

# Feature Diversification and Adaptation for Federated Domain Generalization

Seunghan Yang<sup>Ⓜ</sup>, Seokeon Choi<sup>Ⓜ</sup>, Hyunsin Park<sup>Ⓜ</sup>, Sungha Choi<sup>Ⓜ</sup>,  
Simyung Chang<sup>Ⓜ</sup>, and Sungrack Yun<sup>Ⓜ</sup>

Qualcomm AI Research<sup>†</sup>

{seunghan, seokchoi, hyunsinp, sunghac, simychan, sungrack}  
@qti.qualcomm.com

**Abstract.** Federated learning, a distributed learning paradigm, utilizes multiple clients to build a robust global model. In real-world applications, local clients often operate within their limited domains, leading to a ‘domain shift’ across clients. Privacy concerns limit each client’s learning to its own domain data, which increase the risk of overfitting. Moreover, the process of aggregating models trained on own limited domain can be potentially lead to a significant degradation in the global model performance. To deal with these challenges, we introduce the concept of federated feature diversification. Each client diversifies the own limited domain data by leveraging global feature statistics, *i.e.*, the aggregated average statistics over all participating clients, shared through the global model’s parameters. This data diversification helps local models to learn client-invariant representations while preserving privacy. Our resultant global model shows robust performance on unseen test domain data. To enhance performance further, we develop an instance-adaptive inference approach tailored for test domain data. Our proposed instance feature adapter dynamically adjusts feature statistics to align with the test input, thereby reducing the domain gap between the test and training domains. We show that our method achieves state-of-the-art performance on several domain generalization benchmarks within a federated learning setting.

**Keywords:** Federated domain generalization · Zero-shot adaptation

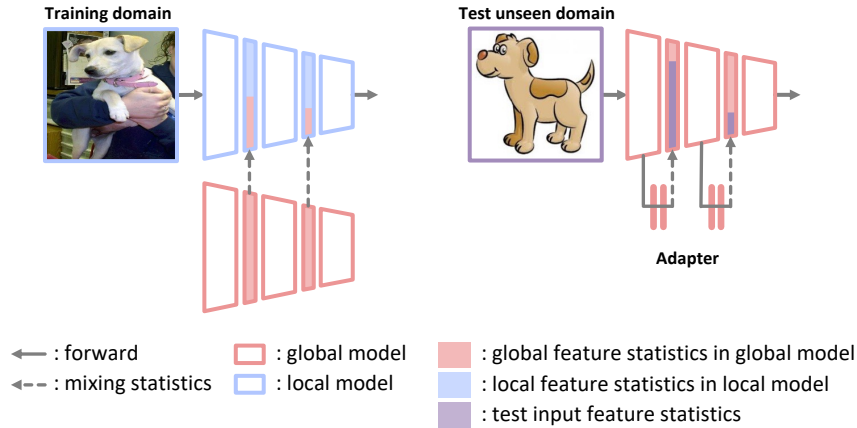
## 1 Introduction

Training a deep learning model with direct access to distributed data presents significant privacy concerns. Consequently, federated learning (FL) [1, 20, 25, 31, 32, 35, 46, 52] has surfaced as a promising solution for avoiding direct data access. Early FL research like FedAvg [35] and FedProx [31] tackled privacy by exchanging locally trained model parameters, rather than the actual data of local clients. These parameters are then aggregated on a central server to generate a global model, which is returned to the clients to continue training until stability is reached.

Contrary to common assumptions in previous studies that local data across clients originates from a uniform source domain, in practice, each client’s local data often derives from its own distinct source domain. This leads to a domain disparity among

---

<sup>†</sup> Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.



**Fig. 1: Our Concept Diagram.** Federated feature diversification: during training, local data is diversified using global feature statistics—averaged statistics across all client data—provided by the global model. This approach aids in learning representations that are invariant to individual client characteristics. Instance feature adaptation: in the test phase, the adapter employs input feature statistics to facilitate instance-adaptive inference tailored for new and unseen test data.

clients. For example, in autonomous driving scenarios, disparity arises as each vehicle captures street views and infrastructure with variations attributable to its specific camera sensors, geographic location, and other factors. Such variation in data distribution between clients results in what is known as a ‘domain shift,’ which poses a prevalent challenge in both the training and testing phases of model development.

This paper addresses these problems by exploring federated domain generalization (federated DG), which aims to improve a federated model’s learning process in handling various distributed source domains while maintaining reliable generalization to unseen domains. While past studies like SiloBN [1] and FedBN [32] have aimed to manage the domain shift within distributed source domains, they do not tackle the domain shift that occurs between training and test distributions. The attempts to enhance generalizability to unseen domains, such as FedDG [33], COPA [48], FedIns [16], CCST [5], CSAC [54], GA [55], and TsmoBN [24], have been made, but these methods have their limitations. These include potential privacy leaks through shared client information [5, 16, 33, 48], limited performance by prioritizing aggregation over local training [54, 55] and the necessity of target data for adaptation to the test domain [24]. Unlike previous works, we resolve federated DG without causing additional privacy leaks and without utilizing target data in learning.

In this study, presented in Figure 1, we propose a novel framework, named *FedFD-A*, which comprises two approaches: federated feature diversification and instance feature adaptation. Our first approach, federated feature diversification, obtains global feature statistics with the average statistics over all client data and diversifies the limited local data using the obtained global feature statistics. Specifically, we access global

feature statistics through batch normalization layers in the server model and apply a combination of local and global feature statistics to normalize local data, thereby achieving augmented features. Training with these data mitigates the overfitting of each local model to its specific domain by learning client-invariant representations. Previous approaches, such as FedDG [33], directly access other clients’ domains for data diversification, while methods like MixStyle [58] are limited to diversifying data within the same batch, where samples are collected from the same domain on the client. In contrast, our method leverages information shared with the server model. Our method maintains privacy while still effectively leveraging information across various domains.

Our second approach, instance feature adaptation, aims to enhance generalization to unseen targets. In the federated learning stage, the adapter is trained to leverage input feature statistics for mitigating domain shift between the input and the training distribution. We achieve this by normalizing the input using a combination of input and global statistics, interpolated through the adapter. At the testing stage, the adapter employs feature statistics from the test data to reduce the domain shift without needing further adjustments, unlike TsmoBN [24], which requires additional adaptation steps during inference.

We evaluate our FedFD-A components through extensive experiments on several DG benchmarks in the image classification task, including PACS [28], VLCS [13], OfficeHome [45], and DomainNet [40]. Our framework consistently enhances the performance of the federated model across multiple domain shift scenarios.

## 2 Related Works

### 2.1 Federated Learning

Federated learning (FL) has gained substantial traction due to its ability to train a global model using decentralized datasets, all while maintaining user privacy. Most recent FL methodologies have focused on addressing the issue of non-iid data spread across clients, specifically those with heterogeneous label distributions [25, 31, 46]. One notable method, FedProx [31], reduces the disparity between local and global models by incorporating a proximal term into local loss functions to regularize these local models. However, the issue of non-iid feature distribution has not garnered as much attention in previous studies. SiloBN [1] and FedBN [32] strive to manage the domain shift by preserving batch normalization (BN) statistics at a local level rather than aggregating them server-side. TsmoBN [24] aims to boost performance by updating test BN to adapt the global model towards target clients, provided they possess a substantial amount of data in the target domain. Distinct from previous studies, our research leverages the entirety of client feature statistics, allowing the effective training of local models to learn client-invariant representations. Furthermore, we introduce an instance adaptation strategy, which enables the global model to generalize directly to unseen domains using test input data alone.

### 2.2 Domain Generalization

Domain generalization aims to train a model on source domains in a manner that facilitates robust performance on unseen target data. Single-source approaches [3, 7, 23, 26,

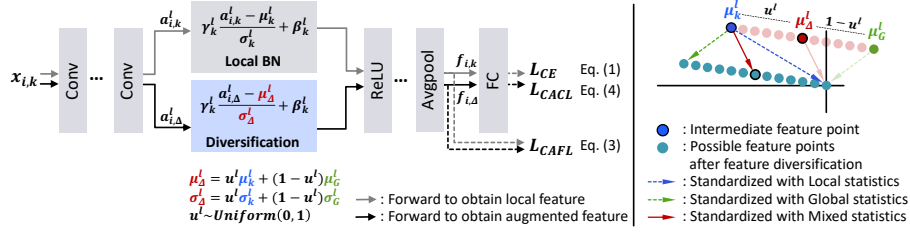
[30, 47, 51, 58] can be applied within a FL context while preserving privacy. However, these methods can underutilize the rich server information inherent in FL, thereby offering only marginal performance improvements. On the other hand, multi-source methods [4, 6, 29, 34, 37, 39, 42, 50, 56] that minimize domain discrepancies across sources necessitate raw data sharing, prompting privacy concerns within FL. Some recent studies, such as FedDG [33], COPA [48], and CCST [5], have utilized multi-source techniques in distributed settings without sharing raw data; nonetheless, they still share potentially sensitive style distributions or classifiers. The CSAC [54] approach mitigates the domain shift via an aggregation method but requires pre-training and results in a limited performance increase. Similarly, GA [55] aggregates local models taking into account client divergences, but the improvements achieved are generally marginal. FedIns [16] provides dynamic global model adjustments per testing instance, but necessitates sharing elements of local models amongst clients, potentially compromising privacy. Contrarily, our study introduces a feature diversification methodology that addresses the domain shift without causing additional privacy issues. Moreover, our instance-adaptive feature adaptation approach directly generalizes the global model to accommodate testing instances without needing to share local models.

### 3 Proposed Methods

#### 3.1 Preliminaries

**Notation and Problem Formulation:** Let  $\mathcal{X}$  and  $\mathcal{Y}$  denote the input and label spaces, respectively. The  $k$ -th client has a single-domain data  $D_k = \{(x_{i,k}, y_{i,k})\}_{i=1}^{n_k}$ , where  $n_k$  is the number of samples. The set of distributed source domain data from  $K$  clients is represented as  $\{D_1, \dots, D_K\}$ . In federated domain generalization (federated DG), there exists a domain shift across clients, where each client data  $D_k$  sampled from a domain-specific distribution  $(\mathcal{X}_k, \mathcal{Y})$  that differs from other clients. The target test domain data from an unseen environment is represented as  $D_t$ , with a distribution  $(\mathcal{X}_t, \mathcal{Y})$  that is shifted from the training data. The feature extractor, parameterized by  $\theta$ , is represented as  $F_\theta$ , and the classifier, parameterized by  $\phi$ , is represented as  $C_\phi$ . Federated DG aims to learn a generalized global model  $C_{\phi_G} \circ F_{\theta_G} : \mathcal{X} \rightarrow \mathcal{Y}$  by aggregating  $K$  distributed clients' models  $\{F_{\theta_k}, C_{\phi_k}\}_{k=1}^K$  trained on source data  $\{D_k\}_{k=1}^K$ , such that the global model generalizes to unseen target domain  $D_t$ . Note that we exploit FedAvg [35] as an aggregation method, where the global model parameters are weighted averaged with the local model parameters based on the dataset size  $\theta_G = \sum_{k=1}^K \frac{n_k}{n} \theta_k$  and  $\phi_G = \sum_{k=1}^K \frac{n_k}{n} \phi_k$ . Here,  $n$  indicates the total number of data across clients.

**Batch Normalization:** In the context of this study, batch normalization (BN) layers operate as  $\gamma_k^l \frac{a_k^l - \mu_k^l}{\sigma_k^l} + \beta_k^l$ . Here,  $\mu_k^l$  and  $\sigma_k^l$  represent the mean and standard deviation statistics for the  $l$ -th BN layer in the  $k$ -th client. These statistics reflect the overall feature statistics from data in the  $k$ -th local client by calculating a running mean and standard deviation. Additionally,  $a_k^l$  refers to an input tensor, while  $\gamma_k^l$  and  $\beta_k^l$  signify learnable scaling and shifting parameters. In this work, we employ the terms 'local statistics' to refer to  $\mu_k$  and  $\sigma_k$ , while 'global statistics' represents  $\mu_G$  and  $\sigma_G$ , which are computed by aggregating the local statistics.



**Fig. 2: Federated Feature Diversification** begins with obtaining the local feature  $f_{i,k}$  using batch normalization (BN) at the individual client level. Simultaneously, the augmented feature  $f_{i,\Delta}$  is derived through mixed statistics, implemented by interpolating local and global statistics randomly. The intermediate feature  $a_{i,\Delta}^l$  is then standardized with these statistics, leading to a range of features (as displayed on the right figure). This procedure is consistently applied across all BN layers to produce a greater variety of features. These diversified features are then employed to learn client-invariant representations, as specified by Eq. 3 and 4.

### 3.2 Federated Feature Diversification

In local training, the  $k$ -th local model is trained with the cross-entropy loss function  $\text{CE}(\cdot)$  on its dataset as follows:

$$\mathcal{L}_{CE} = \frac{1}{n_k} \sum_{i=1}^{n_k} \text{CE}(C_{\phi_k}(F_{\theta_k}(x_{i,k})), y_{i,k}). \quad (1)$$

Before starting local training, each client receives the global model parameters  $\{\theta_G, \phi_G\}$ . Using these parameters, the local model  $\{\theta_k, \phi_k\}$  is initialized. Subsequently,  $F_{\theta_k}$  and  $C_{\phi_k}$  are trained on its data during local epochs. Despite initializing the local model with the global model’s parameters, which are generally robust, a challenge remains. The problem emerges when the local model relies solely on the cross-entropy loss from a single domain, mostly due to privacy concerns associated with using data from other clients directly. This approach can lead to overfitting to the data of the single domain, causing a significant divergence in the local client models. Such divergence can pose difficulties in combining the local models to form the generalized global model as noted in literature [1, 32]. To mitigate this issue, we propose training the local model with data that is diversified across multiple domains while maintaining privacy of the clients. This varied data is created using our proposed method of data diversification.

**Diversification Process:** In federated learning, we accumulate both local and global statistics through BN layers. These global statistics reflect the feature statistics across all clients—a property we utilize for data augmentation during local training, as shown in Fig. 2. In conventional local training, the input tensor  $a_{i,k}^l$  is normalized using batch samples’ statistics in the  $l$ -th BN layer. The local feature  $f_{i,k}$  is then consistently calculated based on statistics derived from single domain local data. Such an approach restricts the local model to learn representations only within a single domain. We propose the resolution of this limitation by normalizing input tensors with a mix of statistics—exploiting global and local statistics concurrently. These mixed statistics are calcu-

lated as follows:

$$\mu_{\Delta}^l = u^l \mu_k^l + (1 - u^l) \mu_G^l \text{ and } \sigma_{\Delta}^l = u^l \sigma_k^l + (1 - u^l) \sigma_G^l. \quad (2)$$

We define  $u^l \in R^{C^l}$  as an interpolation weight vector, where each element is independently drawn from uniform distribution  $U(0, 1)$  with every iteration. Here,  $C^l$  is the feature dimension in the  $l$ -th BN layer. The local statistics can be calculated on a batch or instance basis. However, we opt to use instance-based statistics to generate a more diverse range of domains.

Features normalized by local statistics reflect local characteristics, while features normalized by global statistics incorporate global representations. Random interpolation of these two types of statistics across all BN layers generates a rich diversity of data that fully exploits the features of both local and global domains. In Fig. 2, we denote the intermediate feature by  $a_{i,\Delta}^l$  normalized with mixed statistics, and the augmented feature  $f_{i,\Delta}$  is obtained with  $\{\mu_{\Delta}^l, \sigma_{\Delta}^l\}_{l=1}^L$ . We train the local model using both  $f_{i,k}$  and  $f_{i,\Delta}$  in a client-agnostic way, which is described in the following section. In contrast to previous works [30, 58] that augment features using random noise values or styles of batch samples, our approach employs aggregated feature statistics—safely creating diverse augmentations in line with multi-source domain generalization. Note that our method uses global statistics in the server, which can reduce privacy leakage.

**Client-agnostic Learning Objectives (CAL):** We utilize a client-agnostic feature loss (CAFL) function as follows:

$$\mathcal{L}_{CAFL} = \frac{1}{n_k} \sum_{i=1}^{n_k} \|f_{i,k} - f_{i,\Delta}\|_2^2. \quad (3)$$

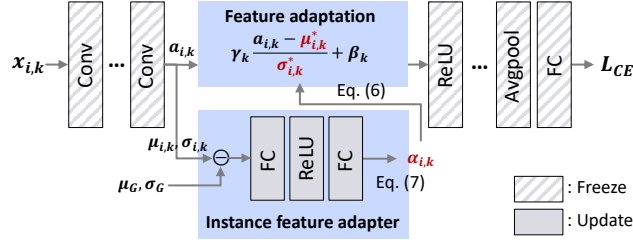
This loss function forces the local model to learn client-invariant representations, which is achieved by minimizing the variation between original and diversified features. Further, we train the local classifier to categorize the diversified features, thus encouraging the classifier to disregard client-specific information. This is achieved through our client-agnostic classification loss (CACL) function as follows:

$$\mathcal{L}_{CACL} = \frac{1}{n_k} \sum_{i=1}^{n_k} \text{CE}(C_{\phi_k}(f_{i,\Delta}), y_{i,k}). \quad (4)$$

Our client-agnostic learning functions as a regularization method, which restrains local models from diverging significantly from the global model. Different from the previous work [31] that directly regularizes local weight parameters in line with the global model, our proposed learning evaluates the significance of weight parameters for client-invariant representations within diverse domain data. Finally, the loss for local optimization is provided as follows:

$$\mathcal{L}_{total} = (1 - \lambda_1) \cdot \mathcal{L}_{CE} + \lambda_1 \cdot \mathcal{L}_{CACL} + \lambda_2 \cdot \mathcal{L}_{CAFL}, \quad (5)$$

where  $\lambda_1$  and  $\lambda_2$  are balancing parameters.



**Fig. 3: Instance Feature Adaptation.** The instance feature adapter takes the difference between instance and global statistics and generates interpolated statistics through the Eq. 6 and 7. This procedure is consistently applied across all BN layers to effectively bridge the domain gap.

### 3.3 Instance Feature Adaptation

At test time, it can be difficult to generalize to completely unseen domains where the data distribution has shifted from the training set. Addressing this domain discrepancy, we propose utilizing test data statistics, drawing inspiration from domain generalization and test-time adaptation methods. Previous domain generalization methods attempted to bridge the domain gap using the most relevant BN layer [4, 59] with the test input among multiple source BN layers. However, these methods are not feasible in a federated setting in which local models can only access the global, not the other clients’ BN layers due to privacy constraints. Moreover, recent test-time adaptation methods [17, 21, 53] use interpolated statistics between instance and learned statistics to reflect the test input distribution, yet they require manual or rule-based generation of interpolation parameters. Instead, we propose a learning-based network to dynamically generate instance-wise interpolation parameters for interpolated statistics.

**Interpolated BN Statistics:** To simplify the notation, we describe only the process at the  $l$ -th BN layer and omit the notation related to the sequence of BN layers. We leverage both instance statistics from the test input  $x_{i,t}$  and global statistics as follows:

$$\mu_{i,t}^* = \alpha_{i,t}\mu_{i,t} + (1 - \alpha_{i,t})\mu_G \text{ and } \sigma_{i,t}^* = \alpha_{i,t}\sigma_{i,t} + (1 - \alpha_{i,t})\sigma_G, \quad (6)$$

where  $\mu_{i,t}$  and  $\sigma_{i,t}$  are the instance mean and standard deviation of the input tensor.  $\mu_G$  and  $\sigma_G$  reflects the overall feature statistics of local clients involved in training. Our method leverages  $\mu_{i,t}^*$  and  $\sigma_{i,t}^*$  to normalize the test input tensor, with  $\alpha_{i,t}$  serving as an interpolation parameter that modulates the impact of instance statistics. Ideally, the optimal  $\alpha_{i,t}$  would be determined upon the availability of the test domain, but this information is typically inaccessible. Consequently, we propose an instance adaptation that dynamically produces  $\alpha_{i,t}$  for each test input in a careful and well-designed manner.

**Design of Instance Adapter:** We construct an instance adapter  $G_{\varphi_G}$ , parameterized by  $\varphi_G$ , and integrated into each BN layer in the feature extractor. The adapter takes the channel-wise distance between instance and global statistics as its input and generates  $\alpha_{i,t}$ . This design is motivated by the out-of-distribution (OOD) detection work [9], which leverages the distance between test inputs and learned statistics for OOD sample selection. Through this design, the adapter generates the appropriate interpolation parameters based on the discrepancy between the test and training distribution of each

layer. Here, we set  $\alpha_{i,t}$  as a scalar value to interpolate instance and global statistics with the same weights along the channel axis in the BN layer. It reduces the network size of the adapter and mitigates overfitting to the local data.

**Training Process:** In the  $k$ -th client, we simulate test conditions where instance statistics differ from global statistics. Instead of using test samples, we input the training sample  $x_{i,k}$  into the main network,  $F_{\theta_k}$  and  $C_{\phi_k}$ . As shown in Fig. 3, BN statistics are replaced with interpolated BN statistics in Eq. 6, where  $\alpha_{i,k}$  is generated by a local adapter  $G_{\varphi_k}$  based on the difference between instance and global statistics. We employ  $\mathcal{L}_{CE}$  to train the adapter, enabling it to generate the optimal interpolation parameters in an instance-adaptive fashion. Although the training data are more similar to global statistics compared to the test data, leveraging their differences teaches the adapter how to balance these statistics. Here, the main network focuses on generalization through feature diversification, while the adapter further adapts to unseen test domain data via instance feature adaptation. Thus, we implement alternating training to maintain this separation.

Moreover, to prevent over-fitting of the adapter, we apply the reparameterization trick [27], generating interpolation parameters by sampling from gaussian distribution reparameterized by the adapter:

$$\alpha_{i,k} = \text{T}(z \cdot \delta_{i,k} + \epsilon_{i,k}), \text{ where } [\delta_{i,k}; \epsilon_{i,k}] = G_{\varphi_k}([\mu_{i,k} - \mu_G; \sigma_{i,k} - \sigma_G]), \quad (7)$$

with  $z$  sampled from  $N(0, 1)$ .  $\text{T}(\cdot)$  is a clamp function to ensure  $\alpha_{i,k}$  within the range  $[0, 1]$ .  $[\mu_{i,k} - \mu_G; \sigma_{i,k} - \sigma_G]$  represents the concatenation of the difference between instance and global statistics along the channel axis. During the federated learning stage, the local adapter  $G_{\varphi_k}$  is trained on each client’s data, and these local adapters are subsequently aggregated into the server adapter  $G_{\varphi_G}$ .

**Test Process:** We use the server adapter  $G_{\varphi_G}$  to get  $\delta_{i,t}$  and  $\epsilon_{i,t}$ . In Eq. 7, we can then calculate  $\alpha_{i,t}$  as  $\text{T}(\epsilon_{i,t})$  because the mean of  $z$  is 0. In every BN layer, normalization is carried out using the interpolated statistics in Eq. 6, and the procedure requiring only single forward propagation. The computational cost is evaluated in the experimental section. FedFD refers to the method using feature diversification, while FedFD-A denotes the approach that also utilizes the adapter. The pseudo-code of our algorithm can be found in supplementary material.

## 4 Experiments

### 4.1 Experimental Setup

**Datasets and Settings:** We evaluate the performance of our FedFD-A on four benchmarks in the domain generalization setting of image classification. We use the PACS [28], VLCS [13], OfficeHome [45], and DomainNet [40] datasets. PACS contains seven categories from Photo (P), Art painting (A), Cartoon (C), and Sketch (S) domains, notable for their substantial domain shifts [4, 10]. VLCS consists of five categories from VOC2007 (V) [12], LabelMe (L) [41], Caltech-101 (C) [15], and SUN (S) [49] domains, primarily experiences domain shifting due to camera type variations. Comparatively, OfficeHome [45], which includes 65 categories from Artistic (A), Clipart (C),



Product (P), and Real world (R) domains, faces a less drastic issue of the domain shift [11, 48]. Consistent with prior federated DG research [33, 48, 54], we assume that each client has data from a single domain and that there are a total of four clients in each domain generalization benchmark. The global model is collaboratively learned with three of the clients and is then evaluated on the remaining domain.

For a more comprehensive understanding of our FedFD-A in practical settings, please refer to supplementary material. This includes further experiments on federated DG with the DomainNet [40] dataset, exploration of an increased number of clients, and scenarios where non-iid label distribution is also present. These scenarios provide valuable insights into the realistic application of our model.

**Implementation Details:** In the federated DG scenario, all clients employ the same architecture and hyper-parameter configurations. The backbone network is a ResNet-18 [19] model pre-trained on ImageNet. In FedFD-A, two fully connected layers are introduced before each BN layer to serve as the instance adapter, where output size is 2. The network is trained via Stochastic Gradient Descent (SGD) with a fixed learning rate of 0.01, momentum of 0.5, and batch size of 64 over 200 iterations per round. We conduct 40 rounds of training. The values of  $\lambda_1$  and  $\lambda_2$  in Eq. 5 are set to 0.1 and 4.0, respectively. Ablation studies of balancing parameters are presented in supplementary material.

**Evaluation Protocols:** In our evaluation of the federated DG performance, we adopt the standard protocols specified in DomainBed [18] for dataset splits and model selection strategies. Specifically, we use the training-domain validation set for model selection. To obtain the best global model in the federated DG setting, different selection strategies are adopted for the client side and the server side. On the client side, the local model is uploaded to the server when the validation accuracy on its local domain is best within 200 iterations, validating every 20 iterations. On the server side, the best round model is selected when the average validation performance on participating clients is maximized among 40 rounds, validating every round. The server can access clients' validation performance using the aggregated server parameters, making it possible to use the average validation performance on participating clients. This strategy of adopting single-source DG validation on local training and multi-source DG validation on the server is both practical and effective. To ensure a fair comparison, we reproduce competitive methods in our federated DG setting. The results are reported as the average accuracy and standard deviation over four runs with different random seeds.

**Baseline Frameworks:** Our FedFD-A is designed to be compatible with a range of federated learning frameworks. Specifically, we have applied our approach in conjunction with FedBN [32] and SiloBN [1], both of which aim to mitigate the domain shift by preserving batch normalization (BN) parameters locally. FedBN retains both BN statistics and affine parameters, while SiloBN only preserves BN statistics, thus enabling the BN layer to implicitly learn client-specific information. Note that parameters are aggregated for the global model with the same process of FedAvg [35]. In Table 1, we demonstrate that our proposed FedFD-A method significantly improves the performance of baseline methods by approximately 5-7%. Among the methods, SiloBN is particularly effective in reducing the domain gap and exhibits the highest performance, as reported in [14]. For the remainder of our experiments, we focus solely on the SiloBN framework.

**Table 1:** Accuracy (%) on PACS using FedFD-A on the baselines.

Methods	Acc.	Methods	Acc.	Methods	Acc.
FedAvg [35]	77.35	FedBN [32]	79.57	SiloBN [1]	80.79
+ FedFD-A	84.22	+ FedFD-A	85.15	+ FedFD-A	85.53

**Table 2:** Accuracy (%) of feature diversification variants using various mixing strategies of global and instance statistics on PACS.

Mixing components	Acc.
Global only: $u = 0.0$	81.75
Instance only: $u = 1.0$	82.52
Global & instance fixed: $u = 0.5$	83.43
<b>Global &amp; instance randomly: <math>u \sim U(0, 1)</math></b>	<b>84.07</b>

## 4.2 Ablation Studies of Our Components

**Variants of Federated Feature Diversification:** We perform experiments to examine the properties of our feature diversification approach in Table 2. SiloBN is used as the baseline, and two client-agnostic learning objectives are applied using diversified data. In this experiment, we excluded our adapter to focus solely on diversification analysis.

When solely global statistics are utilized, set as  $u = 0.0$  in Eq. 2, the inputs are normalized using the statistics of the global model. These global statistics encapsulate the feature statistics across all training clients. By augmenting local data through normalizing with these statistics, the model is primed to learn domain-invariant representations. Conversely, when utilizing only instance statistics, the style-removed features, proposed in [22], become more similar to the original features. This indicates that domain-invariant information can be taught within the local domain. Normalizing inputs with both global and instance statistics leads to performance improvements of 0.96% and 1.73%, respectively, on the PACS dataset’s average results. By combining global and instance statistics, the model can benefit from both types of information, with domain-invariant information being learned from both global and local domains. Additionally, the use of randomly mixed statistics introduces a broader diversity of domain data, further elevating performance. We dive deeper into the impacts of the distribution range in our supplementary material.

**Comparison with L2 Weight Regularization:** As depicted in Table 3-(a), we evaluate the effects of weight regularization using the FedProx method [31], which entails L2 regularization with  $\mu$  set to 0.1. Using only global statistics in client-agnostic learning serves as a type of regularization that acts to prevent deviation from the global model. Notably, this method surpasses FedProx, achieving 81.75% accuracy in comparison to FedProx’s 80.86% accuracy. Our approach goes beyond simple L2 regularization by training the model while considering the relevance of model parameters, aiming to generate client-invariant representations.

**Effectiveness of Federated Feature Diversification:** Our FedFD stands as a feature augmentation technique specifically designed for federated DG, effectively leveraging the federated characteristics that provide server model access. This contrasts with

**Table 3:** Ablation studies comparing the individual components of our model, with **ours** denoted in bold.

	Aug.	Loss	Inference	Acc. (%)				
				P	A	C	S	Avg.
(a) Baselines	X	CE FedProx	Global statistics	92.74 (0.97)	77.08 (0.59)	75.08 (0.85)	78.28 (1.14)	80.79
				92.82 (0.90)	76.43 (0.93)	75.26 (0.99)	78.95 (1.64)	80.86
(b) Effectiveness of FedFD	Mixstyle FedFD	CAL	Global statistics	92.04 (0.63)	80.19 (0.28)	77.14 (0.48)	82.53 (0.94)	82.98
				<b>92.99 (0.61)</b>	<b>82.17 (1.15)</b>	<b>77.71 (1.13)</b>	<b>83.40 (0.19)</b>	<b>84.07</b>
(c1) Effectiveness of instance feature adapter	X	CE	Global statistics	92.74 (0.97)	77.08 (0.59)	75.08 (0.85)	78.28 (1.14)	80.79
			Instance statistics (TBN [36])	15.52 (4.89)	10.91 (2.41)	9.36 (7.09)	16.09 (9.44)	12.97
			<b>Instance feature adapter</b>	<b>93.11 (0.08)</b>	<b>80.93 (0.79)</b>	<b>77.01 (0.24)</b>	<b>82.54 (0.79)</b>	<b>83.40</b>
(c2) Comparison with other inference methods	FedFD	CAL	BIN [37]	93.86 (0.53)	78.81 (0.45)	79.54 (0.40)	81.94 (0.88)	83.54
			DSON [42]	93.70 (0.39)	83.17 (1.09)	76.08 (1.30)	83.14 (1.02)	84.02
			IABN [17]	93.29 (0.13)	80.18 (1.16)	79.38 (1.33)	83.58 (0.91)	84.11
			MixNorm [21]	93.60 (0.52)	82.77 (0.41)	78.30 (1.13)	83.12 (0.74)	84.45
			Random alpha	90.67 (0.59)	74.82 (2.97)	74.96 (0.76)	79.03 (1.64)	79.87
			Fixed alpha [53]	93.59 (0.25)	79.27 (1.40)	79.64 (1.00)	82.43 (1.12)	83.73
			<b>Instance feature adapter</b>	<b>94.24 (0.33)</b>	<b>84.30 (0.44)</b>	<b>79.80 (1.16)</b>	<b>83.79 (0.49)</b>	<b>85.53</b>

Mixstyle [58], which diversifies styles through the random mixing of instance statistics within batch samples, enabling the learning of style-agnostic features. As observed in Table 3-(b), in terms of performance across all domains, our FedFD consistently outperforms Mixstyle in a fair comparison where Mixstyle substitutes FedFD. This performance superiority stems from the utilization of global statistics, while Mixstyle is limited to single-domain data access during local training. Such findings highlight that augmenting features with global statistics can enhance the model’s domain robustness.

**Comparison Studies of Instance Feature Adapter:** In Table 3-(c), we analyze the effectiveness of our instance feature adapter. We first apply our adapter to the baseline model, SiloBN, and demonstrate its effectiveness when using instance statistics along with global statistics during testing. In Table 3-(c1), we compare the performance of the model trained with SiloBN using global statistics for testing (conventional approach), instance statistics for testing with TBN [36] holding a batch size of one, and our instance-wise approach of appropriately combining global and instance statistics during testing. Our approach improves performance by reflecting the test domain with instance statistics on global statistics that are sensitive to distribution shift.

In the following step, we present a comparison of our adapter with various inference methods in Table 3-(c2), where we utilize FedFD as the baseline. Specifically, we compare our approach with BIN [37] that employs learned alpha to incorporate batch normalization (BN) and instance normalization (IN) layers in the backbone network. During training, the interpolation parameters are optimized and then applied to test samples. However, since the interpolation parameters are fitted to the training data, they can not generalize well to unseen domains. Ensemble predictions (DSON [42]) from multiple local BN layers show degradation in performance on domains that are severely different from the training domains and also raise privacy concerns due to the sharing of multiple local statistics. IABN [17] calibrates learned statistics with instance statistics using a rule-based function, but it requires sensitive hyper-parameters to be properly selected for each domain. Our adapter outperforms IABN on all domains without the need for sensitive hyper-parameters. Additionally, we compare our adapter with MixNorm [21], which augments test input data with spatial augmentations to estimate the test distri-

**Table 4:** Comparison of computational cost and accuracy (%) on the PACS dataset.

Methods	#Params.	Training	Inference	Acc. (%)
FedAvg [35]	11.31M	1.43ms	1.34ms	77.35
SiloBN [1]	11.31M	1.43ms	1.34ms	80.79
<b>FedFD</b>	11.31M	2.25ms	1.34ms	84.07
BIN [37]	11.31M	3.24ms	1.35ms	83.54
MixNorm [21]	11.31M	2.25ms	6.70ms	84.45
IABN [17]	11.31M	2.25ms	1.57ms	84.11
<b>FedFD-A</b>	11.53M	3.50ms	1.37ms	85.53

bution more accurately. While it slightly improves performance, it also increases the inference time and memory usage. We compare with the naive approaches of random and fixed alpha [53]. This comparison demonstrates that layer-wise and instance-wise generation strategies are markedly more effective than utilizing a fixed alpha strategy, further validating our adaptive approach.

**Computational Overhead Analysis:** Table 4 presents the computational cost analysis of the various methods. The time required for training and inference was measured as the average per-iteration time with a batch size of 64 on an Intel Xeon Gold 6342 processor and a single NVIDIA RTX A5000 graphics card. FedAvg, SiloBN, and FedFD do not incur additional computational overhead during inference, while BIN requires additional processing time to calculate instance normalization. In addition to FedFD, we also consider computation efficiency and performance of FedFD-A by incorporating inference methods such as MixNorm and IABN. MixNorm requires five times more memory and computation compared to other methods, as it forward passes multiple data (in this case, five) per batch sample. IABN calculates instance statistics at each BN layer and derives the corresponding interpolation parameter using a rule-based function, which results in better computational efficiency compared to MixNorm, but with lower performance improvement. FedFD-A achieves the best accuracy with a marginal increase in parameters and computation. Our FedFD and FedFD-A balance performance and computational overhead, offering superior efficiency and performance compared to other methods. Users can select the appropriate method based on computational requirements.

### 4.3 Comparison with Competitive Methods

**Competitive Methods:** In Table 5, we compare our FedFD-A with representative methods of FL and DG in the classification task. (1) Federated learning methods: FedAvg [35], FedProx [31], FedBN [32], and SiloBN [1]; (2) A data augmentation method: RandAug [8]; (3) Augmentation-based DG methods: Mixstyle [58], SFA [30], RandConv [51], and L2D [47]; (4) Regularization-based DG methods: JiGen [3], RSC [23], and Self-Reg [26]; (5) Federated DG methods: COPA [48], FedDG [33], CCST [5], CSAC [54], FedSR [38], FedSAM [2], and GA [55].

**Performance Analysis:** Our analysis focuses on the results obtained from PACS, where the domain distribution shift is substantial. It is observed that decentralized methods without DG fail to provide high performance compared to other approaches. While FedProx regularizes the local models, it fails to address the domain shift. FedBN and

**Table 5:** Comparison results on PACS, VLCS, and OfficeHome. Methods displayed in gray are associated with privacy issues.

Paradigm	Method	PACS					VLCS	OfficeHome	Avg. of 3 datasets
		P	A	C	S	Avg.	Avg.	Avg.	
Decentralized w/o DG	FedAvg [35]	90.40 (0.97)	72.52 (2.28)	72.59 (0.37)	73.90 (1.74)	77.35	74.86	63.47	71.89
	FedProx [31]	90.49 (0.69)	72.41 (1.06)	73.09 (0.91)	72.93 (1.48)	77.23	74.42	63.02	71.56
	FedBN [32]	92.63 (0.65)	76.22 (1.05)	72.82 (2.50)	76.59 (2.48)	79.57	74.85	63.75	72.72
	SiloBN [1]	<b>92.74 (0.97)</b>	<b>77.08 (0.59)</b>	<b>75.08 (0.85)</b>	<b>78.28 (1.14)</b>	<b>80.79</b>	<b>75.33</b>	<b>64.08</b>	<b>73.40</b>
Decentralized w/ DG	RandAug [8]	93.57 (0.60)	77.15 (1.39)	71.68 (2.14)	66.64 (2.34)	77.26	74.32	<b>64.59</b>	72.06
	Mixstyle [58]	92.75 (0.45)	<b>80.21 (0.33)</b>	<b>76.77 (0.72)</b>	79.98 (2.22)	82.43	<b>75.54</b>	63.24	73.74
	SFA [30]	87.22 (1.76)	72.01 (3.22)	73.12 (1.03)	79.52 (1.63)	77.98	72.77	59.47	70.06
	RandConv [51]	91.78 (0.40)	77.16 (1.55)	70.56 (3.28)	78.78 (1.46)	79.57	73.68	63.25	72.17
	L2D [47]	<b>93.59 (0.55)</b>	79.69 (1.12)	75.81 (1.37)	<b>82.76 (0.55)</b>	<b>82.96</b>	75.37	63.29	<b>73.87</b>
	JiGen [3]	92.14 (0.40)	72.51 (2.00)	72.76 (1.24)	73.34 (1.96)	77.69	75.13	64.09	72.30
RSC [23]	92.53 (0.86)	78.10 (0.56)	<b>75.95 (1.08)</b>	<b>79.82 (1.31)</b>	<b>81.60</b>	<b>75.90</b>	63.01	<b>73.50</b>	
SelfReg [26]	<b>93.41 (0.76)</b>	<b>78.30 (1.16)</b>	74.94 (0.43)	77.13 (2.09)	80.94	72.79	<b>64.85</b>	72.86	
Decentralized w/ federated DG	COPA [48]	94.70 (1.07)	83.75 (0.29)	78.58 (0.96)	84.45 (1.33)	85.37	74.51	62.42	74.10
	FedDG [33]	94.45 (0.44)	83.83 (0.28)	73.41 (1.33)	78.40 (0.73)	82.52	75.28	64.90	74.23
	CCST [5]	93.59 (0.49)	80.22 (0.55)	77.79 (1.95)	82.61 (0.84)	83.56	75.20	63.50	74.09
	CSAC [54]	94.83 (0.42)	80.48 (1.25)	75.46 (1.58)	79.56 (1.35)	82.58	75.21	64.35	74.05
	FedSR [38]	91.71 (0.62)	76.49 (1.26)	72.87 (1.56)	75.46 (2.22)	79.13	75.36	63.45	72.65
	FedSAM [2]	90.73 (0.89)	74.13 (1.96)	73.46 (0.99)	75.97 (1.39)	78.58	74.37	61.93	71.63
	SiloBN w/ GA [55]	<b>94.91 (0.59)</b>	78.10 (1.42)	71.15 (0.67)	79.17 (0.50)	80.83	75.41	64.19	73.48
	<b>FedFD</b>	92.99 (0.61)	82.17 (1.15)	77.71 (1.13)	83.40 (0.19)	84.07	76.62	64.01	74.90
	<b>FedFD-A</b>	94.24 (0.33)	<b>84.30 (0.44)</b>	<b>79.80 (1.16)</b>	<b>83.79 (0.49)</b>	<b>85.53</b>	<b>76.68</b>	<b>64.71</b>	<b>75.64</b>

SiloBN mitigate the domain shift through the use of local BN layers, but they are limited in their performance as they do not tackle the domain shift that occurs between training and test distributions. The decentralized methods with DG demonstrate improved performance across several domains. In particular, augmentation-based DG methods achieve substantial accuracy improvements, although there are some domains where performance is significantly degraded. This is due to the fact that such methods only learn domain-invariant representations within a single domain and are not effective if the augmentation applied to the source domain does not cover the test distribution. JiGen, RSC, and SelfReg consistently improve or maintain the performance on four domains compared to FedAvg, but the improvement is very marginal. While COPA, FedDG, and CCST achieve competitive performance across multiple domains by enabling the model to learn client-invariant representations from other clients, they do so at the expense of serious privacy concerns. On the other hand, our methods, FedFD and FedFD-A, deliver state-of-the-art performance without compromising client privacy. It is noteworthy that amongst the methods that preserve privacy, ours exhibits superior performance. On VLCS and OfficeHome, where the domain shift is relatively small, FedFD and FedFD-A consistently improve the performance on almost all domains compared to other methods, demonstrating their robustness across a wide range of domains. Further results on in-domain performance and a cross-silo FL setting can be found in supplementary material.

**Table 6:** Comparison results on PACS and VLCS using the transformer architecture.

(a) Performance (%) on PACS						(b) Performance (%) on VLCS							
Method	PACS					Avg.	Method	VLCS					Avg.
	P	A	C	S				V	L	C	S		
DoPrompt [57]	99.24 (0.16)	90.56 (0.54)	81.06 (0.42)	77.71 (0.56)		87.14	DoPrompt [57]	78.32 (1.29)	<b>65.94 (1.28)</b>	97.02 (0.49)	77.80 (1.22)		79.82
DoPrompt w/ GA [55]	99.33 (0.24)	89.98 (0.53)	81.03 (0.24)	79.01 (1.21)		87.34	DoPrompt w/ GA [55]	79.84 (0.49)	65.15 (0.81)	97.14 (0.17)	76.72 (0.59)		79.71
FedFD-A	<b>99.39 (0.06)</b>	<b>92.29 (0.42)</b>	<b>83.88 (1.07)</b>	<b>79.62 (1.23)</b>		<b>88.79</b>	FedFD-A	<b>81.41 (0.81)</b>	65.06 (0.62)	<b>97.46 (0.79)</b>	<b>79.64 (0.94)</b>		<b>80.89</b>

## 5 Discussion

In this study, we assume the utilization of BN layers in local models, an assumption that may not always be applicable. Our objective is to leverage both local and global characteristics during the local models’ training stage. The decision to employ BN layers stems from their capacity to encapsulate domain-specific information. Even in the absence of BN layers in certain models, there exist components that learn domain-specific information, such as parts of the model or hypernetworks [43, 44, 57]. We can apply random interpolation to these components between the local and global models.

We have tested the effectiveness of our method on Transformer-based multi-source domain generalization with the use of DoPrompt [57]. DoPrompt introduces domain-specific prompts to the input during training to capture domain-related information. Each domain prompt is optimized for a single domain, enabling efficient learning of domain-specific knowledge. Although DoPrompt includes a prompt adapter that generates appropriate prompts for each input image based on the learned domain prompts, we chose not to utilize the prompt adapter in our demonstrations to focus on the efficacy of our federated diversification approach. We adapt DoPrompt to a federated learning setting with domain prompts placed locally, akin to SiloBN [1] and FedBN [32], and employ our FedFD algorithm using local and global domain prompts to create interpolated prompts. We then train the local models with our proposed loss function and evaluate our final aggregated global model on unseen domains. We show the results of applying DoPrompt in federated learning, with and without GA [55], and our method in Table 6. The experiment shows that our method extends to models with various domain-specific components, moving beyond those that rely solely on batch normalization.

## 6 Conclusion

We propose two methods specifically designed for federated domain generalization. The first method suggests augmenting local data into various domains by utilizing the global model at the server. This technique encourages each local model to learn client-invariant representations. To improve the model’s performance on unseen domains after training, we propose utilizing an instance adaptation method. This approach utilizes the test sample’s instance statistics dynamically during inference when a test sample is presented. Our methods were rigorously verified through a series of experiments on domain generalization benchmarks. The results effectively demonstrated their applicability and efficiency in a federated learning environment.

## References

1. Andreux, M., du Terrail, J.O., Beguier, C., Tramel, E.W.: Siloed federated learning for multi-centric histopathology datasets. In: Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning: Second MICCAI Workshop, DART 2020, and First MICCAI Workshop, DCL 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4–8, 2020, Proceedings 2. pp. 129–139. Springer (2020) [1](#), [2](#), [3](#), [5](#), [9](#), [10](#), [12](#), [13](#), [14](#)
2. Caldarola, D., Caputo, B., Ciccone, M.: Improving generalization in federated learning by seeking flat minima. In: Eur. Conf. Comput. Vis. pp. 654–672. Springer (2022) [12](#), [13](#)
3. Carlucci, F.M., D’Innocente, A., Bucci, S., Caputo, B., Tommasi, T.: Domain generalization by solving jigsaw puzzles. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2229–2238 (2019) [3](#), [12](#), [13](#)
4. Chen, C., Li, J., Han, X., Liu, X., Yu, Y.: Compound domain generalization via meta-knowledge encoding. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7119–7129 (2022) [4](#), [7](#), [8](#)
5. Chen, J., Jiang, M., Dou, Q., Chen, Q.: Federated domain generalization for image recognition via cross-client style transfer. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 361–370 (January 2023) [2](#), [4](#), [12](#), [13](#)
6. Chen, Y., Wang, Y., Pan, Y., Yao, T., Tian, X., Mei, T.: A style and semantic memory mechanism for domain generalization. In: Int. Conf. Comput. Vis. pp. 9164–9173 (2021) [4](#)
7. Choi, S., Das, D., Choi, S., Yang, S., Park, H., Yun, S.: Progressive random convolutions for single domain generalization. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 10312–10322 (June 2023) [3](#)
8. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: IEEE Conf. Comput. Vis. Pattern Recog. Worksh. pp. 702–703 (2020) [12](#), [13](#)
9. Dong, X., Guo, J., Li, A., Ting, W.T., Liu, C., Kung, H.: Neural mean discrepancy for efficient out-of-distribution detection. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 19217–19227 (2022) [7](#)
10. Dou, Q., Coelho de Castro, D., Kamnitsas, K., Glocker, B.: Domain generalization via model-agnostic learning of semantic features. Adv. Neural Inform. Process. Syst. **32** (2019) [8](#)
11. Du, D., Chen, J., Li, Y., Ma, K., Wu, G., Zheng, Y., Wang, L.: Cross-domain gated learning for domain generalization. Int. J. Comput. Vis. pp. 1–16 (2022) [9](#)
12. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010) [8](#)
13. Fang, C., Xu, Y., Rockmore, D.N.: Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In: Int. Conf. Comput. Vis. pp. 1657–1664 (2013) [3](#), [8](#)
14. Fantauzzo, L., Fani, E., Caldarola, D., Tavera, A., Cermelli, F., Ciccone, M., Caputo, B.: Feddrive: generalizing federated learning to semantic segmentation in autonomous driving. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 11504–11511. IEEE (2022) [9](#)
15. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: IEEE Conf. Comput. Vis. Pattern Recog. Worksh. pp. 178–178. IEEE (2004) [8](#)
16. Feng, C.M., Yu, K., Liu, N., Xu, X., Khan, S., Zuo, W.: Towards instance-adaptive inference for federated learning. In: Int. Conf. Comput. Vis. pp. 23287–23296 (2023) [2](#), [4](#)
17. Gong, T., Jeong, J., Kim, T., Kim, Y., Shin, J., Lee, S.J.: Robust continual test-time adaptation: Instance-aware bn and prediction-balanced memory. arXiv preprint arXiv:2208.05117 (2022) [7](#), [11](#), [12](#)

18. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. In: *Int. Conf. Learn. Represent.* (2021), <https://openreview.net/forum?id=lQdXeXDoWtI> **9**
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *IEEE Conf. Comput. Vis. Pattern Recog.* pp. 770–778 (2016) **9**
20. Hosseini, H., Park, H., Yun, S., Louizos, C., Soriaga, J., Welling, M.: Federated learning of user verification models without sharing embeddings (2021) **1**
21. Hu, X., Uzunbas, G., Chen, S., Wang, R., Shah, A., Nevatia, R., Lim, S.N.: Mixnorm: Test-time adaptation through online normalization estimation. *arXiv preprint arXiv:2110.11478* (2021) **7, 11, 12**
22. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: *Int. Conf. Comput. Vis.* pp. 1501–1510 (2017) **10**
23. Huang, Z., Wang, H., Xing, E.P., Huang, D.: Self-challenging improves cross-domain generalization. In: *Eur. Conf. Comput. Vis.* **3, 12, 13**
24. Jiang, M., Zhang, X., Kamp, M., Li, X., Dou, Q.: Tsmobn: Interventional generalization for unseen clients in federated learning. *arXiv preprint arXiv:2110.09974* (2021) **2, 3**
25. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T.: SCAFFOLD: Stochastic controlled averaging for federated learning. *PMLR* (2020) **1, 3**
26. Kim, D., Yoo, Y., Park, S., Kim, J., Lee, J.: Selfreg: Self-supervised contrastive regularization for domain generalization. In: *Int. Conf. Comput. Vis.* pp. 9619–9628 (2021) **3, 12, 13**
27. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013) **8**
28. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: *Int. Conf. Comput. Vis.* pp. 5542–5550 (2017) **3, 8**
29. Li, D., Zhang, J., Yang, Y., Liu, C., Song, Y.Z., Hospedales, T.M.: Episodic training for domain generalization. In: *Int. Conf. Comput. Vis.* pp. 1446–1455 (2019) **4**
30. Li, P., Li, D., Li, W., Gong, S., Fu, Y., Hospedales, T.M.: A simple feature augmentation for domain generalization. In: *Int. Conf. Comput. Vis.* pp. 8886–8895 (2021) **3, 6, 12, 13**
31. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems* **2**, 429–450 (2020) **1, 3, 6, 10, 12, 13**
32. Li, X., JIANG, M., Zhang, X., Kamp, M., Dou, Q.: FedBN: Federated learning on non-IID features via local batch normalization. In: *Int. Conf. Learn. Represent.* (2021), <https://openreview.net/forum?id=6YEQUn0QICG> **1, 2, 3, 5, 9, 10, 12, 13, 14**
33. Liu, Q., Chen, C., Qin, J., Dou, Q., Heng, P.A.: Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. In: *IEEE Conf. Comput. Vis. Pattern Recog.* pp. 1013–1023 (2021) **2, 3, 4, 9, 12, 13**
34. Lv, F., Liang, J., Li, S., Zang, B., Liu, C.H., Wang, Z., Liu, D.: Causality inspired representation learning for domain generalization. In: *IEEE Conf. Comput. Vis. Pattern Recog.* pp. 8046–8056 (2022) **4**
35. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: *Artificial intelligence and statistics.* pp. 1273–1282. *PMLR* (2017) **1, 4, 9, 10, 12, 13**
36. Nado, Z., Padhy, S., Sculley, D., D’Amour, A., Lakshminarayanan, B., Snoek, J.: Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963* (2020) **11**
37. Nam, H., Kim, H.E.: Batch-instance normalization for adaptively style-invariant neural networks. *Adv. Neural Inform. Process. Syst.* **31** (2018) **4, 11, 12**
38. Nguyen, A.T., Torr, P., Lim, S.N.: Fedsr: A simple and effective domain generalization method for federated learning. *Adv. Neural Inform. Process. Syst.* (2022) **12, 13**



39. Pandey, P., Raman, M., Varambally, S., Ap, P.: Generalization on unseen domains via inference-time label-preserving target projections. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 12924–12933 (2021) [4](#)
40. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Int. Conf. Comput. Vis. pp. 1406–1415 (2019) [3](#), [8](#), [9](#)
41. Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T.: Labelme: a database and web-based tool for image annotation. Int. J. Comput. Vis. **77**(1), 157–173 (2008) [8](#)
42. Seo, S., Suh, Y., Kim, D., Kim, G., Han, J., Han, B.: Learning to optimize domain specific normalization for domain generalization. In: Eur. Conf. Comput. Vis. pp. 68–83. Springer (2020) [4](#), [11](#)
43. Shamsian, A., Navon, A., Fetaya, E., Chechik, G.: Personalized federated learning using hypernetworks (2021) [14](#)
44. Sun, B., Huo, H., Yang, Y., Bai, B.: Partialfed: Cross-domain personalized federated learning via partial initialization. Adv. Neural Inform. Process. Syst. (2021) [14](#)
45. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5018–5027 (2017) [3](#), [8](#)
46. Wang, J., Liu, Q., Liang, H., Joshi, G., Poor, H.V.: Tackling the objective inconsistency problem in heterogeneous federated optimization. Adv. Neural Inform. Process. Syst. **33**, 7611–7623 (2020) [1](#), [3](#)
47. Wang, Z., Luo, Y., Qiu, R., Huang, Z., Baktashmotlagh, M.: Learning to diversify for single domain generalization. In: Int. Conf. Comput. Vis. pp. 834–843 (2021) [3](#), [12](#), [13](#)
48. Wu, G., Gong, S.: Collaborative optimization and aggregation for decentralized domain generalization and adaptation. In: Int. Conf. Comput. Vis. pp. 6484–6493 (2021) [2](#), [4](#), [9](#), [12](#), [13](#)
49. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3485–3492. IEEE (2010) [8](#)
50. Xu, Q., Zhang, R., Zhang, Y., Wang, Y., Tian, Q.: A fourier-based framework for domain generalization. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 14383–14392 (2021) [4](#)
51. Xu, Z., Liu, D., Yang, J., Raffel, C., Niethammer, M.: Robust and generalizable visual representation learning via random convolutions. In: Int. Conf. Learn. Represent. (2021), <https://openreview.net/forum?id=BVSM0x3EDK6> [3](#), [12](#), [13](#)
52. Yang, S., Park, H., Byun, J., Kim, C.: Robust federated learning with noisy labels. IEEE Intelligent Systems (2022) [1](#)
53. You, F., Li, J., Zhao, Z.: Test-time batch statistics calibration for covariate shift. arXiv preprint arXiv:2110.04065 (2021) [7](#), [11](#), [12](#)
54. Yuan, J., Ma, X., Chen, D., Kuang, K., Wu, F., Lin, L.: Collaborative semantic aggregation and calibration for separated domain generalization. arXiv e-prints pp. arXiv–2110 (2021) [2](#), [4](#), [9](#), [12](#), [13](#)
55. Zhang, R., Xu, Q., Yao, J., Zhang, Y., Tian, Q., Wang, Y.: Federated domain generalization with generalization adjustment. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3954–3963 (2023) [2](#), [4](#), [12](#), [13](#), [14](#)
56. Zhao, Y., Zhong, Z., Yang, F., Luo, Z., Lin, Y., Li, S., Sebe, N.: Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 6277–6286 (2021) [4](#)
57. Zheng, Z., Yue, X., Wang, K., You, Y.: Prompt vision transformer for domain generalization. arXiv preprint arXiv:2208.08914 (2022) [14](#)
58. Zhou, K., Yang, Y., Qiao, Y., Xiang, T.: Domain generalization with mixstyle. In: Int. Conf. Learn. Represent. (2021), <https://openreview.net/forum?id=6xHJ37MVxxp> [3](#), [6](#), [11](#), [12](#), [13](#)

59. Zhou, Z., Qi, L., Yang, X., Ni, D., Shi, Y.: Generalizable cross-modality medical image segmentation via style augmentation and dual normalization. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 20856–20865 (2022) [7](#)