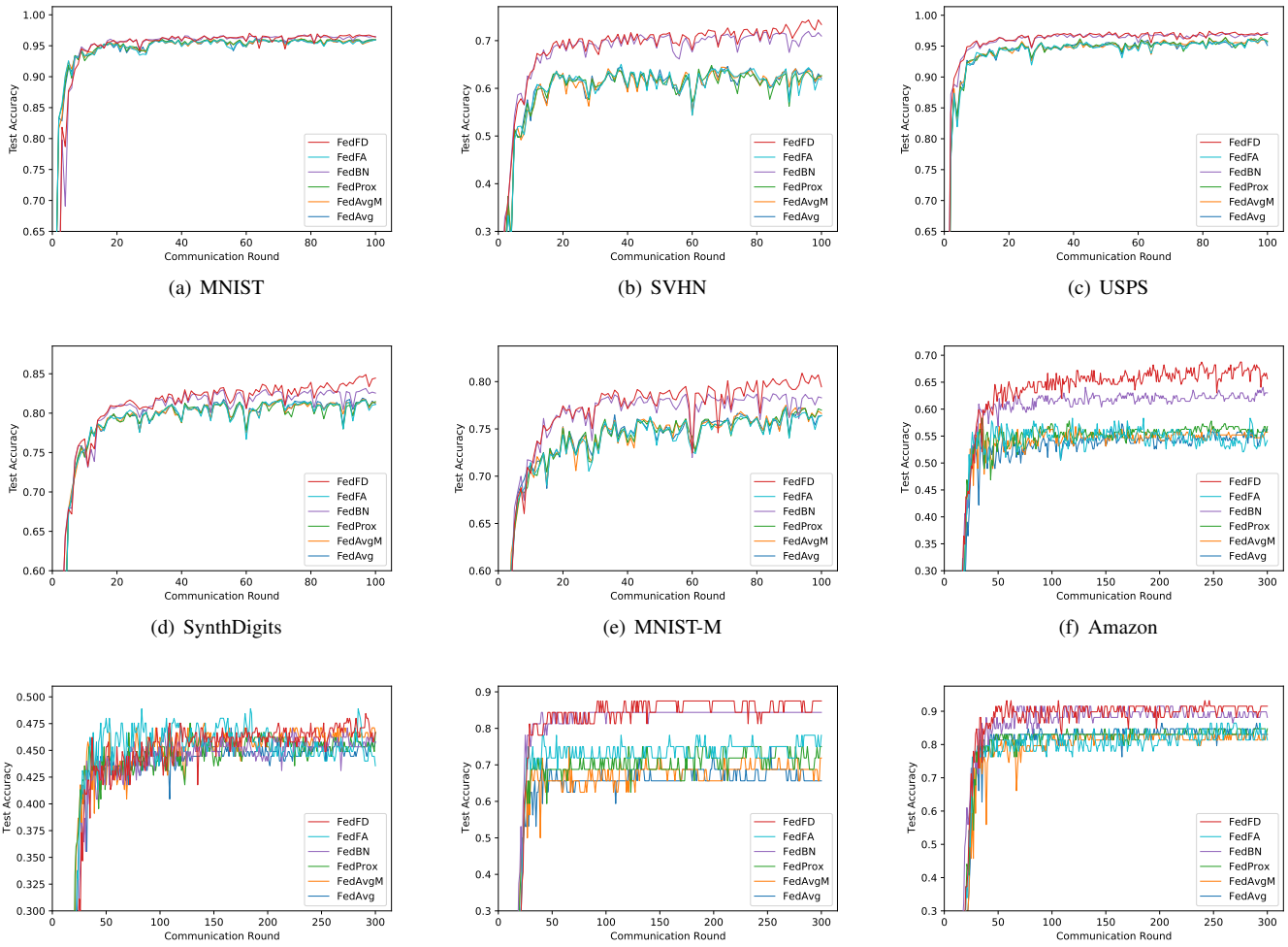
α \ β	0.1	0.05	0.01	0.005	0.001
0.1	-	-	-	-	45.85
0.05	-	-	-	46.66	47.27
0.01	-	-	48.55	48.70	48.92
0.005	-	45.38	48.55	**49.71**	48.94
0.001	44.67	46.77	48.88	49.40	49.46

TABLE IV
TEST ACCURACY (%) WITH DIFFERENT METHODS ON DIGIT5

Method	MNIST	SVHN	USPS	Syn	M-M	Average
FedAvg	95.81	64.59	95.65	81.73	76.15	82.79
FedAvgM	95.94	64.12	95.54	81.22	77.24	82.81
FedProx	95.95	63.54	96.02	81.36	77.17	82.81
FedBN	96.65	71.89	96.61	82.97	78.69	85.36
FedFA	95.69	64.11	95.54	81.00	77.33	82.73
FedFD	**96.69**	**76.13**	**96.83**	**85.16**	**83.23**	**87.61**

A. Experimental Setup

We conducted experiments on three feature distribution skew datasets: Digit5 [8], Office-Caltech10 [10], and DomainNet [11]. All three datasets consist of samples from different domains that have different features but the same labels and label distributions. Digit5 consists of five datasets: MNIST [12], SVHN [13], USPS [14], SynthDigits [15], and MNIST-M [15]. Office-Caltech10 consists of data from four domains: Caltech, Amazon, Webcam, and DSLR. DomainNet consists of data from six domains: Clipart, Infograph, Painting, Quickdraw, Real, and Sketch. In all the following experiments,



(a) MNIST (b) SVHN (c) USPS

(d) SynthDigits (e) MNIST-M (f) Amazon

(g) Caltech (h) DSLR (i) Webcam

Fig. 2. Test accuracy for FedFD and other baselines on different datasets from Digit5 and Office-Caltech10.

the number of domains in these three datasets is the number of clients in FL.

For Digit5, our client model uses a convolutional neural network (CNN) consisting of three convolutional layers and two fully connected layers, where BN layers are added following each feature extraction layer. For Office-Caltech10 and DomainNet, we adopt AlexNet [16] following FedFA [6]. For the local models, we use the SGD optimizer and set the learning rate to 0.01, the batch size to 32, and the local epoch to 1. For the hyperparameters of the comparison experiments, we set the hyperparameter $\mu = 0.01$ for FedProx in Digit5 and $\mu = 0.001$ for the other two datasets. We set the number of communication rounds to 100 for Digit5 experiments, and 300 for Office-Caltech10 and DomainNet experiments. All these settings are the same as in FedBN [8].

B. Hyperparameter Experiments

In FedFD, we explored hyperparameters α and β in the range $[0.001, 0.005, 0.01, 0.05, 0.1]$. We show the results of the hyperparameter experiments in Tables I, II and III. The experimental metrics are the average test accuracy over multiple domains for each dataset. Ultimately, based on the results of these experiments, we set the hyperparameters of FedFD $\alpha = 0.001, \beta = 0.05$ for Digit5, and $\alpha = 0.005, \beta = 0.005$ for Office-Caltech10 and DomainNet.

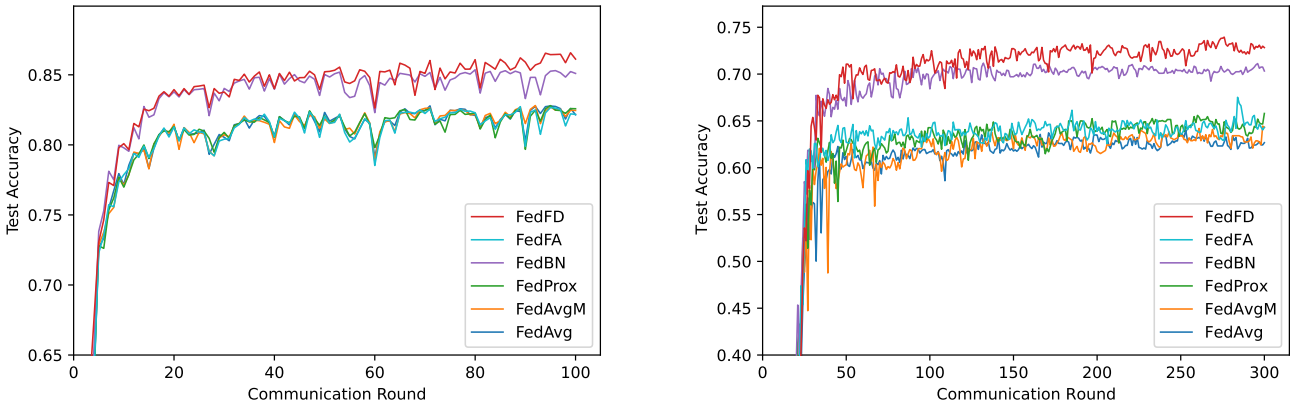
C. Comparison Experiments

For a comprehensive evaluation, we compare FedFD with several state-of-the-art FL algorithms, including FedAvg [2], FedAvgM [17], FedProx [18], FedBN [8], and FedFA [6]. FedAvgM introduces momentum to update the weights on top of FedAvg, and FedProx adds proximal terms to constrain the local model. FedBN and FedFA propose different solutions to the feature distribution skew problem, FedBN achieves personalized FL by retaining the BN layer parameters of each client on the server side, while FedFA improves the performance of the model through federated feature augmentation.

Benchmark digits classification task. We first perform a comparison experiment on Digit5 and report the top-1 test accuracies of FedFD against the other baselines in Table IV,

TABLE V
TEST ACCURACY (%) WITH DIFFERENT METHODS ON OFFICE-CALTECH10 AND DOMAINNET

Method	Office-Caltech10					DomainNet						
	A	C	D	W	Average	C	I	P	Q	R	S	Average
FedAvg	55.73	45.78	75.00	79.66	64.04	48.80	24.90	36.50	56.10	46.30	36.60	41.53
FedAvgM	55.73	47.11	71.88	83.05	64.44	51.90	26.03	38.61	60.80	46.10	36.56	43.17
FedProx	56.67	46.67	75.00	84.75	65.80	48.90	24.90	36.60	54.40	47.80	36.90	41.58
FedBN	63.02	45.78	84.38	91.53	71.17	51.20	26.80	41.50	71.30	54.80	42.10	47.95
FedFA	58.33	**48.89**	78.12	84.75	67.52	51.14	23.59	39.10	58.20	47.82	39.89	43.29
FedFD	**68.75**	48.00	**87.50**	**93.22**	**73.94**	**53.80**	**28.77**	**42.97**	**71.70**	**58.59**	**42.42**	**49.71**



(a) Digit5 (b) Office-Caltech10

Fig. 3. Test accuracy for FedFD and other baselines on Digit5 and Office-Caltech10.

including the test accuracies on each domain and the average test accuracies over the five domains. Note that "Syn" in Table IV refers to "SynthDigits" and "MM" is an abbreviation for "MNIST-M".

As can be seen in Table IV, FedFD not only achieves the best test accuracy in all five domains, but also in the domain average test accuracy. In the domain average test accuracy, FedFD achieves 87.61%, which is 2.25% higher than the second highest, FedBN, and 4.83% higher than the average of the other four baselines. This indicates that FedFD's idea of improving model performance by correcting feature drift is effective. In addition, the test accuracies on the five domains also illustrate the excellence of FedFD. For simpler digital classification tasks, such as MNIST and USPS, FedFD does not provide a significant improvement in model performance. Instead, on SynthDigits, FedFD outperforms the other baselines by an average of 3.50%, and on MNIST-M, FedFD outperforms FedAvg by 7.08%, compared to only 2.54% for FedBN. On the most difficult SVHN, FedFD can reach 76.13%, which is 4.24% higher than the next best FedBN and 12.04% higher on average than the other four methods. All these experimental results fully illustrate the superiority of FedFD in dealing with heterogeneous datasets with feature distribution skew, and we analyze that FedFD improves the recognition of local samples by learning feature information from model parameters from other datasets, and thus improves the recognition of local samples. In addition, we show the test accuracy curves of FedFD and the other five baseline methods with respect to the number of communication rounds on the Digit5 dataset in Fig. 2(a)-2(e) and Fig. 3(a). It can be seen that MNIST and USPS are not significantly different due to their own overprecision, and that FedFD has a significant advantage over the other baselines after about 80 communication rounds on the SVHN, SynthDigits, MNIST-M, and domain averaging tests.

Real-world dataset experiments. To better understand how our proposed algorithm gains in real-world heterogeneous data with feature distribution skew, we validate the effectiveness of FedFD compared to other methods on two real-world datasets: the image classification dataset Office-Caltech10 using images acquired in different cameras or environments, and the image classification dataset DomainNet with different image styles. Table V reports the top-1 test accuracies on these two datasets, where A, C, D, and W are the domain names of Office-Caltech10 "Amazon", "Caltech", "DSLR" , and "Webcam", and C, I, P, Q, R, and S stand for the DomainNet domains "Clipart", "Infograph", "Painting", "Quickdraw", "Real", and "Sketch", respectively.

For Office-Caltech10, as shown in Table V, the average test accuracy of FedFD on the four domains is significantly better than the other methods, which is 9.9% higher than FedAvg and 6.42% higher than FedFA. The model performance on the four domains is also only 0.89% lower than FedFA on Caltech, and the test accuracy on the other three domains

has a significant advantage. For example, the test accuracy of FedFD on Amazon is 5.73% higher than the second highest FedBN and 13.02% higher than FedAvg. The test accuracy of FedFD on DSLR is 15.62% and 12.5% higher than FedAvgM and FedProx, respectively. These results show that FedFD is still robust to real-world feature distribution skew datasets and can achieve better model performance. Such results are attributed to the fact that FedFD can effectively deal with the problem of feature distribution skew among clients with the ability to learn from data and model parameters from other domains. In Fig. 2(f)-2(i) and Fig. 3(b), we simultaneously visualize the individual test accuracy curves for each domain and the domain-averaged test accuracy curves for FedFD and the other five baseline methods as a function of the number of communication rounds on the Office-Caltech10 dataset. It is clear from the figures that FedFD performs poorly only on Caltech, while it achieves remarkable success on the other three domains, especially on Amazon, where FedFD achieves excellent classification performance after about 50 communication rounds. In the domain averaging test, FedFD also achieves better model performance than other methods after about 100 communication rounds.

For DomainNet, as shown in Table V, FedFD also achieves the best model performance in the test. On the domain average test accuracy, FedFD is 1.76% higher than FedBN and 6.42% higher than FedFA. From the test results on all six domains, FedFD provides a good performance gain compared to FedAvg, with the largest increase obtained on Quickdraw at 15.60%, followed by Real at 12.29%. These results likewise illustrate the effectiveness of FedFD on real dataset DomainNet. It is encouraging to see that FedFD not only performs well on the benchmark dataset Digit5 but also generalizes well to real-world datasets. This suggests that FedFD can be widely used in domains such as healthcare, insurance, and finance, where privacy is highly scrutinized and data silos are prevalent.

IV. Conclusion and Future Work

In this paper, we propose a novel client-side feature drift-corrected federated learning algorithm designed to address the challenge of feature distribution skew in data-heterogeneous federated learning. First, we define the feature drift variable based on the client's local model parameters and global model parameters and add it to the local objective function to correct the local model parameters. Second, we use the global model parameter differences to explicitly update the local model, and leverage the differences in global model parameters to explicitly update the local model, ensuring it remains aligned with the global model and does not deviate excessively. Extensive experiments on multiple datasets demonstrate that our method can effectively mitigate the problem of model performance degradation caused by feature distribution skew.

We envision potential enhancements for FedFD in the future through the following avenues. Firstly, FedFD currently does not show accelerated convergence speed, implying a limited improvement in communication overhead. Exploring strategies to expedite FedFD's convergence and minimize the number of communication rounds will be a key focus of our future work. Secondly, we contemplate whether a more precise representation of feature drift exists, allowing FedFD to acquire more accurate model parameters and, consequently, significantly enhance its model performance in the presence of feature distribution skew. This area also warrants further investigation.

References

[1] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3347–3366, 2023.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and A. y. B. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 54, pp. 1273–1282, 2017.

[3] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 12:1–12:19, 2019.

[4] T. Li, K. A. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[5] P. Kairouz, H. B. McMahan, B. Avent, and et al., "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1-2, pp. 1–210, 2021.

[6] T. Zhou and E. Konukoglu, "FedFA: Federated feature augmentation," in *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023.

[7] M. Jiang, Z. Wang, and Q. Dou, "HarmoFL: harmonizing local and global drifts in federated learning on heterogeneous medical images," in *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, vol. 36, pp. 1087–1095, 2022.

[8] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "FedBN: Federated learning on non-iid features via local batch normalization," in *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021.

[9] P. S. Karimireddy, S. Kale, M. Mohri, J. S. Reddi, U. S. Stich, and T. A. Suresh, "SCAFFOLD: stochastic controlled averaging for on-device federated learning," in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, vol. 119, pp. 5132–5143, 2020.

[10] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2066–2073, 2012.

[11] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1406–1415, 2019.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[13] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and Y. A. Ng, "Reading digits in natural images with unsupervised feature learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[14] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 139, no. 5, pp. 550–554, 1994.

[15] Y. Ganin and S. V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proceedings of the 32th International Conference on Machine Learning (ICML)*, vol. 37, pp. 1180–1189, 2015.

[16] A. Krizhevsky, I. Sutskever, and E. G. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[17] H. T. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *CoRR*, vol. abs/1909.06335, 2019.

[18] T. Li, K. A. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems (MLSys)*, 2020.