BENCHMARKING ALGORITHMS FOR FEDERATED DOMAIN GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we present a unified platform to study domain generalization in the federated learning (FL) context and conduct extensive empirical evaluations of the current state-of-the-art domain generalization algorithms adapted to FL. In particular, we perform a fair comparison of nine existing algorithms in solving domain generalization either centralized domain generalization algorithms adapted to the FL context or existing FL domain generalization algorithms to comprehensively explore the challenges introduced by FL. These challenges include statistical heterogeneity among clients, the number of clients, the number of communication rounds, etc. The evaluations are conducted on three diverse datasets including PACS (image dataset covering photo, sketch, cartoon, and painting domains), iWildCam (image dataset with 323 domains), and Py150 (natural language processing dataset with 8421 domains). The experiments show that the challenges brought by federated learning stay unsolved in the realistic experiment setting. Furthermore, the code base supports fair and reproducible new algorithm evaluation with easy implementation.

1 INTRODUCTION

Federated learning (FL) Konečnỳ et al. (2016) is a distributed machine learning approach that assumes each client or device owns a local dataset and this local dataset cannot be exchanged or centrally collected because of privacy or communication constraints. Given this context, a natural paradigm for FL (e.g., FedAvg McMahan et al. (2017)) is to alternate between two stages: clients locally update the model based on its local dataset and a central server aggregates client models. Because the clients may be phones, network sensors, hospitals, or alternative local information sources, the local datasets are naturally heterogeneous between clients. Specifically, there are at least two types of realistic statistical data heterogeneity in the FL context. *Client heterogeneity* is the data heterogeneity between clients involved in training—e.g., hospitals may use different staining procedures or imaging equipment. *Train-test heterogeneity* is the data heterogeneity between the training and testing data—e.g., the performance on a new client that was not involved in training or a natural shift in real-world test data due to changes over time, location, or context.

Client heterogeneity has long been considered a statistical challenge since federated learning was introduced. FedAvg McMahan et al. (2017) has experimentally shown that their methods effectively mitigate some client heterogeneity. There are many other extensions based on the FedAvg framework tackling the heterogeneity among clients in FL Hsieh et al. (2020); Li et al. (2020); Karimireddy et al. (2020). There is an alternative setup in FL, known as the personalized setting, which aims to learn personalized models for different clients to tackle heterogeneity. Numerous recent papers have proposed FL models and algorithms to accommodate personalization Smith et al. (2017); Chen et al. (2018); Hanzely et al. (2020); T Dinh et al. (2020); Deng et al. (2020); Acar et al. (2021). However, these prior works only train the model on simple data sets such as MNIST, EMIST, and CIFAR10 and the client heterogeneity is constructed only through *class imbalance*, which assumes the ratio of data from each class is different for each client but the class distributions are homogeneous across clients. Class imbalance is a special kind of heterogeneity called prior probability shift (See section 3.1). In practice, due to the difference between the location of the local data collector (cameras, sensors, etc), real data heterogeneity is more complex than simple class imbalance as in many prior works. Furthermore, most prior FL works do not consider train-test heterogeneity but

rather assume the train and test data are drawn from the same distribution (i.e., independent and identically distributed or i.i.d).

While most FL works have usually ignored train-test heterogeneity, the task of *domain generalization* (DG) Blanchard et al. (2011) formalizes a special case of train-test heterogeneity in which the training algorithm has access to data from multiple source domains and the goal is to perform well on data from an *unseen* test domain. There is an active line of research on domain generalization in the centralized setting Muandet et al. (2013); Saito et al. (2018); Ganin & Lempitsky (2015); Long et al. (2015); Arjovsky et al. (2019); Sagawa et al. (2019); Shi et al. (2021); Li et al. (2018). Current benchmark papers Koh et al. (2021); Gulrajani & Lopez-Paz (2020) provide thorough comparisons between different algorithms for DG. However, these centralized DG benchmarks do not consider the unique constraints of the FL context. In particular, they fail to provide insights on how the client dataset heterogeneity, the number of clients, and the communication budget will influence the generalization ability. From the FL side, to the best of our knowledge, there are currently only two published works, FedADG Zhang et al. (2021) and FedDG Liu et al. (2021), that tackle DG in the FL context but their evaluations are limited in three ways. **1)** The evaluation datasets are limited in the number and diversity of domains. FedDG Liu et al. (2021) only evaluates on retinal fundus images with 4 domains and prostate T2-weighted MRI images from 6 domains, and FedADG only focuses on PACS and VLCS, which share similarities in terms of number of domains (each with only 4 domains) and sample size. **2)** Their evaluations are restricted to the case when the number of clients is equal to the number of domains, which may be an unrealistic assumption (e.g., a hospital that has multiple imaging centers or a device that is used in multiple locations). **3)** They fail to show the network effect of FL; in particular, neither work considers the influence of the number of clients on the performance, and FedADG Zhang et al. (2021) does not consider the effect of the number of communication rounds. In summary, the current DG benchmarks fail to consider challenges unique to FL, and the few FL methods for DG have limited evaluations.

Therefore, more systematic evaluation is needed both to aid in systematic progress at the intersection of FL and DG but also to aid in answering many open questions. For example, what is the performance of a direct translation of centralized DG algorithms (if applicable) to the FL context? How does client dataset heterogeneity influence the performance of current algorithms? How does the performance of current algorithms scale with the number of clients and communications rounds on complicated real-world datasets?

To answer these questions and enable systematic evaluation, we provide an experimental benchmark of DG algorithms in FL on both simple and complex multi-domain datasets. We provide a fair comparison over multiple state-of-the-art centralized DG methods adapted to the FL context as well as the current two works on federated domain generalization on three different benchmark datasets. The evaluations are conducted in three different concrete contexts. **1) Baseline:** We evaluate methods on a simple dataset (PACS) with a small number of clients and sufficient communication rounds. Additionally, we assume that all clients have some data from each of the domains. In this simple setting, most DG algorithms in the centralized setting can be directly adapted for the FL context, and we show that they achieve the comparable accuracy to centralized domain generalization. (See section 4.1) **2) Domain separation:** We assume each client only contains examples from one unique domain, and therefore, the data distributions among agents are fully heterogeneous. Current DG algorithms designed for the centralized setting fail in this setting since they require examples from multiple domains Arjovsky et al. (2019); Shi et al. (2021); Zhang et al. (2017); Gretton et al. (2006); Sun & Saenko (2016); Sagawa et al. (2019). FedDG Liu et al. (2021) and FedADG Zhang et al. (2021) first attempt to tackle this hard case, they could achieve comparable results comparing to the naïve empirical risk minimization (ERM) with FedAvg McMahan et al. (2017). (See section 4.2) **3) More realistic setting:** We evaluate methods on more realistic and complex datasets with a large number of clients, a large number of domains, and varying degrees of client dataset heterogeneity. In particular, we evaluate Py150 with total 8421 domains and 100 clients and iWildCam Koh et al. (2021) with total 323 domains and 243 clients. These settings are in sharp contrast to the baseline setting where the dataset is PACS Li et al. (2017) with only 4 domains and 2 clients. None of the above algorithms can achieve accuracy comparable to the centralized generalization accuracy within a reasonable running time. However, ERM achieves the best accuracy among them when the communication budget is fixed. (See section 4.3) We also investigate other challenges introduced by federated setting including increasing data heterogeneity, increasing the number of clients, and reducing the communication rounds. (See section 4.3) Furthermore, to enable fair and systematic

evaluation of new DG methods in FL, we provide the benchmark code base that only requires the specification of the client and server algorithms but handles the complexities of both DG and FL experimental setup including baseline model implementations, dataset loading, dataset heterogeneity, communication tracking, and metrics. (See anonymous link here.)

2 Background and Related Work

Domain generalization. Domain generalization (DG) task Blanchard et al. (2011) tackles the classification problem where the domains of test dataset and training dataset are heterogeneous, i.e., the test dataset comes from an unseen domain distribution. Given M training (source) domains $\mathcal{D}^{\text{train}}$, domain generalization can be formalized as learning a robust and generalizable model parameter θ from $\mathcal{D}^{\text{train}}$ to achieve a minimum prediction error on the unseen test domain $\mathcal{D}^{\text{test}}$ such that:

$$\min_{h} \mathbb{E}_{\mathcal{D}\sim\mathcal{D}^{\text{test}}} \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(((x,y);\theta))]. \tag{1}$$

Existing DG algorithms design alternative objectives seeking for a good approximation usually by utilizing training data from multiple domains. Schölkopf et al. (2021) claims that current machine learning model only captures superficial statistical correlation rather than underlying causal relationship between data and their labels. Inspired by causality Lopez-Paz et al. (2017); Peters et al. (2016); Heinze-Deml et al. (2018), Arjovsky et al. (2019) proposes IRM algorithm seeking invariant predictor on top of the representation. Li et al. (2018) instead, explicitly tries to align the feature representation distribution by minimizing the maximum mean discrepancy (MMD) Gretton et al. (2006). Similar to MMD, Sun & Saenko (2016) proposes CORAL which utilizes the second-order information. Besides invariant representation methods, Shi et al. (2021) formulates the DG problem using regularization method, the proposed FISH algorithm is a first-order method trying to align the gradient directions generated by data from different domains. Sagawa et al. (2019) introduces the objective that tries to control the worst-performed domain. The proposed algorithm did not directly optimize the objective by solving the minimax objective, but to solve ERM while increasing the importance of domains with larger loss during training. Instead of working on the feature space, or gradient information, Zhang et al. (2017) proposes a straightforward data augmentation principle called Mixup that trains a neural network on convex combinations of data from two distinct domains.

Federated Domain Generalization: Limited works are seeking to solve the DG task in the FL context. Liu et al. (2021) recently proposed a federated learning paradigm specifically designed for medical image classification. The proposed method requires sharing the amplitude spectrum of images among local clients. Zhang et al. (2021) applies generative adversarial network (GAN) Goodfellow et al. (2020) in the FL context, where each client contains four models: a featurizer Φ, a classifier w, a generator $\mathcal{G}$ and a discriminator $\mathcal{F}$. The featurizer Φ and classifier w solves the classification task, while the generator $\mathcal{G}$ tries to learn the feature distribution Z. During local training, FedAdg McMahan et al. (2017) introduces two steps of learning. The first step trains the featurizer Φ and classifier w by minimizing the empirical loss. The second step trains the generator $\mathcal{G}$ and the discriminator $\mathcal{F}$ using the GAN Goodfellow et al. (2020) approach. After the local training, the central server aggregates the local featurizer, classifier and generator. However, this method requires training 4 models together along with tens of hyperparameters to tune, which makes it hard to converge empirically.

Table 1: Different Tasks with Domain Heterogeneity in the FL Context

		between training dataset and testing dataset	
		homogeneity	domain separation
among clients	homogeneity ($\lambda = 1$)	Standard FL with Domain Homogeneity	Federated DG with Domain Homogeneity
	heterogeneity $0 < \lambda < 1$	FL with Domain Heterogeneity	Federated DG with Domain Heterogeneity
	domain separation $\lambda = 0$	FL with Domain Separation	Federated DG with Domain Separation

The above overview of domain generalization and federated domain generalization shows that there is not a fair evaluation of federated domain generalization in terms of the influence of data heterogeneity, communication budget, and the number of clients. A new fair evaluation is needed to understand the behaviour of domain generalization algorithms in the federated context.

3 PROBLEM SETUP

We denote $\mathcal{D}^{\text{train}}$ as the training domains and $\mathcal{D}^{\text{test}}$ as the test domains. The model function as $h : \mathcal{X} \to \mathcal{Y}$ with parameters θ. Some algorithms formulate the model as $h = w \circ \Phi$, where $\Phi : \mathcal{X} \to \mathcal{Z}$ is the featurizer, $\mathcal{Z}$ is the feature space, and $w : \mathcal{Z} \to \mathcal{Y}$ is the classifier. We define the loss function as ℓ. We define the number of domains as M. In the FL context, we denote the number of clients as K and the number of communication rounds as C.

3.1 DOMAIN HETEROGENEITY.

Domain heterogeneity means the heterogeneity of the joint distribution P_{XY}. Current literature simplifies this notion through Bayes's rule $P_{XY} = P_{X|Y} \cdot P_Y = P_{Y|X} \cdot P_X$, and restricts to the following three types Moreno-Torres et al. (2012). *1) Prior probability shift*, where the domain heterogeneity is due to the changes in the distribution of the class variable Y, while $P_{X|Y}$ remains unaltered. It is widely studied in the recent work of FL on large heterogeneity (non-iid-ness) between clients, where it appears with the name class imbalance, see McMahan et al. (2017); Karimireddy et al. (2020). *2) Covariate shift* corresponds to the conditional probability where $P(Y|X)$ remains unchanged, but the input distribution $P(X)$ changes. *3) Concept shift* is also referred to as "functional relation change Yamazaki et al. (2007)," where $P(Y|X)$ varies from one domain to another while $P(X)$ is not altered. For instance, in a self-driving vehicle application, the model receives captured images as input X, and the output Y is the classification of whether the vehicle is on the correct side. Though X might be the same, e.g., the vehicle is in the right side lane, $P(Y|X)$ might differ depending on the traffic law of different countries. In this work, we do not restrict ourselves to the above three cases. Instead, we consider the general case: heterogeneity of the joint distribution P_{XY}.

Implementation of domain heterogeneity We recall some notations for convenience and introduce here some new notations related to splitting. Recall here that $\lambda \in [0, 1]$ represents the homogeneity level, as λ larger, the data distribution among clients are more similar; M denotes the total domain number; and K as the number of clients. For each domain m, we define n_m as the number of training examples for domain m; in addition, we denote $n_{m,k,\lambda}$ as the number of examples from domain m to client k when homogeneity level is λ.

To construct the data heterogeneity among clients, we first consider two extreme case: the homogeneous split and fully heterogeneous split. In the homogeneous split, define $\lambda = 1$, we assign training sample from each domain to every client evenly, $n_{m,k,1} = n_m/K$, for all $k = 1, \ldots, K$, and $m = 1, \ldots, M$. That is, each client contains all types of domains, and the data for each client are independent and identical distributed. In fully heterogeneous split, define $\lambda = 0$, we try to distribute the training samples from one domain to as fewest number of client as possible. That is, we aim to set the domains of each client "non-overlapping." In the case when there are more domains than clients, i.e., $K \leq M$, non-overlapping domain for each agent can be fulfilled in this scenario, thus, we have $n_{m,k,0} = n_m$, if domain m is included in client k, otherwise $n_{m,k,0} = 0$. In the case when there are more clients than domains $K > M$, we only assign one domain to each client. Even some clients may share the same domain, no agent holds more than one domain. To reduce the influence of imbalanced training domain sample size, we iteratively choose the largest training domain as the candidate for the clients to share, $n_{m,k,0}$ is related to specific domain size. For any intermediate case $\lambda \in (0, 1)$, we interpolate between homogeneous and fully heterogeneous split. The final examples for client k from each domain m with homogeneity λ is given by the convex combination: $n_{m,k,\lambda} = \lambda \cdot n_{m,k,1} + (1 - \lambda) \cdot n_{m,k,0}$. With λ approaching to 0, the data heterogeneity is higher, as well as the domains are more "non-overlapping" among clients.

3.2 FEDERATED DOMAIN GENERALIZATION

In the federated learning context, data heterogeneity can introduce unique challenges for domain generalization. In particular, local client may not have access to all the domains, and their domains may only overlap partially; in the extreme case, local data included in one client may naturally form a unique domain. Therefore, whether centralized generalization ability is achievable when local agent has heterogeneous data distribution remains unknown. Further, if some distributed algorithms can achieve the centralized generalization ability as if the server has information of all the domains,

the communication cost remains unclear. In addition, the impact on the generalization ability of the total client number remains unknown.

Specific DG Algorithms In this benchmark, we extend six representative algorithms into the FL context and compare them with FedDG Liu et al. (2021) and FedAdg Zhang et al. (2021), which are naturally designed for solving domain generalization tasks in federated learning. The six algorithms that we built on are IRM Arjovsky et al. (2019), Fish Shi et al. (2021), MMD Li et al. (2018), Coral Sun & Saenko (2016), GroupDRO Sagawa et al. (2019), Mixup Zhang et al. (2017), which are designed for solving domain generalization tasks in the centralized case. Denote E as the training passes each client makes over its local dataset on each round. An implementation of the federated domain generalization algorithm has the following form: for each communication round, each client locally runs E epochs domain generalization algorithm on the current model using its local training domain data, and then the server takes a weighted average of the resulting models, i.e., $\theta^{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \theta_k^{t+1}$, where n_k is the training sample size at agent k. See Appendix A.2 for the detailed hyperparameters choice of learning rate and batch size choice.

Note that Mixup Zhang et al. (2017), FedDG Liu et al. (2021), and FedADG Zhang et al. (2021) only work for image datasets, thus we do not include them on the Py150 dataset.

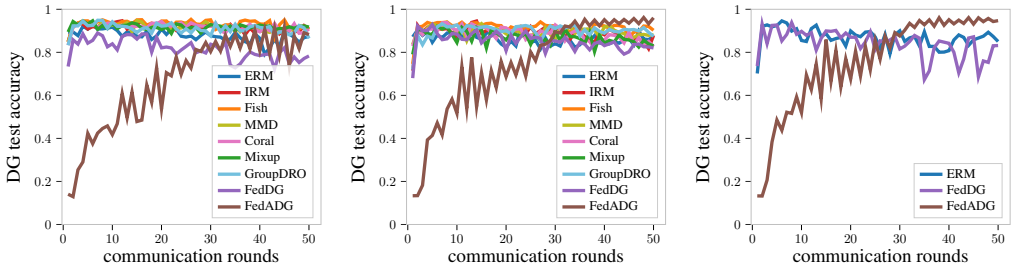
Model Selection In DG, model selection could significantly affect the model performance. During training, we evaluate the aggregated model by using the validation dataset after each communication round. After performing C communication rounds, we select the model that achieves the best performance on the validation dataset. This work adopts two model selection methods for the DG task: in-domain and held-out-domain model selection Gulrajani & Lopez-Paz (2020). In-domain model selection method uses validation dataset which is independent and identically sampled from the training dataset. The held-out-domain model selection method uses validation dataset that only contains examples from a set of domains that do not overlap with the training and testing domains.

4 Main Results

This benchmark aims to evaluate the performance of current DG algorithms in the FL context. We construct the evaluation upon three different settings according to the different level of challenges brought by the FL.

4.1 Baseline Setting $\lambda \in \{1, 0.1\}$

We start with simple dataset, and restrict to small number of clients and provide sufficient communication rounds. In addition, each client is restricted to hold overlapping domains. In particular, we evaluate PACS Li et al. (2017) dataset with $K = 2$ clients, training domain number $M_1 = M_2 = 2$, communication round is $C = 50$, and two homogeneity parameter $\lambda = (0.1, 1)$.



(a) PACS with $K = 2, \lambda = 1$ (b) PACS with $K = 2, \lambda = 0.1$ (c) PACS with $K = 2, \lambda = 0$

Figure 1: Accuracy versus communication rounds for PACS; total clients number $K = 2$; increasing heterogeneity from left panel to right panel: $\lambda = (1, 0.1, 0)$.

To observe the convergence of each algorithm, we plot Fig 1 (a) and (b) showing the test accuracy versus communication rounds C for PACS with 2 clients for homogeneous case $\lambda = 1$ and heterogeneous case $\lambda = 0.1$. As seen in Fig 1, FedADG requires more communication rounds than most

to converge to a similar accuracy. Most algorithms perform reasonably well in the homogeneous ($\lambda = 1$) and heterogeneous case ($\lambda = 0.1$) for this simple PACS dataset. Overall, as expected, existing DG algorithms with enough communication rounds are able to perform reasonably well in this simple setting where the dataset is simple and the number of clients and domains are small.

We compare the final test accuracy using two validation criterions for PACS with clients number 2. in Table 2. The table shows that the federated homogeneity case ($\lambda = 1$ column) may even slightly outperform the its counterpart centralized domain generalization accuracy in the simple case (small client and domain numbers). This could come from the natural regularization brought by the FL. As predicted, the federated heterogeneous case ($\lambda = 0.1$ column) is slightly worse than the homogeneous setting ($\lambda = 1$) for all centralized DG methods extended to the FL context, but the degradation is small because both client and domain number are small.

Table 2: Test accuracy on PACS dataset with two model selection criterion: in-domain / held-out domain validation; total client number $K = 2$. First column "centralized" corresponding to the centralized domain generalization accuracy. Increasing heterogeneity from left panel to right panel: $\lambda = (1, 0.1, 0)$. The number in boldface highlights the highest test accuracy in that column.

	In-Domain Validation				Held-Out-Domain Validation			
	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$
ERM	0.8389	0.8982	0.8234	0.8958	0.8593	0.9084	0.9144	0.9377
IRM	0.918	0.9174	0.8814	N.A.	0.9347	0.9467	**0.9389**	N.A.
Fish	0.8916	**0.9359**	0.9204	N.A.	**0.9449**	**0.9497**	0.9251	N.A.
Mixup	0.9060	0.9371	0.9072	N.A.	0.9012	0.9371	0.9072	N.A.
MMD	**0.9072**	0.9299	0.8928	N.A.	0.9210	0.9263	0.9335	N.A.
Coral	0.8886	0.9138	0.8988	N.A.	0.9204	0.9341	0.9311	N.A.
GroupDRO	0.8952	0.9018	0.8964	N.A.	0.9150	0.9491	0.9269	N.A.
FedDG	0.9024	0.8701	0.8683	0.6856	0.8922	0.8772	0.9036	**0.9383**
FedADG	0.9024	0.7958	**0.9569**	**0.9473**	0.8922	0.9126	0.9329	0.9341

4.2 Domain Separation $\lambda = 0$

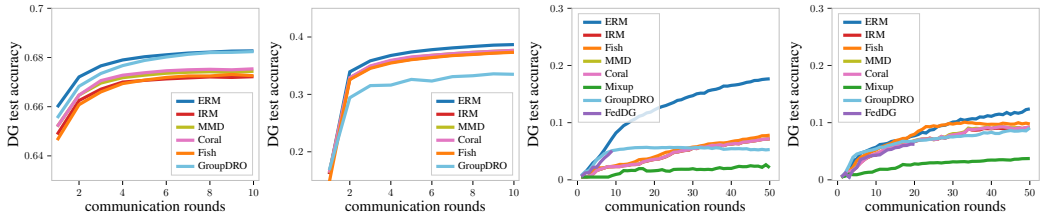
In this section, we consider domain separation case where each client only contains training examples from a unique domain, thus the data heterogeneity is extremely high. In this case, we evaluate on PACS dataset with $K = 2$ clients, training domain number $M_1 = M_2 = 1$, and communication round is $C = 50$, and homogeneity parameter $\lambda = 0$ (corresponding to per client with a unique domain).

We consider domain separation experiments where each client only has samples from one domain and the number of clients is equal to the number of domains, i.e., $K = M$. Current centralized DG methods can not directly be adapted to this FL context because they all need samples from multiple domains locally for the algorithm to apply. Therefore, we only compare the ERM, FedDG Liu et al. (2021) and FedADG Zhang et al. (2021) for this domain separation setting. Fig 1 (c) plots the test accuracy versus communication rounds C for PACS with 2 clients for fully heterogeneous case $\lambda = 0$. Table 2 column $\lambda = 0$ presents the two model selection results in this case. Table 2 (column $\lambda = 0$) show that ERM, FedADG, and FedDG achieve comparable performance if using the held-out domain selection strategy. However, FedADG seems more robust across validation strategies whereas FedDG performs poorly using in-domain validation. On the other hand as seen in Fig 1 (c), ERM and FedDG are communication efficient, while FedADG requires a large number of communication rounds to achieve comparable accuracy.

4.3 More realistic settings

In this subsection, we tackle more complicated case in the FL context. First, to explore the performance when the number of clients and domains are massive, we construct two realistic benchmark datasets: Py150 and iWildCam Koh et al. (2021), where total training domain number is 5477 for Py150 and 243 for iWildCam, and we consider total client number as $K = 100$ for Py150 and $K = 243$ for iWildCam, respectively. We further design different levels of heterogeneity between agents by setting $\lambda = (0.1, 1)$ for both datasets. Secondly, we investigate the challenges brought by

FL, including increasing data heterogeneity, increasing the number of clients and reducing communication rounds. Specifically, for each homogeneity number $\lambda = (0, 0.1, 1)$, we conduct experiments on PACS by increasing the number of clients $K = 20, 50, 100, 200$ with fixed communication round $C = 50$, and decreasing rounds of communication (with increasing local epochs) $C = (50, 10, 5)$ (with $E = (1, 5, 10)$) with fixed number of clients $K = 100$.



(a) Py150; $K = 100, \lambda = 1$ (b) Py150; $K = 100, \lambda = 0.1$ (c) IWildCam; $K = 243, \lambda = 1$ (d) IWildCam; $K = 243, \lambda = 0.1$

Figure 2: Accuracy versus communication rounds for Py150 and (first row) and IWildCam (second row); total clients number $K = 100$ for Py150 and $K = 243$ for IWildCam; increasing heterogeneity from left to right panel: $\lambda = (1, 0.1)$.

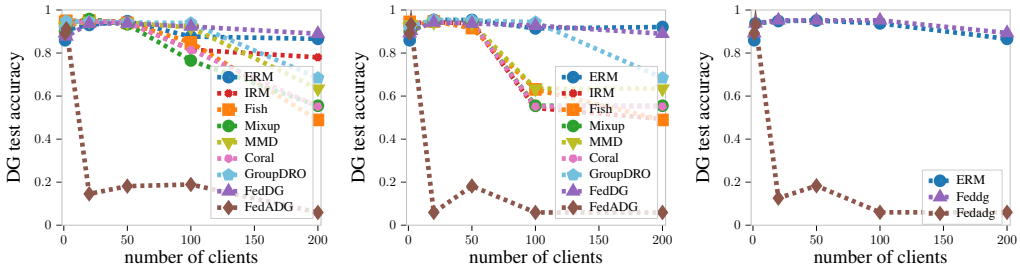
Challenges from more realistic datasets In this experiment, we consider two realistic datasets: Py150 and IWildCam. We exclude Mixup, FedDG and FedADG on Py150 because they can only be applied to computer vision tasks and are thus not applicable to the NLP dataset Py150. We summarize the results for Py150 in Fig. 2 (a), (b) and Table. 3 and IWildCam in the Fig. 2 (b), (c) and Table 4. To observe the convergence of each algorithm, we plot Fig. 2. It shows that with more realistic dataset as well as non-trivial number of clients, all of the algorithms tend to be more sensitive to client heterogeneity. We compare the final test accuracy using two validation criterion in Tables 3 and 4. The results show that the performance of ERM dominates the other algorithms on these two datasets. Additionally, though FedDG performs worse than ERM, it achieves much higher accuracy comparing to other algorithms on IWildCam dataset.

Table 3: Test accuracy on Py150 dataset with two model selection criterion: in-domain / held-out-domain validation; total client number $K = 100$.

	In-Domain Validation			Held-Out-Domain Validation		
	Centralized	$\lambda = 1$	$\lambda = 0.1$	Centralized	$\lambda = 1$	$\lambda = 0.1$
ERM	0.6743	**0.6827**	**0.3874**	0.6743	**0.6827**	**0.3874**
IRM	**0.6771**	0.6722	0.374	**0.6771**	0.6722	0.374
Fish	0.6634	0.6726	0.3734	0.6634	0.6731	0.3734
MMD	0.6495	0.6731	0.3745	0.6558	0.6731	0.3745
Coral	0.6495	0.6723	0.3755	0.6558	0.6723	0.3755
GroupDRO	0.5127	0.6824	0.3358	0.5127	0.6824	0.3358

Table 4: Test accuracy on IWildCam dataset with two model selection criterion: in-domain / held-out-domain validation; total client number $K = 243$.

	In-Domain Validation			Held-Out-Domain Validation		
	Centralized	$\lambda = 1$	$\lambda = 0.1$	Centralized	$\lambda = 1$	$\lambda = 0.1$
ERM	0.2727	**0.1762**	**0.1241**	0.2727	**0.1762**	**0.1241**
IRM	0.2845	0.0710	0.0903	0.2629	0.0718	0.0903
Fish	0.3103	0.0781	0.0977	0.3103	0.0781	0.0977
Mixup	0.2620	0.0264	0.0371	0.2923	0.0264	0.0370
MMD	**0.3225**	0.0711	0.0924	0.3134	0.0718	0.0924
Coral	0.2857	0.0718	0.0926	**0.3270**	0.0718	0.0926
GroupDRO	0.2531	0.0524	0.0908	0.2531	0.0530	0.0908
FedDG	0.2866	0.1445	0.1023	0.2771	0.1445	0.1191
FedADG	0.0049	0.0049	0.0049	0.0049	0.0049	0.0049



(a) PACS with $\lambda = 1, C = 50$ (b) PACS with $\lambda = 0.1, C = 50$ (c) PACS with $\lambda = 0, C = 50$

Figure 3: Held-out DG test accuracy versus number of clients for different level data heterogeneity. Held-out DG test accuracy is the test accuracy when we use held-out validation to select the model.

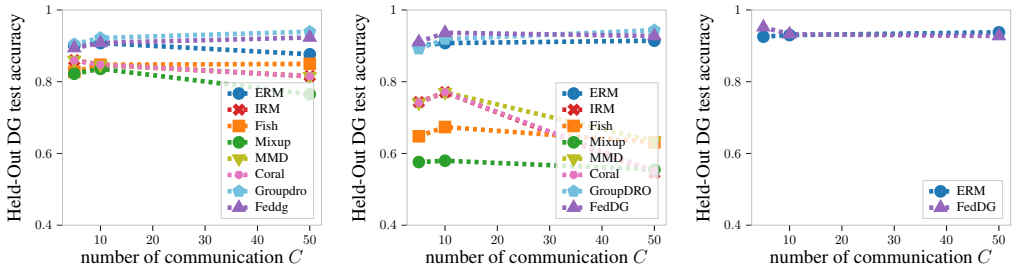
Challenges from increasing the number of clients: In this experiment, we explore the performance of different algorithms when the number of clients K increases using the simpler PACS dataset. We fix the communication rounds $C = 50$ and the local number of epoch $E = 1$ (synchronizing the models every epoch). Fig 3 plots the held-out DG test accuracy versus number of clients for different levels of data heterogeneity. The following comments are in order: given communication budget, 1) current domain generalization methods all degrade a lot in particular after $K \geq 50$, while the performance ERM and FedDG maintain relatively unchanged as the clients number increases given communication budget. FedADG performs poorly when the client number is large; 2) even in the simplest homogeneous setting $\lambda = 1$, where each local client has i.i.d training data, current domain generalization methods IRM, FISH, Mixup, MMD, Coral, GroupDRO work poorly in the existence of large clients number, this means new methods are needed for domain generalization in FL context.

Challenges from communication constraint: To show the effect of communication rounds on convergence, we plot the test accuracy versus communication rounds in Fig. 4. We fix the number of clients $K = 100$ on PACS and decreases rounds of communication (with increasing local epochs) $C = (50, 10, 5)$ (with $E = (1, 5, 10)$). For different number of communication rounds, note that we change the local training epochs E when communication budget C changes, that is, if one can have more communication rounds, then we decrease its local computation E to have the same the total computations, therefore, the comparison on the dependence of communication is fair between algorithms because the total data pass is fixed. FedADG performs poorly when $K = 100$ thus we do not include it in the comparison. 1) The curves are relatively "flat" for most of algorithms, this is predictable because we vary the number of pass of the data E per round for a changing C, and locally the aggregation rules are the same $\theta^{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \theta_k^{t+1}$, where n_k is the training sample size at agent k. 2) In particular, ERM, GroupDRO, and FedDG can achieve comparable good performance when communication budget is low ($C = 10$) as when there communication budget is high ($C = 50$), showing their communication efficient in the FL context.

Challenges from statistical heterogeneity: In this section, we investigate the influence of data heterogeneity on the generalization accuracy. Fig. 3 shows that for a given communication budget, as the data heterogeneity increases (λ changes from 1, 0.1, and 0), the test accuracy trending for ERM, GroupDRO, and FedDG are comparably good (except GroupDRO is worse than the other two when K is large), and remains roughly the same for different level of data heterogeneity, while the others degrade a lot. Fig. 4 shows that for a given number of clients, as the data heterogeneity increases (λ changes from 1, 0.1, and 0), the test accuracy trending for ERM, GroupDRO, and FedDG are comparably good (notice that GroupDRO is not applicable given that each local agent only has access to one domain in the case of $K > M$), and remains roughly the same for different level of data heterogeneity, while the others decrease a lot. These two observations suggest that ERM, GroupDRO, and FedDG may be able to deal with the data heterogeneity in the FL context.

5 Conclusion and Discussion

This work evaluates multiple algorithms to solve domain generalization task in the Federated Learning context. We evaluate influence of the data heterogeneity induced by different domains among



(a) PACS with $K = 100$, $\lambda = 1$ (b) PACS with $K = 100$, $\lambda = 0.1$ (c) PACS with $K = 100$, $\lambda = 0$

Figure 4: held-out DG test accuracy versus communication rounds.

clients, the total number clients, and communication rounds on the domain generalization ability. We show that domain generalization in the federated learning context is an unsolved problem, and it brings abundant new challenges. Specifically, the following aspect might be future directions of interest: 1) even in the homogeneous setting, current algorithms in the centralized DG extended to the FL context even fail to tackle the large client setting $K \geq 50$ for simple domain dataset PACS. 2) Local data statistical heterogeneity across clients brings extra challenge for domain generalization in the FL context. 3) For the domain separation case, FedDG and FedADG provide some initial attempts. However, FedDG requires data sharing and FedAdg requires heavy hyper-parameter tuning. Besides, both of these they do not work with NLP datasets. Other DG methods proposed in the centralized learning fail to apply in the domain separation setting because they need multiple domain data locally to perform. We list the gap table below for summarizing the current DG algorithms performance gap w.r.t ERM in the FL context, in particular, positive means outperform ERM, negative means worse than ERM. It can be seen that in the on the simple dataset, the best DG migrated from centralized setting is better than ERM. In the domain separation case, no centralized DG algorithms can be adapted to it, and FedDG and FedADG performs comparably good in this setting. However, they fail in harder datasets. Federated DG algorithms that outperforming ERM, supporting NLP dataset, and free of data sharing are still in need.

Table 5: Gap Table: Current Progress in solving DG in FL context

	Comparing to FedAvg-ERM		
	DG migrated from centralized setting	FedDG	FedADG
Baseline Dataset: PACS	+4.13%	-3.12%	+0.42%
Domain Separation	✗	-1.08%	+0.06%
Realistic Dataset: Py150	-1.05%	✗	✗
Realistic Dataset: IWildCam	-10.44%	-3.17%	-17.13%
Free of Data Sharing	✓	✗	✓

6 Reproducibility Statement

Code for reproduce the result is available at the anonymous link. We include detailed documentation in using our code to reproduce the results throughout the paper. We also provide documentation in adding new algorithm's DG evaluation in the FL context.

REFERENCES

Durmus Alp Emre Acar, Yue Zhao, Ruizhao Zhu, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Debiasing model updates for improving personalized federated training. In *International Conference on Machine Learning*, pp. 21–31. PMLR, 2021.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24, 2011.

Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*, 2018.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pp. 1180–1189. PMLR, 2015.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. A kernel method for the two-sample-problem. *Advances in neural information processing systems*, 19, 2006.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtárik. Lower bounds and optimal algorithms for personalized federated learning. *Advances in Neural Information Processing Systems*, 33:2304–2315, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Christina Heinze-Deml, Jonas Peters, and Nicolai Meinshausen. Invariant causal prediction for nonlinear models. *Journal of Causal Inference*, 6(2), 2018.

Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, pp. 4387–4398. PMLR, 2020.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 5542–5550, 2017.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5400–5409, 2018.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

Quande Liu, Cheng Chen, Jing Qin, Qi Dou, and Pheng-Ann Heng. Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1013–1023, 2021.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pp. 97–105. PMLR, 2015.

David Lopez-Paz, Robert Nishihara, Soumith Chintala, Bernhard Scholkopf, and Léon Bottou. Discovering causal signals in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6979–6987, 2017.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Jose G Moreno-Torres, Troy Raeder, Rocío Alaiz-Rodríguez, Nitesh V Chawla, and Francisco Herrera. A unifying view on dataset shift in classification. *Pattern recognition*, 45(1):521–530, 2012.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pp. 10–18. PMLR, 2013.

Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):947–1012, 2016.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Towards causal representation learning, 2021. URL `https://arxiv.org/abs/2102.11107`.

Yuge Shi, Jeffrey Seely, Philip HS Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. *arXiv preprint arXiv:2104.09937*, 2021.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pp. 443–450. Springer, 2016.

Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33:21394–21405, 2020.

Keisuke Yamazaki, Motoaki Kawanabe, Sumio Watanabe, Masashi Sugiyama, and Klaus-Robert Müller. Asymptotic bayesian generalization error when training and test distributions are different. In *Proceedings of the 24th international conference on Machine learning*, pp. 1079–1086, 2007.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Liling Zhang, Xinyu Lei, Yichun Shi, Hongyu Huang, and Chao Chen. Federated learning with domain generalization. *arXiv preprint arXiv:2111.10487*, 2021.

A EXPERIMENTS SETTING SUPPLEMENTARY

A.1 DATASETS AND MODELS

In this section, we introduce the datasets we used in our experiments, and the split method we used to build heterogeneous datasets in the training and testing phase as well as the heterogeneous local training datasets among clients in the FL. **Datasets.** We contain three datasets as the benchmark: PACS Li et al. (2017), IWildCam Koh et al. (2021), and Py150 Koh et al. (2021). These three datasets cover different levels of difficulty as well as different types of tasks. PACS and IWildCam are both image classification datasets and Py150 is a natural language processing (NLP) dataset. PACS is an image dataset for domain generalization. It consists of four domains, namely Photo (1,670 images), Art Painting (2,048 images), Cartoon (2,344 images), and Sketch (3,929 images). This task requires learning the classification task on a set of objects by learning on totally different renditions. The Py150 is a natural language processing dataset containing $150,000$ python source code dataset from $8,421$ repository. The goal is to predict the next token given the context of previous tokens. This is a real-world NLP dataset that contains multiple repositories which naturally form multiple domains. The IWildCam contains wild animals captured by multiple heats or motion-activated static cameras. Due to the variation in camera model, position, color, background, and relative animal frequencies, the samples form multiple domains. It contains $203,029$ images from 323 different camera traps, the images contain 182 different animal species. For Py150 and IWildcam datasets, we follow the same split method as the Wilds Koh et al. (2021). For PACS, we use cartoon and sketch as the training domains, art-painting as the held-out-validation domain, and photo as the test domain, and we use 90% of the data from cartoon and sketch domains to be the training domains, and about 5% to be in-domain validation, and other 5% to be the in-domain test set. During the sampling, we keep the class distribution the same among the training dataset, the in-domain validation dataset, and the in-domain test dataset. **Models.** For image classification datasets PACS and IWildCam, we use ResNet50 model He et al. (2016), and Py150 is a natural language processing (NLP) dataset where we use OpenAI GPT2 Radford et al. (2019) to train.

A.2 HYPERPARAMETERS

To make fair comparisons, we allocate the same budget during training for each algorithm on each dataset. The budget includes the times of allowed hyperparameter search, model architecture, local computation resources, and communication rounds. For each dataset, we fix the model architecture and initialization to be the same. We conduct eight times hyperparameter searching for each algorithm, and choose the set of hyperparameters that achieves the best performance. For all the datasets and algorithms, we set 100% for clients' participation in the training during each communication. For PACS, we fix the number of clients $K = 2$, 15 communications in total where each communication happens after one epoch of local training. the batch size is 64. We use Adam optimizer and the learning rate is 1×10^{-3}. For Py150, we set $K = 100$, and the total number of communication is 3 in the centralized case and 10 in the distributed case. The batch size is 96. We use AdamW optimizer, lr $= 8 \times 10^{-5}$, and $\epsilon = 1 \times 10^{-8}$. For IWildcam, the number of clients $K = 243$, 12 communications in the centralized case, and 50 communications in the federated learning case, and the batch size is 16. We use Adam optimizer where the learning rate is 3×10^{-5}.

For each dataset, we choose the hyperparameters starting from their default value consistent with the choice in previous benchmarks DomainBed Gulrajani & Lopez-Paz (2020) and Wilds Koh et al. (2021).

B OTHER EVALUATION

We put our extra evaluation here for reference.

Table 6: Domain Generalization tasks on PACS dataset. All federated learning frameworks contain 20 local clients.

	In-Domain Validation				Held-Out-Domain Validation			
	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$
ERM	0.8389	0.9293	0.9455	**0.9569**	0.8389	0.9305	**0.9539**	0.9485
IRM	0.9180	0.9461	0.9275	N.A.	0.9180	0.9479	0.9431	N.A.
Fish	0.9030	0.9449	0.9443	N.A.	0.9030	0.9515	0.9467	N.A.
Mixup	0.8635	**0.9581**	0.9431	N.A.	0.8635	**0.9581**	0.9461	N.A.
MMD	0.9186	0.9461	0.9389	N.A.	0.9186	0.9491	0.9431	N.A.
Coral	**0.9216**	0.9449	0.9461	N.A.	**0.9216**	0.9491	0.9455	N.A.
GroupDRO	0.9060	0.9401	**0.9521**	N.A.	0.9060	0.9461	**0.9539**	N.A.
FedDG	0.9024	0.9275	0.9275	0.8952	0.8922	0.9365	0.9425	**0.9521**
FedADG	0.9024	0.1347	0.1269	0.1257	0.8922	0.1455	0.1269	0.1257

Table 7: Domain Generalization tasks on PACS dataset. All federated learning frameworks contain 50 local clients.

	In-Domain Validation				Held-Out-Domain Validation			
	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$
ERM	0.8389	0.9371	0.9437	**0.9533**	0.8389	**0.9479**	**0.9539**	0.9509
IRM	0.9180	0.9371	0.9401	N.A.	0.9180	0.9455	0.9389	N.A.
Fish	0.9030	0.9407	0.9030	N.A.	0.9030	0.9401	0.9162	N.A.
Mixup	0.8635	**0.9503**	0.9407	N.A.	0.8635	0.9335	0.9419	N.A.
MMD	0.9186	0.9389	0.9329	N.A.	0.9186	0.9413	0.9389	N.A.
Coral	**0.9216**	0.9389	0.9228	N.A.	**0.9216**	0.9419	0.9413	N.A.
GroupDRO	0.9060	0.9443	**0.9515**	N.A.	0.9060	0.9419	0.9515	N.A.
FedDG	0.9024	0.9329	0.9287	0.9515	0.8922	0.9377	0.9371	**0.9515**
FedADG	0.9024	0.1862	0.1862	0.1862	0.8922	0.1814	0.1820	0.1832

Table 8: Domain Generalization tasks on PACS dataset. All federated learning frameworks contain 100 local clients.

	In-Domain Validation				Held-Out-Domain Validation			
	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$
ERM	0.8389	0.8766	0.9144	**0.8958**	0.8389	0.8766	0.9144	0.9377
IRM	0.9180	0.8156	0.5449	N.A.	0.9180	0.8156	0.5449	N.A.
Fish	0.9030	0.8497	0.6311	N.A.	0.9030	0.8497	0.6311	N.A.
Mixup	0.8635	0.7653	0.5551	N.A.	0.8635	0.7653	0.5551	N.A.
MMD	0.9186	0.8150	0.6341	N.A.	0.9186	0.8150	0.6341	N.A.
Coral	**0.9216**	0.8150	0.5515	N.A.	**0.9216**	0.8150	0.5515	N.A.
GroupDRO	0.9060	**0.9395**	**0.9437**	N.A.	0.9060	**0.9395**	**0.9437**	N.A.
FedDG	0.9024	0.8868	0.8880	0.8952	0.8922	0.9234	0.9275	**0.9521**
FedADG	0.9024	0.0915	0.0732	0.0592	0.8922	0.1892	0.0592	0.0598

Table 9: Domain Generalization tasks on PACS dataset. All federated learning frameworks contain 200 local clients.

	In-Domain Validation				Held-Out-Domain Validation			
	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$
ERM	0.8389	**0.8635**	0.8287	0.8539	0.8389	0.8287	**0.9210**	0.8659
IRM	0.9180	0.6790	0.5000	N.A.	0.9180	0.7802	0.4934	N.A.
Fish	0.9030	0.7802	0.4850	N.A.	0.9030	0.7802	0.4892	N.A.
Mixup	0.8635	0.6701	0.5551	N.A.	0.8635	0.6671	0.5551	N.A.
MMD	0.9186	0.8150	0.6341	N.A.	0.9186	0.8150	0.6341	N.A.
Coral	**0.9216**	0.8150	0.5018	N.A.	**0.9216**	0.8150	0.4916	N.A.
GroupDRO	0.9060	0.8144	0.6090	N.A.	0.9060	0.8162	0.6844	N.A.
FedDG	0.9024	0.7916	**0.8898**	**0.8868**	0.9204	**0.9204**	0.8898	**0.8934**
FedADG	0.9024	0.0915	0.0732	0.0592	0.8922	0.1892	0.0592	0.0598

Table 10: Domain Generalization tasks on PACS dataset. All federated learning frameworks contain 100 local clients and communication rounds $C = 5$.

	In-Domain Validation				hH-Out-Domain Validation			
	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$
ERM	0.8389	0.9006	**0.9132**	**0.9257**	0.8389	0.9006	**0.9132**	**0.9257**
IRM	0.9180	0.8599	0.7431	N.A.	0.9180	0.8599	0.7431	N.A.
Fish	0.9030	0.8275	0.6479	N.A.	0.9030	0.8275	0.6479	N.A.
Mixup	0.8635	0.8222	0.5760	N.A.	0.8635	0.8222	0.5760	N.A.
MMD	0.9186	0.8605	0.7413	N.A.	0.9186	0.8605	0.7413	N.A.
Coral	**0.9216**	0.8593	0.7413	N.A.	**0.9216**	0.8593	0.7413	N.A.
GroupDRO	0.9060	**0.9048**	0.8916	N.A.	0.9060	**0.9048**	0.8916	N.A.
FedDG	0.9024	0.8940	0.9108	0.8952	0.8922	0.8940	0.9108	**0.9521**
FedADG	0.9024	0.0915	0.0732	0.0592	0.8922	0.1892	0.0592	0.0598

Table 11: Domain Generalization tasks on PACS dataset. All federated learning frameworks contain 100 local clients and communication rounds $C = 10$.

	In-Domain Validation				Held-Out-Domain Validation			
	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$	Centralized	$\lambda = 1$	$\lambda = 0.1$	$\lambda = 0$
ERM	0.8389	**0.9186**	0.8587	**0.8958**	0.8389	0.9078	0.8970	0.9377
IRM	0.9180	0.8347	0.7707	N.A.	0.9180	0.8473	0.7707	N.A.
Fish	0.9030	0.8473	0.6737	N.A.	0.9030	0.8473	0.6737	N.A.
Mixup	0.8635	0.8359	0.5796	N.A.	0.8635	0.8359	0.5796	N.A.
MMD	0.9186	0.8341	0.7707	N.A.	0.9186	0.8473	0.7707	N.A.
Coral	**0.9216**	0.8473	0.7707	N.A.	**0.9216**	0.8473	0.7707	N.A.
GroupDRO	0.9060	0.9222	0.9186	N.A.	0.9060	**0.9222**	0.9186	N.A.
FedDG	0.9024	0.9084	**0.8880**	0.8952	0.8922	0.9084	**0.9275**	**0.9521**
FedADG	0.9024	0.0915	0.0732	0.0592	0.8922	0.1892	0.0592	0.0598