Pattern Recognition 149 (2024) 110230

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

FedNN: Federated learning on concept drift data using weight and adaptive group normalizations

Myeongkyun Kang [a], Soopil Kim [a], Kyong Hwan Jin [b], Ehsan Adeli [c,d], Kilian M. Pohl [c], Sang Hyun Park [a,e,*]

[a] Department of Robotics and Mechatronics Engineering, Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu, South Korea
[b] School of Electrical Engineering, Korea University, Seoul, South Korea
[c] Department of Psychiatry and Behavioral Sciences, Stanford University, Stanford, United States of America
[d] Department of Computer Science, Stanford University, Stanford, United States of America
[e] AI Graduate School, Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu, South Korea

ARTICLE INFO

Keywords:
Federated learning
Concept drift
Weight normalization
Adaptive group normalization

ABSTRACT

Federated Learning (FL) allows a global model to be trained without sharing private raw data. The major challenge in FL is client-wise data heterogeneity leading to different model convergence speed and accuracy. Despite the recent progress of FL, most methods verify their accuracy on prior probability shift (label distribution skew) dataset, while the concept drift problem (i.e., where each client has distinct styles of input while sharing the same labels) has not been explored. In real scenarios, concept drift is of paramount concern in FL since the client's data is collected under extremely different conditions making FL optimization more challenging. Significant differences in inputs among clients exacerbate the heterogeneity of clients' parameters compared to prior probability shift, ultimately resulting in failures for previous FL approaches. To address the challenge of concept drift, we use Weight Normalization (WN) and Adaptive Group Normalization (AGN) to alleviate conflicts during global model updates. WN re-parameterizes weights to have zero mean and unit variance while AGN adaptively selects the optimal mean and standard deviation for feature normalization based on the dataset. These two components significantly contribute to having consistent activations after global model updates reducing heterogeneity in concept drift data. Comprehensive experiments on seven datasets (with concept drift) demonstrate that our method outperforms five state-of-the-art FL methods and shows faster convergence speed compared to the previous FL methods.

1. Introduction

Federated Learning (FL) [1] is an emerging distributed learning paradigm that isolates client data to fulfill basic privacy requirements [2]. The pioneering work FedAvg [3] performs FL by averaging the client's parameters over multiple communication rounds to consider privacy and communication constraints. During FL, only parameters are sent from the client to the central server allowing the training of models that represent all clients without exposing any private data. Since each client has to be trained with respect to different data distributions, the increase in heterogeneity between clients poses more difficulties in global model training. Though FedAvg has achieved notable empirical success, the variability in real-world FL scenarios [4] ultimately degrades global model convergence and accuracy [5].

Recently, a series of FL models have been proposed, aiming to achieve an accurate global model even when heterogeneous clients participate in training [6]. These methods use various regularization [7] and gradient correction terms [8] to handle non-Independent Identically Distributed (non-IID) FL problems. These optimization-based approaches constrain each trained client parameter to be consistent towards global optimum, ultimately mitigating heterogeneity between clients and achieving a high-accuracy global model. However, most of these studies focus on evaluating model performance under prior probability shift *e.g.,* Dirichlet distribution [9], where each client has a different label distribution as shown in Fig. 1(a).

With *prior probability shift*, the non-IID in FL contains *covariate shift*, *concept drift*, and *concept shift* [10]. *Covariate shift* exhibits marginal distributional differences in visual features across clients *e.g.,* stroke variations in handwritten digits. *Concept drift* exhibits significant differences (*e.g.,* in image styles) between clients *e.g.,* large visual differences between desert roads and snowfield roads. *Concept shift* involves clients

* Corresponding author.
E-mail address: shpark13135@dgist.ac.kr (S.H. Park).

https://doi.org/10.1016/j.patcog.2023.110230
Received 27 March 2023; Received in revised form 26 November 2023; Accepted 26 December 2023
Available online 28 December 2023

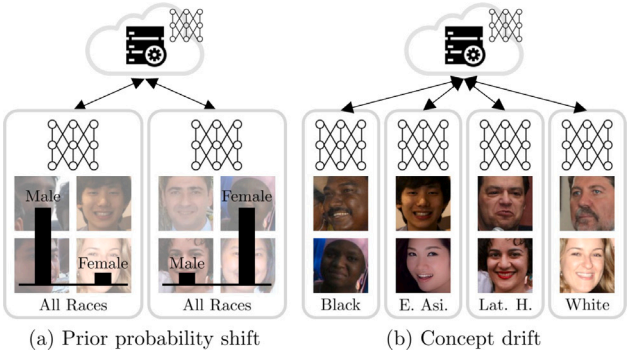


Fig. 1. (a) Federated learning with prior probability shift clients where each client has a different label distribution. (b) Federated learning with concept drift clients where each client has different style images.

with the same data with different labels between clients *e.g.,* the same feature vectors in a training data can have different labels due to personal preferences reflecting sentiment or next word predictors that have personal and regional variation. Despite these distinctions, concept drift has received less attention in FL research due to the focus on evaluating performance under prior probability shift. However, concept drift inevitably occurs in industrial and medical domains [11], where each client collects data under extremely different conditions [12]. As shown in Fig. 1(b), if we train a gender classification model with clients around the world (*e.g.,* Black, East Asian, Indian, Latino Hispanic, Middle Eastern, Southeast Asian, and White [13]), the concept drift problem is more apparent than the prior probability shift issue by gender distribution.

Given the above extreme and challenging cases, previous works do not consistently report high accuracy in concept drift FL experiments. Notably, recent methods (*e.g.,* FedDC [14]) have not exhibited superior accuracy compared to the baseline FedAvg [3] on all datasets (discussed in Section 5.1 with results in Table 4), indicating that the effectiveness of the FL methods on prior probability shift cannot be generalized to all non-iid FL problems. Significant input differences between clients emphasize the heterogeneity of the clients' trained parameters and intermediate features, thus failing to regularize each client model towards the global optimum exacerbating the failure of previous FL methods [5]. In prior probability shifts, though clients have imbalanced label distributions, each client uses consistent style images for training. This implies the front layer allows sharing of knowledge across clients and heterogeneity primarily stems from parameters correlated to frequent label samples. Nonetheless, concept drift introduces varied image styles among clients, leading to significant changes in intermediary feature mean and standard deviations. This results in an overall drift of trained parameters across clients, leading to more severe heterogeneity. Thus, a new FL method for concept drift clients is necessary.

Normalization methods have been proposed to achieve better generalization and faster convergence [15] alongside optimization methods and can be applied to the pre-activations [16] and weights [17] depending on the objective. Unlike previous FL methods focusing on optimization, we employ normalization for weights and intermediate features to mitigate the significant heterogeneity in concept-drifted clients. Specifically, we employ normalization to encourage consistent activations and intermediate features by applying normalization to the weights and calculating better mean and standard deviation. In this paper, we propose FedNN that employs Weight Normalization (WN) and Adaptive Group Normalization (AGN) to address concept drift heterogeneity. Specifically, WN trains a model such that weights have zero mean and unit variance properties via reparameterization. This reduces client parameter variation when updating the global model via averaging. Depending on the data, AGN adaptively selects the optimal mean and standard deviation among data statistics or neighboring-region statistics. In contrast with previous FL methods which restrict the capability of each client using optimization-based methods, our method does not impose such limitations. Consequently, our method mitigates overall parameter drift in clients by ensuring consistency in both weights and features using normalization, effectively tackling the issue of concept drift. For evaluation, we construct seven datasets, where each client only has access to different style (or metadata) images during training. Our method obtains the best average accuracy and fastest convergence speed across all datasets.

In summary, the main contributions are as follows:

- As one of the early works, we investigate FL with concept drift on newly constructed public datasets and highlight the limitations of existing FL methods.
- We employ weight normalization to reduce weight drift caused by heterogeneous clients during global model updates.
- We propose adaptive group normalization to adaptively select the optimal mean and standard deviation for feature normalization. This enables robustness to various data shifts with different characteristics.
- FedNN can be used as a module in existing FL methods since weight and feature normalization of weights is used in any optimization-based process. We demonstrate that FedNN dramatically increases the performance of existing FL methods on seven datasets.

2. Related work

2.1. Federated learning

Various FL methods have been proposed to address the challenges of client heterogeneity stemming from non-IID data. Li et al. [7] proposed FedProx that uses a proximal term that minimizes heterogeneity between the client and global *i.e.,* the squared norm of global and client parameters is used as a regularization. This restricts significant parameter changes during client training, reducing client heterogeneity and resulting in a highly accurate global model. Karimireddy et al. [8] proposed Scaffold, which employs a gradient correction during training clients. The approximated optimal gradient is calculated by averaging the multiple client gradients, and it is used to correct the client's gradient to become more globally optimal. Acar et al. [18] proposed FedDyn to make the client optima asymptotically consistent with the global optima. They trained the model with linear and quadratic terms to modify the local loss dynamically. Since Scaffold and FedDyn adjust each clients' loss to align with the global optimum, it reduces the diversity of client parameters and enables improved global model training. Gao et al. [14] proposed FedDC to limit large-scale updates between clients by using a penalized term with a local drift variable. The local drift variable represents the gap between the client and the global models and approximates the global parameter during the client's update. The approximated global parameters not only maintain consistency among client parameters but also reflect the overall status of client updates, contributing to training an accurate global model. These methods evaluate their superiority on the prior probability shift client but are often limited when addressing concept drift FL. We propose a novel method to address the concept drift and prior probability shift problems.

2.2. Normalization

Various methods including weight normalization [17] and centered weight normalization [19] have been proposed to consider normalization in terms of weights. Each method employs a different strategy for normalization, such as dividing or subtracting different statistical values to achieve specific properties. To be specific, Salimans et al. [17] proposed weight normalization that reparameterizes weights by dividing their Euclidean norm of weights. Huang et al. [19] proposed

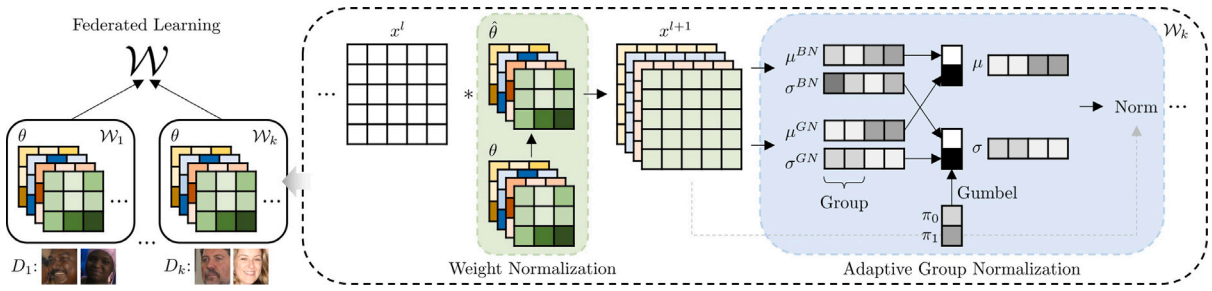


Fig. 2. A schematic illustration of our proposed method for federated learning on concept drift data. $\mathcal{W}$ denotes the global model weight. $\mathcal{W}_k$ and D_k denote the kth client model weight and training data, respectively. To address concept drift heterogeneity, weight normalization (WN) and adaptive group normalization (AGN) are employed. WN reparameterizes the weight θ to have zero mean and unit variance properties *i.e.,* $\hat{\theta}$, a proxy weight used for a convolution operation. AGN uses π_0 and π_1 with Gumbel-Softmax to choose the normalization statistics between BN ($\mu^{BN}\&\sigma^{BN}$) and GN ($\mu^{GN}\&\sigma^{GN}$).

centered weight normalization that reparameterizes the normalized weight to have the zero-mean property. Similarly, Qiao et al. [20] reparameterizes the centered weight by dividing the standard deviation to have unit variance.

Besides, various methods such as batch normalization [16], filter response normalization [21], representative batch normalization [21], group normalization [22], layer normalization [23], instance normalization [24], mode normalization [25], switchable normalization [26], for normalizing features have been proposed. First, Ioffe et al. [16] proposed batch normalization, which uses the batch-wise computed mean and standard deviation to normalize the pre-activation, making the optimization landscape smoother [15]. However, the performance of batch normalization degrades significantly when training with a small batch size [22]. Filter response normalization [21], and representative batch normalization [27] have been proposed to mitigate train-test discrepancy that stems from the bias of non-IID mini-batch statistics. Filter response normalization [21] independently normalizes each feature response, whereas representative batch normalization [27] employs instance-specific statistics to calibrate the centering and scaling operations. Additionally, instead of using batch statistics, the testing discrepancy can be avoided by employing statistics of the neighboring region *e.g.,* group normalization [22], layer normalization [23], and instance normalization [28] where normalization values were calculated on a per-group, per-layer, and per-instance basis, respectively. Moreover, combinations that take advantage of different dimensions statistics have been proposed *e.g.,* mode normalization [25], and switchable normalization [26]. In mode normalization [25], a gating network categorizes samples into distinct modes and normalizes each sample within its respective mode. Meanwhile, switchable normalization [26] employs a set of weighting parameters to switch between three different batch/layer/instance normalization statistics. These methods have been demonstrated in centralized learning scenarios but not explored in the FL setting. We propose appropriate normalizations in FL training to address the heterogeneity between clients.

3. Method

Our objective is to train a global model $\mathcal{W}^*$ that represents all K client datasets D without sharing private kth client data D_k. Formally,

$$\mathcal{W}^* = arg\min_{\mathcal{W}} L(\mathcal{W}) = \sum_{k=1}^{K} p_k L_k(\mathcal{W}), \tag{1}$$

where $L(\mathcal{W})$ and $L_k(\mathcal{W})$ represents the empirical loss on the global and kth client, respectively. We set $p_k = \frac{n_k}{\sum_k n_k}$, where n_k is the number of samples available at kth client. In FedAvg [3], the client model weights, trained on a local dataset, are sent to the server in each communication round, then the server aggregates the kth client models $\mathcal{W}_k$ to obtain a global model $\mathcal{W}$ as follows:

$$\mathcal{W} = \sum_{k=1}^{K} p_k \mathcal{W}_k. \tag{2}$$

Afterward, the new global model parameter is distributed to the clients and employed as an initial parameter for the next stage of local training, as shown in the left side of Fig. 2.

However, if clients are distributed in different environments or owned by diverse users, the data distribution inevitably varies between clients. This causes sizeable intermediate feature differences, increasing client parameter variability. The high variability of client parameters exacerbates the convergence speed since the drifted parameters are used as initial parameters for the next local training. To address this issue, we employ weight normalization (WN) and adaptive group normalization (AGN), as shown in Fig. 2. WN promotes consistent activation after the global model update, while AGN contributes to having a smoother optimization landscape by selecting the optimal Lipschitz constraint [15] that considers the FL scenario. The following sections describe each component in detail.

3.1. Weight normalization (WN)

In convolutional neural networks, a pre-activation $x^l \in \mathbb{R}^{c\times h\times w}$ of the lth layer is fed into the convolution layer and applies a convolution operation as follows:

$$x_i^{l+1} = \theta_i * x^l, \tag{3}$$

where $\theta \in \mathbb{R}^{out\times c\times kh\times kw}$. We abbreviate the batch and the bias term (set to zero) for simplicity. i represents the index of weight in the order of (out, c, kh, kw) where out, c, kh, and kw is the size of the output channel, input channel, kernel height, and kernel width *i.e.,* $i = (i_{out}, i_c, i_{kh}, i_{kw})$. In WN, the weight θ is reparameterized to the proxy weight $\hat{\theta}$ as follows:

$$x_i^{l+1} = \hat{\theta}_i * x^l, \quad \hat{\theta}_i = \frac{\theta_i - \mu_i^{WN}}{\sigma_i^{WN}}, \tag{4}$$

where

$$\mu_i^{WN} = \frac{1}{m}\sum_{j\in S_i}\theta_j, \quad \sigma_i^{WN} = \sqrt{\frac{1}{m}\sum_{j\in S_i}(\theta_j - \mu_i^{WN})^2 + \epsilon}.$$

We define the set S_i as $\{j \mid j_{out} = i_{out}\}$, and this indicates μ_i^{WN} and σ_i^{WN} are computed along the (c, kh, kw) axes. m is the number of elements in S_i, and ϵ denotes a small constant. During training, stochastic gradient descent (SGD) optimizes θ using the proxy weight $\hat{\theta}$ and the loss L.

In FL scenario, the client heterogeneity causes client parameter drifts *e.g.,* shift and scale. This high variability leads to inconsistent activations after the global model update, as shown in Fig. 3(a). These

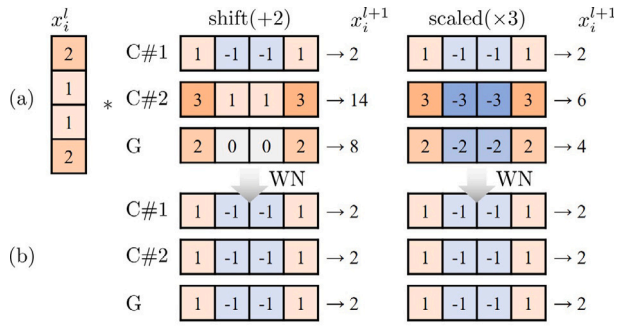


Fig. 3. We illustrate parameter drift when the global model is updated. (a) a convolution example in FedAvg, and (b) a convolution example in FedAvg with WN. C#1, C#2, and G represent the parameter for client #1, client #2, and global, respectively. We set the client's heterogeneity to shift(+ 2) and scaled(×3). WN shows consistent activation after a global update.

inconsistencies can be resolved through WN, as shown in Fig. 3(b). Since weight variations inevitably occur during FL training, consistent activation leads to faster convergence and higher accuracy.

3.2. Adaptive Group Normalization (AGN)

Similarly, the normalization can be applied to features given as

$$\hat{x}_i^l = \frac{x_i^l - \mu_i}{\sigma_i}, \tag{5}$$

where

$$\mu_i = \frac{1}{m}\sum_{j\in S_i} x_j^l, \quad \sigma_i = \sqrt{\frac{1}{m}\sum_{j\in S_i}(x_j^l - \mu_i)^2 + \epsilon}.$$

x_i^l is an input of the lth layer, and i indicates the index of the feature in the order of (b, c, h, w) where b, c, h, and w denote the size of batch, channel, height, and width, respectively, *i.e.,* $i = (i_b, i_c, i_h, i_w)$. In BN [16], the set S_i for μ_i^{BN} and σ_i^{BN} is defined as follows:

$$S_i = \{j \mid j_c = i_c\}. \tag{6}$$

This indicates μ_i^{BN} and σ_i^{BN} are computed along the (n, h, w) axes. In GN [22], S_i for μ_i^{GN} and σ_i^{GN} is defined as follows:

$$S_i = \{j \mid j_b = i_b, \left\lfloor \frac{j_c}{c/g} \right\rfloor = \left\lfloor \frac{i_c}{c/g} \right\rfloor\}, \tag{7}$$

where g denotes the number of groups, and $\lfloor\cdot\rfloor$ is a floor operation. This indicates μ_i^{GN} and σ_i^{GN} are computed along the (h, w) and c/g channel axes. Subsequently, a normalization module has a trainable scale and shift parameters *i.e.,* γ and β to compensate for the representation ability as follows:

$$x_i^{l+1} = \gamma\hat{x}_i^l + \beta. \tag{8}$$

For simplicity, we omit an index (by i_c) in γ.

Although normalization ensures that the activation of the layer has zero mean and unit variance, the characteristics of normalization vary from method to method. BN stands for data statistics than GN since the μ_i^{BN} and σ_i^{BN} are computed batch-wise. On the other hand, GN is less dependent on the data distribution, thus leading to stable training even with small batch sizes [22]. Especially in concept drift FL, the differences between the client's $\mu_i^{BN}\&\sigma_i^{BN}$ and the global-optimal $\mu_i^{BN}\&\sigma_i^{BN}$ become greater, because the different sub-set is only accessible during client training. From this perspective, GN can be considered an appropriate normalization technique for FL. However, from another perspective, data-dependent μ_i^{BN} and σ_i^{BN} can often provide more important statistical information for making a better global model. Thus, we need to choose one of the methods that fits the objective. However, selecting a suitable normalization method is non-trivial and time-consuming. To address this issue, we propose adaptive group normalization (AGN) that selects better normalization statistics μ_i and σ_i among BN and GN. To achieve this, AGN employs the Gumbel-Softmax trick [29] that is fully differentiable and allows to learn a discrete decision sampling as follows:

$$s_r = \frac{\exp((\log(\pi_r) + g_r)/\tau)}{\sum_{q\in\{0,1\}}\exp((\log(\pi_q) + g_q)/\tau)}, \quad r \in \{0, 1\}, \tag{9}$$

where $g_r = -\log(-\log(u))$ with $u \sim Uniform(0, 1)$ and τ denotes the temperature of the softmax. π_0 and π_1 are learnable parameters denotes the selection of BN and GN ($r = 0$ implies using BN and $r = 1$ for GN) and s_0 and s_1 are selected probabilities of BN and GN. Formally,

$$\mu_i = s_0\mu_i^{BN} + s_1\mu_i^{GN}, \quad \sigma_i = s_0\sigma_i^{BN} + s_1\sigma_i^{GN}, \tag{10}$$

is used instead of $\mu_i^{BN}\&\sigma_i^{BN}$ and $\mu_i^{GN}\&\sigma_i^{GN}$ when normalization is applied. Note that when τ approaches zero, the soft decision becomes discrete. We initially set τ to 5 and gradually decreased it to zero during training. Since the Gumbel-Softmax trick is differentiable, gradually decreasing τ to zero allows AGN to make a discrete decision, selecting better normalization statistics between BN and GN by optimizing π_0 and π_1. Besides, AGN eliminates the necessity to run multiple experiments to choose a better normalization that varies among datasets, ultimately contributing to efficient FL. Additionally, the computation overheads of WN and AGN are marginal (less than 0.001 s for each training step) compared to the regular convolution and normalization modules, improving the applicability for practical use.

In prior probability shift, though each client has an imbalanced label distribution, the consistent style images are fed to the model. This implies that the front layer of the model facilitates sharing of knowledge among clients, and client heterogeneity arises primarily from the emphasis on parameters that are highly correlated to labels with high-frequency. However, concept drift supplies varying image styles into the model between clients. Consequently, they exhibit a shift in the mean and standard deviation of intermediate features, leading to overall changes in learned parameters among clients. Previous FL methods attempted to address this using optimization-based approaches, however, these techniques exhibit drawbacks. They might restrict efficient client model training and potentially deteriorate overall training quality if the aggregated global model performs poorly. Therefore, the focus of this study shifts towards incorporating normalization methods to achieve consistent intermediate features for both weights and features. WN calibrates mean and standard deviation drifts in weights from global model aggregation, facilitating consistent activation. Meanwhile, AGN offers the advantage of selecting appropriate normalization statistics based on either the data distribution or neighboring-regions. Ultimately, this selection significantly contributes to preserving consistent features across different clients. By adopting these approaches, we are able to effectively tackle the challenges stemming from significant client heterogeneity compared to the conventional optimization-based methods. In summary, we posit that concept drift introduces different forms of heterogeneity, prompting the proposition of WN and AGN methods. These methods facilitate feature consistency in FL scenarios, ultimately leading to high accuracy.

4. Experiments

4.1. Evaluation scenarios

To show the superiority of FedNN, we compare our method against the state-of-the-art FL methods on concept drift datasets (Section 5.1). Since communication overhead is a major concern in FL, we show convergence plots to highlight efficiency (Section 5.2) and visualize global model features with/without FedNN to understand improvement sources (Section 5.3). To evaluate the effect of our method on non-concept drift datasets, we conduct experiments on a prior probability

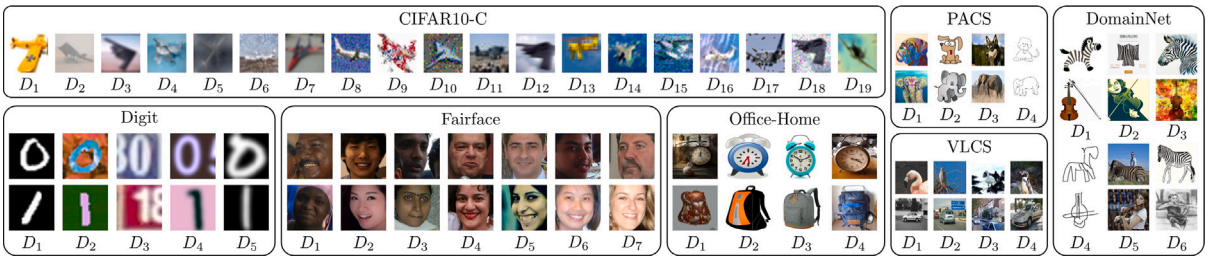


Fig. 4. The concept drift datasets. Each client's image (*i.e.,* D_k) is displayed in a different column. Since each client has different sources of acquisition, the styles of images vary significantly.

shift dataset (Section 5.4). Additionally, we run experiments in personalized FL scenarios to show that FedNN is valid in various FL scenarios (Section 5.5). Moreover, we run experiments on a larger number of clients(= 500) to verify the scalability of FedNN (Section 5.6). By comparing FedNN with state-of-the-art normalization methods, we demonstrate that using FedNN is a valid choice for FL (Section 5.7). Finally, we also report accuracy improvements over centralized learning to show FedNN's benefit (Section 5.8).

4.2. Datasets

We construct seven different datasets (Fig. 4) from public data to evaluate our method on concept drift FL. Concept drift datasets have significantly different style images between clients, but share the same labels.

CIFAR10-C: CIFAR10-C [30] is a ten-object category dataset comprising 19 corrupted and perturbed images *e.g.,* brightness, contrast, defocus blur, etc. We select level-4 strength images for our experiments. We construct a dataset such that each client has different corrupted and perturbed images.

Digit: Digit consists of five different style images collected from MNIST [31], MNIST-M [32], SVHN [33], SynthDigits [32], and USPS [34]. We construct a dataset consisting of images of different styles per client.

Fairface: Fairface [13] is an evenly distributed facial dataset with multiple metadata *i.e.,* age, gender, and race. We construct a gender classification dataset consisting of seven clients where each client has different race metadata images (Black, East Asian, Indian, Latino Hispanic, Middle Eastern, Southeast Asian, and White).

Office-Home: Office-Home [35] has four different style images *i.e.,* Art, Clipart, Product, and Real-World. This dataset has 65 categories of objects *e.g.,* Alarm clock, Backpack, Batteries, etc. We construct a dataset such that each client has different styles of images.

PACS: PACS [36] consists of four different style images *i.e.,* Photo, Art painting, Cartoon, and Sketch. This dataset has seven common categories *i.e.,* dog, elephant, giraffe, guitar, horse, house, and person. We construct a dataset such that each client has different styles of images.

VLCS: VLCS [37] is a dataset that consists of images from PASCAL VOC2007, LabelMe, Caltech-101, and SUN09. VLCS has five object categories *i.e.,* bird, car, chair, dog, and person. We construct a dataset in which each client is collected from a different source.

DomainNet: DomainNet [38] consists of six different style images *i.e.,* clipart, real, sketch, infograph, painting, and quickdraw. This dataset has 345 categories. We construct a dataset such that each client has different styles of images. We employ DomainNet to validate the accuracy of our method on larger amounts of data.

For CIFAR10-C, we reserve 500 samples of each client's images as the test split. For Office-Home, PACS, and VLCS, we reserve 10% of each client's images as a test split. We use an image size of $(3 \times 32 \times 32)$ for CIFAR10-C and Digit, and $(3 \times 224 \times 224)$ for Fairface, Office-Home, PACS, VLCS, and DomainNet, respectively.

Additionally, we use **CIFAR10** [39] for prior probability shift experiments with different Dirichlet [9] distribution ratios ($\beta = 0.3$ and $\beta = 0.5$) following FedDyn [18] and FedDC [14].

Table 1
Classification average accuracy of FL methods (FedAvg [3], FedProx [7], Scaffold [8], FedDyn [18], and FedDC [14]) with and without normalization on CIFAR10-C and Digit datasets. **Black bold** indicates the best accuracy within without normalization (w/o), BN, and GN, respectively.

Method	CIFAR10-C			Digit		
	w/o	BN	GN	w/o	BN	GN
FedAvg	58.99	**61.95**	61.65	77.77	81.02	**81.8**
FedProx	58.99	**61.95**	61.65	77.77	81.02	**81.8**
Scaffold	58.42	61.23	**61.43**	77.27	81.02	**82.02**
FedDyn	60.06	**60.67**	58.27	79.09	79.35	**79.86**
FedDC	59.13	60.01	**62.23**	73.92	**79.13**	78.9

4.3. Baseline

We compare our method against five FL methods for non-IID clients *i.e.,* FedAvg [3], FedProx [7], Scaffold [8], FedDyn [18], and FedDC [14]. FedAvg, FedProx, and Scaffold are typical FL methods for non-IID clients, thus, a comparison is essential. FedDyn and FedDC are state-of-the-art FL methods for non-IID clients; these methods are used to demonstrate the superiority of the proposed method. Additionally, we employ FedBN [40] and LG-FedAvg [41] for personalized FL comparison.

4.4. Models

We used LeNet [31] for CIFAR10-C and Digit experiments, and ResNet18 [42] for Fairface, Office-Home, PACS, VLCS, and DomainNet experiments. For LeNet with feature normalization, we place BN and GN after each convolution layer. For LeNet with FedNN (*i.e.,* WN and AGN), we apply WN to all convolutional layers with subsequent AGN. For ResNet18 [42] with GN, we replace BN with GN and set the number of groups in GN to 2 following [18]. For ResNet18 with FedNN, we apply WN to all convolutional layers with subsequent AGN.

4.5. Training details

We run experiments for 200 communication rounds and set the device participation rate to 40%. In the local training phase, we set the batch size to 50 and trained 5 epochs for each local training using SGD. We set the initial learning rate to 0.1 and set the learning rate decay to 0.998. For the specific hyper-parameters of each method, we follow the same settings as the prior work (FedAvg [3], FedProx [7], Scaffold [8], FedDyn [18], and FedDC [14]). For prior probability shift experiments, we set the number of clients to 100, and device participation rate to 15% following FedDyn [18] and FedDC [14]. We report all results obtained by the global model.

During client training, data augmentations *i.e.,* random-cropping and horizontal flips with a probability of 0.5 were used on CIFAR10-C, Office-Home, PACS, VLCS, and DomainNet following CIFAR10 training settings in FedDyn [18] and FedDC [14]. For Fairface, we apply random-affine translations (rotation, translation, and scale) and horizontal flips with a probability of 0.5. In the case of Digit, no

Table 2
Classification accuracy ours (*i.e.,* FedNN) and ablation studies on six datasets. **Red** is the first and blue the second highest accuracy. We exclude experiments without feature normalization (*e.g.,* BN) for large-scale datasets, as ResNet18 requires feature normalization.

Method	CIFAR10-C	Digit	Fairface	Office-Home	PACS	VLCS	Avg.
FedAvg	58.99	77.77	–	–	–	–	68.38
FedAvg+Ours(w/o AGN)	53.4	70.76	–	–	–	–	62.08
FedAvg(BN)	61.95	81.02	84.55	68.91	87.22	72.84	76.08
FedAvg(GN)	61.65	81.8	83.52	64.27	87.42	68.7	74.56
FedAvg(BN)+Ours(w/o AGN)	63.66	82.5	**86.01**	71.1	88.54	74.06	77.64
FedAvg(GN)+Ours(w/o AGN)	**67.75**	**83.51**	84.51	**72.23**	**90.47**	71.62	78.34
FedAvg+Ours	67.27	83.44	85.9	71.4	90.06	**74.25**	**78.72**

Table 3
Classification accuracy of FL methods with BN+Ours(w/o AGN) and GN+Ours(w/o AGN) on six datasets. **Black bold** indicates the higher accuracy between BN and GN.

Method	CIFAR10-C	Digit	Fairface	Office-Home	PACS	VLCS	Avg.
FedAvg(BN)+Ours(w/o AGN)	63.66	82.5	**86.01**	71.12	88.54	**74.06**	77.64
FedProx(BN)+Ours(w/o AGN)	63.66	82.5	**86.01**	**71.88**	88.54	**74.06**	77.64
Scaffold(BN)+Ours(w/o AGN)	64.22	**82.23**	**85.76**	**72.23**	88.74	**74.62**	77.96
FedDyn(BN)+Ours(w/o AGN)	62.51	81.03	**84.57**	**58.73**	**77.89**	**66.17**	**71.81**
FedDC(BN)+Ours(w/o AGN)	63.51	80.77	**85.76**	**69.18**	87.32	**74.06**	76.76
FedAvg(GN)+Ours(w/o AGN)	**67.75**	**83.51**	84.51	**72.23**	**90.47**	71.62	**78.34**
FedProx(GN)+Ours(w/o AGN)	**67.75**	**83.51**	84.51	71.12	**90.47**	71.62	**78.34**
Scaffold(GN)+Ours(w/o AGN)	**68.26**	83.21	84.14	70.43	**90.87**	72.74	**78.27**
FedDyn(GN)+Ours(w/o AGN)	**64.24**	**82.24**	68.48	51.18	73.33	54.51	65.66
FedDC(GN)+Ours(w/o AGN)	**66.89**	**81.2**	84.15	68.84	**88.84**	72.27	**77.03**

Table 4
Classification accuracy of ours (*i.e.,* FedNN) and previous FL methods. **Black bold** indicates the best accuracy within each sub-row. ↑ indicates improved accuracy compared to the accuracy of without ours.

Method	CIFAR10-C	Digit	Fairface	Office-Home	PACS	VLCS	Avg.
FedProx(GN)	61.65	81.8	83.39	**64.27**	**86.31**	68.7	**74.35**
Scaffold(GN)	61.43	**82.02**	**83.43**	60.6	83.27	68.42	73.19
FedDyn(GN)	58.27	79.86	58.45	10.25	27.69	50.85	47.56
FedDC(GN)	**62.23**	78.9	80.49	62.6	83.06	**68.8**	72.68
FedProx+Ours	**67.27**	**83.44**	**85.9**	71.4	**90.06**	**74.25**	**78.72** (**4.37**↑)
Scaffold+Ours	66.45	83.27	85.74	**71.95**	**90.06**	74.15	78.6 (5.41↑)
FedDyn+Ours	64.77	81.36	83.48	52.56	76.77	67.29	71.03 (23.47↑)
FedDC+Ours	64.65	81.95	85.42	67.31	86.21	71.9	76.24 (3.56↑)

augmentations were used following FedDyn [18] and FedDC [14]. For DomainNet, we run experiments for 100 communication rounds. In all experiments, we fixed the random seed for reproducibility similar to prior works FedDyn [18] and FedDC [14].

4.6. Implementation details

All methods are implemented with Pytorch [43]. For comparison methods, we prioritize using the author's implementation. In particular, methods FedDyn[1] [18] and FedDC[2] [14] follow author implementations. For FedAvg [3], FedProx [7], and Scaffold [8], we follow FedDC's implementation. As for FedBN [40], we integrate the author's implementation[3] in our code, and for LG-FedAvg [41], we follow the author's implementation.[4] For prior probability shift experiments, we only integrate our method as a module in FedDC without modifying default experimental settings.

5. Results

We demonstrate the superiority of FedNN in model accuracy and convergence speed, including t-Stochastic Neighbor Embedding (t-SNE) visualization.

[1] https://github.com/alpemreacar/FedDyn
[2] https://github.com/gaoliang13/FedDC
[3] https://github.com/med-air/FedBN
[4] https://github.com/pliang279/LG-FedAvg

5.1. Comparison against previous FL methods

Table 2 shows the accuracy of FedNN (*i.e.,* WN with AGN) and its ablations. FedNN reports higher accuracy among GN and BN, and achieves the best average accuracy across all datasets. This implies that both normalizations are greatly beneficial for concept drift FL. AGN contributes to finding better statistics of normalization using Gumbel-Softmax during training. Whereas WN reduces client heterogeneity by alleviating conflicts during global model updates.

Next, we analyze the impact of each module *i.e.,* WN and AGN. Comparisons with/without feature normalization (*i.e.,* BN or GN) imply that both normalizations are still beneficial for concept drift FL. Although, recent FL methods argue BN can be problematic in FL [18], this finding is valid for other FL algorithms as the models with normalization outperform those without normalization, as shown in Table 1. Additionally, WN considerably improves accuracy when applied to feature normalization. These improvements demonstrate that WN is advantageous on concept drift FL. Note that the accuracy of the feature normalization varies from dataset to dataset. In CIFAR10-C, Digit, Office-Home, and PACS, GN shows the best accuracy, while BN takes second place. On the other hand, BN obtains the best accuracy on Fairface and VLCS, and BN (without WN) places second. Similar observations can be found in other FL algorithms, as shown in Table 3. Although one of the normalizations can improve the FL performance significantly, finding the optimal normalization is time-consuming and non-trivial. Overall, an adaptive way of finding the best statistic for normalization contributes to performing FL more effectively.

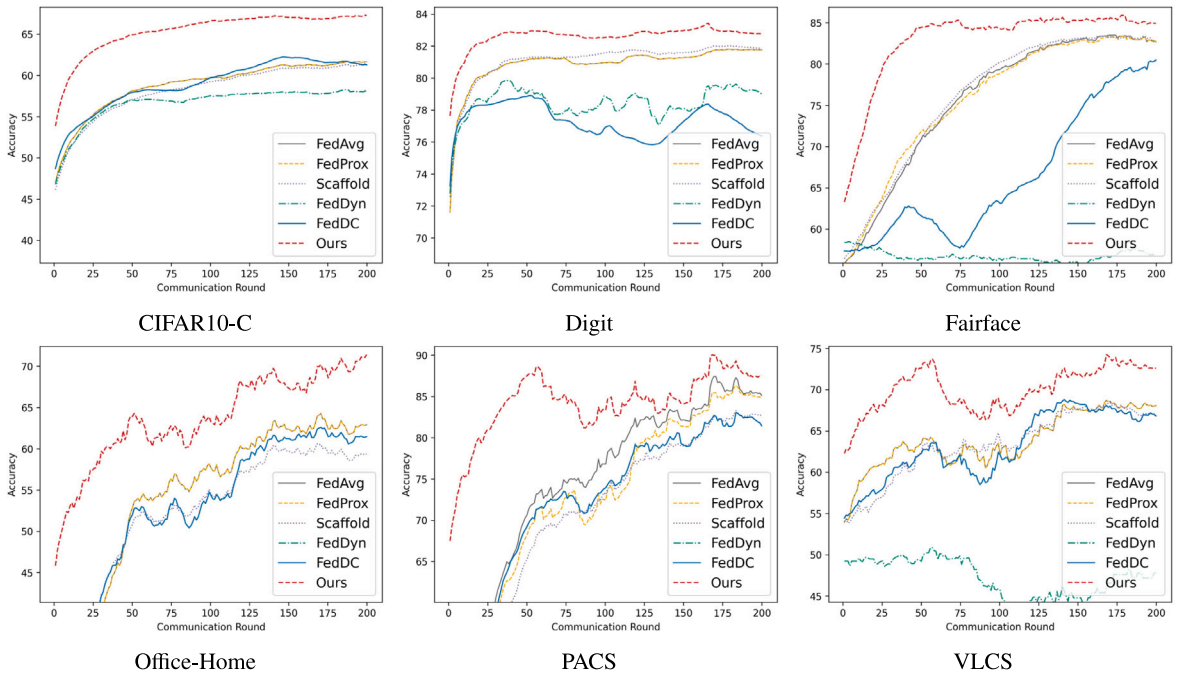


Fig. 5. Convergence plots of ours (*i.e.,* FedNN with FedAvg) and previous FL methods (with GN) on six concept drift datasets. Our method shows the fastest convergence speed.

Table 5
Classification accuracy of previous FL methods with WN with/without feature normalization on CIFAR10-C and Digit datasets.

Method	CIFAR10-C		Digit	
	w/o	GN	w/o	GN
FedAvg	53.4	61.65	70.76	81.8
FedProx	53.4	61.65	70.76	81.8
Scaffold	48.46	61.43	69.68	82.02
FedDyn	46.48	58.27	15.02	79.86
FedDC	19.24	62.23	12.44	78.9

In Table 4, we show the accuracy of previous FL methods. None of the methods obtain the best accuracy on all six datasets. We empirically demonstrate that the previous state-of-the-art FL methods cannot resolve concept drift FL adequately. For all cases, the baseline FL algorithm FedAvg with ours in Table 2 outperforms other methods. Prior FL methods aim to regularize the client model towards the global optimum that leads to inconsistent activations after global updates and results in high accuracy. However, due to the challenge of predicting the global optimum in concept drift, previous optimization-based FL approaches fail to address heterogeneity in concept drift data. In contrast, FedNN is a simple and effective solution for concept drift datasets. WN minimizes parameter drift during global model updates leading to consistent activations. At the same time, feature normalization (*e.g.,* BN, GN) improves stable client training and enables a robust global model, ultimately resulting in higher accuracy. Additionally, the proposed method is independent of the optimization-based FL methods. Hence, it can be integrated as a module into any FL method. FedNN (+Ours in Table 4) significantly improves accuracy compared to the algorithms without FedNN. This shows that our proposal is valid and beneficial regardless of FL algorithms.

Besides, we report the accuracy of prior FL methods using WN without feature normalization in Table 5. WN without feature normalization shows significantly lower accuracy compared to methods with GN, suggesting that feature normalization is essential for WN. Additionally, the proximal term effects are marginal when the concept drift dataset is used for FL training (we set hyper-parameter μ to 0.001 following FedDyn [18] and FedDC [14]). Thus, FedAvg and FedProx show similar accuracy in a deterministic training setup.

In Table 6, we show the accuracy of FL methods (FedAvg [3], FedProx [7], Scaffold [8], FedDyn [18], and FedDC [14] with GN) and FedNN on DomainNet dataset. Overall, our method outperforms all competitors similar to previous experiments (*e.g.,* Table 2). Additionally, this demonstrates the effectiveness of our method on large-scale data and also suggests that success in small-scale data (*e.g.,* PACS) represents its applicability in large-scale data (*e.g.,* DomainNet). Besides, similar to Office-Home and PACS in Table 4, substantial image style differences among clients have a detrimental impact on FedDyn accuracy. In FedDyn, the evaluation (with ResNet18) was limited to the IID FL setting and they noted that employing networks with feature normalization (*e.g.,* BN) is problematic to use with FedDyn. Despite the challenges posed by severe heterogeneity, our method consistently shows superior accuracy compared to the previous FL methods.

In Table 7, we show the accuracy of FedNN with different initial τ values of AGN on CIFAR10-C and Digit datasets. Though our setting ($\tau = 5$) shows the best-averaged accuracy, there are no significant differences among different initial τ values. This suggests that adjusting the initial value of τ has a marginal effect on the final accuracy, indicating that AGN is robust to hyper-parameter changes.

5.2. Fast convergence of FedNN

Reducing communication overhead is one of the major concerns in distributed training. Fast model convergence allows a global model to be trained within a few communication rounds, improving FL efficiency. Fig. 5 shows convergence plots of FedNN and the previous FL methods. Our method shows the fastest convergence speed across all datasets. This highlights that FedNN is a valid choice for FL training since it significantly reduces communication overheads. Note that parameter drift occurs due to the high variability in client parameters, thus significantly reducing convergence speed. This indicates that optimization-based FL methods fail to regularize client gradients towards the global optimum resulting in slow convergence. Rather than regularizing each client's training to reduce inconsistent activations and global updates, FedNN offers a simple yet effective approach to address FL heterogeneity.

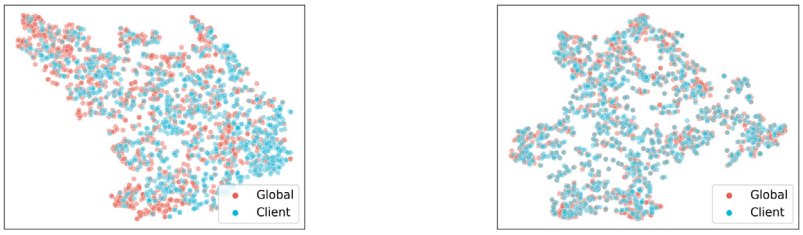


Fig. 6. t-SNE visualization of global and client(#1) model's features at 10 communication rounds for (a) FedAvg and (b) FedAvg+Ours on the same Digit test dataset. Red and cyan dots indicate the global and client features, respectively. Gray indicates the overlapping of red and cyan dots. Our method shows a more similar distribution between the global and client features. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

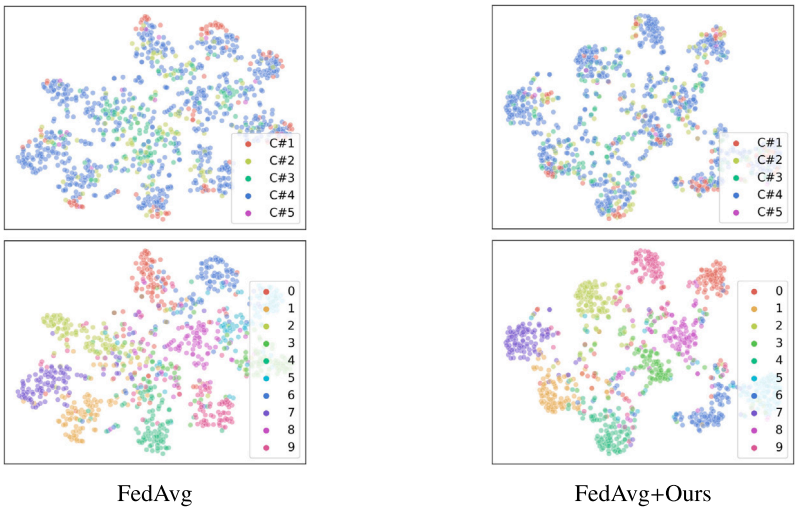


Fig. 7. t-SNE visualization of the features extracted by the global model for (a) FedAvg and (b) FedAvg+Ours on the Digit test dataset. In the top row, each color represents each client, and the clustered distribution by clients indicates robust features. In the bottom row, each color represents a classification label (zero to nine), and the clustered distribution by labels indicates discriminative features. Our method extracts more robust and discriminative features. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 6
Classification accuracy of ours (*i.e.,* FedNN) and previous FL methods on DomainNet dataset.

	FedAvg	FedProx	Scaffold	FedDyn	FedDC	FedAvg+Ours
Avg.	29.62	30.15	31.84	1.08	31.55	**35.67**

Table 7
Classification accuracy of ours (*i.e.,* FedNN) with different initial τ values of AGN on CIFAR10-C and Digit datasets.

	$\tau = 3$	$\tau = 4$	$\tau = 5$	$\tau = 6$	$\tau = 7$
CIFAR10-C	67.68	67.44	67.27	66.53	67.09
Digit	82.83	83.14	83.44	83.36	83.41
Avg.	75.25	75.29	75.35	74.94	75.25

Table 8
Classification accuracy with and without ours (*i.e.,* FedNN) on a prior probability shift dataset (*i.e.,* CIFAR10). D-0.3 and D-0.6 denote a dataset of 0.3 and 0.6 Dirichlet distributions. **Black bold** indicates the best accuracy. ↑ indicates improved accuracy compared to the accuracy without FedNN.

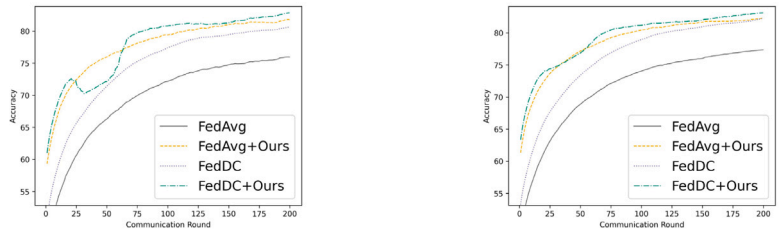
Method	CIFAR10 D-0.3		CIFAR10 D-0.6	
	w/o	+Ours	w/o	+Ours
FedAvg	75.97	81.8(5.83↑)	77.34	82.31(4.97↑)
FedProx	76.36	81.55(5.19↑)	77.52	82.26(4.74↑)
Scaffold	78.97	81.23(2.26↑)	80.36	82.14(1.78↑)
FedDyn	79.61	71.65(7.96↓)	80.7	74.48(6.22↓)
FedDC	**80.6**	**82.83(2.23↑)**	**82.29**	**83.11(0.82↑)**

5.3. t-SNE visualization analysis

To understand the superiority of our method, we visualize the features of the global and client models in Fig. 6. Compared to FedAvg, where feature distribution drift is observed, FedNN shows consistent feature distribution after the global model update. Consistent activations accelerate global model convergence, as the next client initialization utilizes a less shifted global model. This represents that more stable FL training can be achieved through FedNN. In Fig. 7, we visualize the features of the global model. FedNN shows more discrete and client-invariant feature distributions compared to FedAvg. Client models close to the global optimum are robust and reduce variability, resulting in improved performance. This shows the viability and benefit of our method on concept drift FL.

5.4. Experiments on prior probability shift

We further conduct experiments on prior probability shift FL dataset to confirm that FedNN can still be useful on other non-IID data. We run experiments on CIFAR10 dataset with 0.3 and 0.6 Dirichlet distributions. The results are shown in Table 8. The state-of-the-art method *i.e.,*



CIFAR10 D-0.3 CIFAR10 D-0.6

Fig. 8. Convergence plots of FedAvg and FedDC with ours (*i.e.,* FedNN) on prior probability shift dataset. Our method contributes to faster convergence.

Table 9
Classification average accuracy of FedBN [40], LG-FedAvg [41], and ours (*i.e.,* FedNN) on six datasets. Average accuracy is calculated by averaging the accuracy of each personalized test set. **Black bold** indicates the best accuracy.

Method	CIFAR10-C	Digit	Fairface	Office-Home	PACS	VLCS
FedBN	62.26	85.55	80.39	57.06	87.21	74.75
LG-FedAvg	45.25	81.88	73.31	51.38	80.06	73.3
Ours	**64.71**	**86.41**	**82.89**	**64.13**	**87.59**	**78.67**

FedDC obtains the best performance against all FL methods. However, as shown in Table 8 +Ours, FedNN significantly improves the accuracy of most FL methods (except FedDyn) when integrated into a module. Note that the FedAvg+Ours outperforms FedDC(w/o) in 0.3 and 0.6 Dirichlet distribution experiments. This indicates that simple normalizations can beat a complex state-of-the-art FL algorithm. Interestingly, the improvement is greater when the non-IID becomes severe (0.3 is more challenging). Our method reduces the performance of FedDyn. In FedDyn [18], they state that BN is problematic to use with FedDyn, and we obtained similar results in prior probability shift data. In Fig. 8, we show convergence plots for two FL methods with and without FedNN. Although FedDC achieves faster convergence than FedAvg, our method further improves the convergence speed of both FedAvg and FedDC methods. Consequently, the effectiveness of our method has been demonstrated in prior probability shift non-IID problems.

5.5. Comparison against personalized FL

As personalized FL is gaining attention in addition to general FL, we also validate our method in a personalized FL setting [44] against FedBN [40] and LG-FedAvg [41] in Table 9. In contrast to general FL, which evaluates the global model using a global test set, personalized FL evaluates each client model using its respective client test set. Personalized FL achieves higher accuracy by keeping its unique layers in the client. FedBN enhances the representation of each client by skipping the synchronization of the BN layers during global model updates. On the other hand, LG-FedAvg divides the network into a local model consisting of forward layers, and a global model consisting of backward layers. During global model aggregation, LG-FedAvg transmits a global model to the central server while retaining a local model within each client. This enables each client to possess a unique representation. Our method outperforms FedBN and LG-FedAvg on all datasets, showing it is also valid for personalized FL. Note that our method differs from FedBN and LG-FedAvg since these methods do not consider global model training and have unique normalization statistics for each client. Also, the objective of our AGN is not to improve the performance of each client, but rather to choose normalization statistics from BN or GN that have a less adverse effect. Additionally, whereas FedBN and LG-FedAvg aim to perform well on each individual client in personalized FL scenarios, our approach aims to obtain a robust global model in FL scenarios.

Table 10
Classification accuracy with and without ours (*i.e.,* FedNN) on a prior probability shift dataset (*i.e.,* CIFAR10). D-0.3 and D-0.6 denote a dataset of 0.3 and 0.6 Dirichlet distributions. Clients indicate the number of clients participating in FL. ↑ indicates improved accuracy compared to the accuracy without FedNN.

Clients	Method	CIFAR10 D-0.3		CIFAR10 D-0.6	
		w/o	+Ours	w/o	+Ours
100	FedAvg	75.97	81.8(5.83↑)	77.34	82.31(4.97↑)
	FedDC	80.6	82.83(2.23↑)	82.29	83.11(0.82↑)
500	FedAvg	61.54	74(12.46↑)	66.94	77.11(10.17↑)
	FedDC	62.59	75.69(13.1↑)	69.11	76.18(7.07↑)

Table 11
Classification accuracy of recent normalization methods with FedAvg on six datasets. **Black bold** indicates the best accuracy.

Dataset	FRN [21]	RBN [27]	SN [26]	Ours
CIFAR10-C	62.49	62.58	62.84	**67.27**
Digit	80.79	80.87	82.47	**83.44**
Fairface	83.42	85.72	85.61	**85.9**
Office-Home	62.67	68.28	67.94	**71.4**
PACS	87.42	89.15	87.93	**90.06**
VLCS	67.2	72.27	68.98	**74.25**
Avg.	73.99	76.47	75.96	**78.72**

5.6. Comparison against a larger number of clients

To evaluate the scalability of the proposed method, we increase the number of clients to 500. We conduct experiments on the CIFAR10 dataset with 0.3 and 0.6 Dirichlet distributions. The results are shown in Table 10. Though increasing the number of clients slightly degrades the overall accuracy, our method exhibited lower accuracy drops compared to those without FedNN. This indicates that our method is more robust for a larger number of clients, thereby mitigating potential scalability concerns.

5.7. Comparison against other feature normalization

We compare our method against recent feature normalization methods (*e.g.,* filter response normalization (FRN) [21] and representative batch normalization (RBN) [27]) that address training-test discrepancies caused by small-batch training. Additionally, we further compare our method against switchable normalization (SN) [26] that adaptively combines multiple normalizations using softmax (not discrete) to be

Table 12
Classification accuracy with/without ours in CL and FL(FedAvg) on CIFAR10-C, Digit, Fairface, Office-Home, PACS, and VLCS datasets. GN is used as a feature normalization of ResNet18 for w/o. ↑ indicates improved accuracy compared to the accuracy without ours.

	CIFAR10-C		Digit		Fairface		Office-Home		PACS		VLCS	
	w/o	+Ours	w/o	+Ours	w/o	+Ours	w/o	+Ours	w/o	+Ours	w/o	+Ours
CL	62.43	68.71 (6.28↑)	78.42	82.15 (3.73↑)	85.59	86.48 (0.89↑)	64.2	71.81 (7.6↑)	90.26	89.86 (0.4↓)	72.18	72.18 (0↑)
FL	58.99	67.27 (8.28↑)	77.52	83.44 (5.67↑)	83.52	85.9 (2.38↑)	64.27	71.4 (7.13↑)	87.42	90.06 (2.64↑)	68.7	74.25 (5.55↑)

robust for a wide range of batch sizes and tasks. The results are shown in Table 11. Although the recent normalization methods outperform BN (or GN) in centralized scenarios, these methods do not achieve significant accuracy in FL scenarios (compared to BN in Table 2). On the other hand, our method outperforms recent normalization methods. To the best of our knowledge, FedNN is the first FL approach to use normalization on weights. FedNN is valid for concept drift FL resulting in improved accuracy and stability.

5.8. Fednn on centralized learning

In Table 12, we report accuracy (with/without ours) on six datasets for the Centralized Learning (CL) scenario to show the effectiveness of our proposed method. First, we observed that FedAvg had significant and comparable improvements over CL when using WN and AGN across all datasets. WN alleviates conflicts that occur during global model updates, and AGN contributes to finding better normalization statistics during training. Both components greatly reduce client heterogeneity resulting in stable training and high accuracy. Note that compared to CL experiments on prior probability shift data where the non-IID is completely resolved, CL experiments on concept drift data still remain non-IID since a subset of the training dataset is only accessible due to mini-batch training. Thus, our method improves accuracy on some datasets (*e.g.,* CIFAR10-C, Digit, Fairface, and Office-Home) in CL scenarios.

6. Conclusion

Existing FL methods are challenged when the client's data has concept drift, resulting in unstable and slow convergence. We employ weight and adaptive group normalization to mitigate adverse effects during global model updates. To validate our method, we conduct experiments on seven concept drift datasets. Whereas previous state-of-the-art FL methods do not obtain consistent results, normalizing weights and features considerably improves accuracy and convergence speeds. Our success can be attributed to employing tailored normalization in terms of both features and weights that contribute to consistent intermediate activation at all levels, even after global model aggregation. Additionally, we analyze the features of global and client models and confirm our method can learn more consistent and robust features during FL training, with equally good results on prior probability shift datasets and personalized FL scenarios. However, it is worth noting that our method has a limitation; while AGN contributes to preventing accuracy degradation, it does not ultimately improve model accuracy. Further improving accuracy for concept drift scenarios is left as a topic of future research. As one of the early works addressing concept drift FL, this study may inspire the FL research community to test their methods on a wider range of heterogeneity and enhance the practicality of future FL methods. We hope that these findings spur solving real-world non-IID FL problems.

CRediT authorship contribution statement

Myeongkyun Kang: Conceptualization, Formal analysis, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Soopil Kim:** Investigation, Validation, Writing – review & editing. **Kyong Hwan Jin:** Funding acquisition, Project administration, Resources, Supervision, Validation, Writing – review & editing. **Ehsan Adeli:** Project administration, Resources, Supervision, Validation, Writing – review & editing. **Kilian M. Pohl:** Funding acquisition, Project administration, Resources, Supervision, Validation, Writing – review & editing. **Sang Hyun Park:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Writing – review & editing, Writing – original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We use publicly available datasets.

Acknowledgments

This work was supported by funding from the DGIST R&D program of the Ministry of Science and ICT of KOREA (22-KUJoint-02) and the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT)(No. 2019R1C1C1008727) and the Digital Innovation Hub project supervised by the Daegu Digital Innovation Promotion Agency (DIP) grant funded by the Korea government (MSIT and Daegu Metropolitan City) in 2023 (DBSD1-01).

References

[1] Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: Concept and applications, ACM Trans. Intell. Syst. Technol. 10 (2) (2019) 1–19.
[2] M. Ribero, J. Henderson, S. Williamson, H. Vikalo, Federating recommendations using differentially private prototypes, Pattern Recognit. 129 (2022) 108746.
[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, PMLR, 2017, pp. 1273–1282.
[4] S. Huang, W. Shi, Z. Xu, I.W. Tsang, J. Lv, Efficient federated multi-view learning, Pattern Recognit. 131 (2022) 108817.
[5] X. Li, K. Huang, W. Yang, S. Wang, Z. Zhang, On the convergence of fedavg on non-IID data, in: International Conference on Learning Representations, 2019.
[6] T. Li, A.K. Sahu, A. Talwalkar, V. Smith, Federated learning: Challenges, methods, and future directions, IEEE Signal Process. Mag. 37 (3) (2020) 50–60.
[7] T. Li, A.K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks, in: Proceedings of Machine Learning and Systems, Vol. 2, 2020, pp. 429–450.
[8] S.P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, A.T. Suresh, Scaffold: Stochastic controlled averaging for federated learning, in: International Conference on Machine Learning, PMLR, 2020, pp. 5132–5143.
[9] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, Y. Khazaeni, Bayesian nonparametric federated learning of neural networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 7252–7261.
[10] P. Kairouz, H.B. McMahan, B. Avent, A. Bellet, M. Bennis, A.N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al., Advances and open problems in federated learning, Found. Trends® Mach. Learn. 14 (1–2) (2021) 1–210.
[11] M.J. Sheller, B. Edwards, G.A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R.R. Colen, et al., Federated learning in medicine: Facilitating multi-institutional collaborations without sharing patient data, Sci. Rep. 10 (1) (2020) 1–12.
[12] M. Kang, D. Won, M. Luna, P. Chikontwe, K.S. Hong, J.H. Ahn, S.H. Park, Content preserving image translation with texture co-occurrence and spatial self-similarity for texture debiasing and domain adaptation, Neural Netw. 166 (2023) 722–737.

[13] K. Karkkainen, J. Joo, Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 1548–1558.
[14] L. Gao, H. Fu, L. Li, Y. Chen, M. Xu, C.-Z. Xu, FedDC: Federated learning with non-IID data via local drift decoupling and correction, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 10112–10121.
[15] S. Santurkar, D. Tsipras, A. Ilyas, A. Madry, How does batch normalization help optimization? in: Advances in Neural Information Processing Systems, vol. 31, 2018.
[16] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning, PMLR, 2015, pp. 448–456.
[17] T. Salimans, D.P. Kingma, Weight normalization: A simple reparameterization to accelerate training of deep neural networks, in: Advances in Neural Information Processing Systems, vol. 29, 2016.
[18] D.A.E. Acar, Y. Zhao, R.M. Navarro, M. Mattina, P.N. Whatmough, V. Saligrama, Federated learning based on dynamic regularization, in: International Conference on Learning Representations, 2021.
[19] L. Huang, X. Liu, Y. Liu, B. Lang, D. Tao, Centered weight normalization in accelerating training of deep neural networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2803–2811.
[20] S. Qiao, H. Wang, C. Liu, W. Shen, A. Yuille, Micro-batch training with batch-channel normalization and weight standardization, 2019, arXiv preprint arXiv:1903.10520.
[21] S. Singh, S. Krishnan, Filter response normalization layer: Eliminating batch dependence in the training of deep neural networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11237–11246.
[22] Y. Wu, K. He, Group normalization, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 3–19.
[23] J.L. Ba, J.R. Kiros, G.E. Hinton, Layer normalization, 2016, arXiv preprint arXiv:1607.06450.
[24] S. Singh, A. Shrivastava, Evalnorm: Estimating batch normalization statistics for evaluation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 3633–3641.
[25] L. Deecke, I. Murray, H. Bilen, Mode normalization, in: International Conference on Learning Representations, 2019.
[26] P. Luo, J. Ren, Z. Peng, R. Zhang, J. Li, Differentiable learning-to-normalize via switchable normalization, in: International Conference on Learning Representations, 2019.
[27] S.-H. Gao, Q. Han, D. Li, M.-M. Cheng, P. Peng, Representative batch normalization with feature calibration, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 8669–8679.
[28] D. Ulyanov, A. Vedaldi, V. Lempitsky, Instance normalization: The missing ingredient for fast stylization, 2016, arXiv preprint arXiv:1607.08022.
[29] E. Jang, S. Gu, B. Poole, Categorical reparameterization with gumbel-softmax, in: International Conference on Learning Representations, 2016.
[30] D. Hendrycks, T. Dietterich, Benchmarking neural network robustness to common corruptions and perturbations, in: Proceedings of the International Conference on Learning Representations, 2019.
[31] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[32] Y. Ganin, V. Lempitsky, Unsupervised domain adaptation by backpropagation, in: International Conference on Machine Learning, PMLR, 2015, pp. 1180–1189.
[33] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A.Y. Ng, Reading digits in natural images with unsupervised feature learning, 2011.
[34] J.J. Hull, A database for handwritten text recognition research, IEEE Trans. Pattern Anal. Mach. Intell. 16 (5) (1994) 550–554.
[35] H. Venkateswara, J. Eusebio, S. Chakraborty, S. Panchanathan, Deep hashing network for unsupervised domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5018–5027.
[36] D. Li, Y. Yang, Y.-Z. Song, T.M. Hospedales, Deeper, broader and artier domain generalization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5542–5550.
[37] A. Torralba, A.A. Efros, Unbiased look at dataset bias, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 1521–1528.
[38] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, B. Wang, Moment matching for multi-source domain adaptation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1406–1415.
[39] A. Krizhevsky, G. Hinton, et al., Learning Multiple Layers of Features from Tiny Images, Toronto, ON, Canada, 2009.
[40] X. Li, M. Jiang, X. Zhang, M. Kamp, Q. Dou, Fedbn: Federated learning on non-IID features via local batch normalization, in: International Conference on Learning Representations, 2021.
[41] P.P. Liang, T. Liu, L. Ziyin, N.B. Allen, R.P. Auerbach, D. Brent, R. Salakhutdinov, L.-P. Morency, Think locally, act globally: Federated learning with local and global representations, in: International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with NeurIPS, 2019.
[42] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[43] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems, vol. 32, 2019, pp. 8026–8037.
[44] A.Z. Tan, H. Yu, L. Cui, Q. Yang, Towards personalized federated learning, IEEE Trans. Neural Netw. Learn. Syst. (2022).

Myeongkyun Kang received the B.S. degrees in Software from Kookmin University, South Korea, in 2020. He is currently an integrated Master & Ph.D. student with the Department of Robotics and Mechatronics Engineering, Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu, South Korea.

Soopil Kim received the B.S. degree from School of Undergraduate Studies, Daegu Gyeongbuk Institute of Science & Technology (DGIST), South Korea, in 2019. He is currently pursuing the Master & Ph.D. integrated degree at DGIST.

Kyong Hwan Jin received the B.S. and Ph.D. degrees in the department of bio and brain engineering from KAIST, South Korea, at 2008 and 2015, respectively. He was a Postdoctoral Fellow at EPFL (2016-2019) and staff engineer at Samsung Research (2019-2021) and Assistant professor at DGIST (2021-2023). Since 2023, he has been an Associate professor at Korea University, South Korea.

Ehsan Adeli is a faculty member at the Department of Psychiatry and Behavioral Sciences, at Stanford University, and is affiliated with the Department of Computer Science. He primarily leads research at the Computational Neuroscience (CNS) Lab as well as the Partnership in AI-Assisted Care (PAC; a partnership between Stanford AI Lab and Clinical Excellence Research Center). His research interests include computer vision, ambient intelligence, computational neuroscience, medical image analysis, and AI-assisted healthcare.

Kilian M. Pohl received the PhD degree from the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology. He is currently an Associate Professor at the Department of Psychiatry and Behavioral Sciences, Stanford University.

Sang Hyun Park received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2008, and the Ph.D. degree in electrical and computer engineering from Seoul National University, Seoul, in 2014. He was a Postdoctoral Fellow at SRI International (2016-2017) and in the Biomedical Research Imaging Center at the University of North Carolina (2014-2016). Since 2017, he has been an Assistant/Associate professor at DGIST, South Korea.