

Towards Instance-adaptive Inference for Federated Learning

Chun-Mei Feng¹ Kai Yu^{1*} Nian Liu² Xinxing Xu^{1*} Salman Khan^{2,3} Wangmeng Zuo⁴

¹Institute of High Performance Computing (IHPC),

Agency for Science, Technology and Research (A*STAR), Singapore

²Mohamed bin Zayed University of Artificial Intelligence (MBZUAI), UAE

³Australian National University, Canberra ACT, Australia

⁴Harbin Institute of Technology, Harbin, China

fengcm.ai@gmail.com; yu_kai@ihpc.a-star.edu.sg; xuxinx@ihpc.a-star.edu.sg

<https://github.com/chunmeifeng/FedIns>

Abstract

Federated learning (FL) is a distributed learning paradigm that enables multiple clients to learn a powerful global model by aggregating local training. However, the performance of the global model is often hampered by non-i.i.d. distribution among the clients, requiring extensive efforts to mitigate inter-client data heterogeneity. Going beyond inter-client data heterogeneity, we note that intra-client heterogeneity can also be observed on complex real-world data and seriously deteriorate FL performance. In this paper, we present a novel FL algorithm, i.e., FedIns, to handle intra-client data heterogeneity by enabling instance-adaptive inference in the FL framework. Instead of huge instance-adaptive models, we resort to a parameter-efficient fine-tuning method, i.e., scale and shift deep features (SSF), upon a pre-trained model. Specifically, we first train an SSF pool for each client, and aggregate these SSF pools on the server side, thus still maintaining a low communication cost. To enable instance-adaptive inference, for a given instance, we dynamically find the best-matched SSF subsets from the pool and aggregate them to generate an adaptive SSF specified for the instance, thereby reducing the intra-client as well as the inter-client heterogeneity. Extensive experiments show that our FedIns outperforms state-of-the-art FL algorithms, e.g., a 6.64% improvement against the top-performing method with less than 15% communication cost on Tiny-ImageNet.

1. Introduction

The availability of large-scale data dramatically promotes the development of deep learning models. However, as these abundant data tend to be distributed across

*Corresponding author.

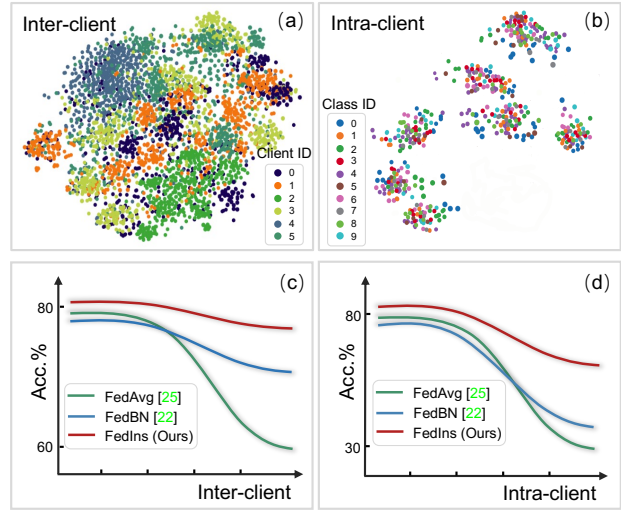


Figure 1: **Illustration of inter- and intra-client data heterogeneity** with t-SNE visualizations (see (a) and (b)) on DomainNet and their effect on accuracy of different FL algorithms (see (c) and (d)). Please refer to the *Suppl.* for details on the settings to increase data heterogeneity. One can see that both inter- and intra-client data heterogeneity degrades FL performance. While FedBN [27] is able to alleviate the effect of inter-client data heterogeneity, both FedAvg [30] and FedBN [27] are limited in handling intra-client data heterogeneity.

many devices due to logistical and privacy concerns, decentralized training is often required to train the deep neural network [1]. As a promising distributed learning paradigm, federated learning (FL) can train global models in a distributed manner without sharing local private data [25, 30, 29]. During this process, each client first trains a local model on their private data and then sends the model parameters to the server for aggregation and distribution back to the client [30].

However, each client collects the local data in its own

manner (*e.g.*, different devices, different ways). Consequently, the data distribution among different clients is heterogeneous (see Fig. 1 (a)), resulting in serious performance degradation of conventional algorithms [4] (*e.g.*, FedAvg [30]). Many FL algorithms have been suggested to solve the problem of inter-client data heterogeneity for making them more robust in the non-i.i.d. setting [25, 16, 27, 24, 9] (see Fig. 1 (c)).

Actually, even for an individual client, the data may be collected by different devices [7], under different environmental conditions, *etc.* Therefore, *intra-client data heterogeneity* can also be observed on complex real-world data [3] (see Fig. 1 (b)), and give rise to serious degradation of FL performance (see Fig. 1 (d)). A naive solution is to build a specific model for each instance [42]. However, such a scheme results in significant challenges in the training, *storage and communication* of the instance-adaptive models. Instead, we resort to the parameter-efficient fine-tuning, *e.g.*, prompt tuning [14], of pre-trained models. As for prompt tuning, we can simply freeze the backbone of pretrained models, only learn and communicate a small number of learnable prompts between the server and the client [6, 36, 35, 14]. Nonetheless, instance-wise modeling remains an *unstudied* issue for parameter-efficient fine-tuning. Additionally, in comparison to the original pre-trained model, prompt tuning also introduces *additional* parameters and *increases* the computation cost in the inference stage [28].

To handle the intra-client data heterogeneity, instead of instance-adaptive models, this paper resorts to **instance-adaptive inference**, and presents a novel FL algorithm, *i.e.*, FedIns, built upon pre-trained models. For parameter-effective fine-tuning, following [28], we scale and shift the deep features (SSF) to fine-tune the pre-trained model in the local training phase and merge them into the original pre-trained model weights by reparameterization in the inference phase. To enable instance-adaptive inference, we train a SSF pool for each client and aggregate them on the server, and thus *low storage and communication costs* can still be maintained. For example, compared with FedAvg [30], FedIns only has less than 15% communication costs. Our federated SSF pool is aggregated from multiple groups of local SSF, which implicitly add client-relevant knowledge to the **federated SSF pool** and dynamically guide the client to handle the corresponding response in an instance-wise fashion. During inference, for a given instance, we dynamically find the best-matched SSF subsets from the pool and aggregate them to generate an adaptive SSF specified for the instance. As such, the model can reduce both inter- and intra-client heterogeneity, achieving a 6.64% gain against FedAvg on CIFAR-100.

In a nutshell, our contributions are three-fold:

- A novel FL algorithm, *i.e.*, FedIns, is presented to han-

dle *intra-client data heterogeneity*, which has been overlooked by the existing FL literature.

- Federated SSF is proposed and extended by allowing each client to have an SSF pool, and instance-adaptive inference is fulfilled by dynamically finding and aggregating the best matched SSF subsets for each test instance.
- Extensive experimental results show FL performance can be effectively improved by alleviating the intra- and inter-client data heterogeneity.

2. Related Work

Data Heterogeneity in FL. Generally, federated learning algorithm aims to obtain an aggregated model that minimizes training losses for all clients. The classic FL algorithm, FedAvg [30] simply sends the local model to the server for aggregation to learn a global model. However, as each device generates its own local data, data heterogeneity across different clients occurs, making FedAvg becomes a sub-optimal solution for FL [26, 40, 20, 8].

Existing FL methods usually alleviate this problem in two aspects. One is to improve local training. For example, FedProx adds a proximal term to the objective function of the local model to tackle heterogeneity [25]. Karimireddy *et al.* used control variates to correct the client-drift caused by data heterogeneity [16]. Li *et al.* kept all the batch normalization in the local to alleviate the heterogeneity of local data across clients [27]. Inspired by contrastive learning, Li *et al.* [24] corrected the local clients by computing similarity between model representations to handle the heterogeneity issue. Gao *et al.* used an auxiliary local drift variable to bridge the gap between the local and the global model parameters, thereby alleviating the data heterogeneity [9]. Mendieta *et al.* alleviated data heterogeneity by promoting local learning generality rather than proximal restriction [31]. The non-i.i.d. problem caused by data heterogeneity can also be alleviated by addressing catastrophic forgetting from the server to the client, where each local and global communication is regarded as a new task [39, 19].

Besides improving local training, another is to improve the server aggregation process for alleviating data heterogeneity. For example, Yurochkin *et al.* replaced classical aggregation schemes by matching neuron aggregation in local models based on a Bayesian non-parametric approach [41]. Analogously, Wang *et al.* created a normalized averaging method as an alternative to the average update mechanism [38]. To sum up, to handle the data heterogeneity across clients, existing methods either limit the impact of local updates on the server (*e.g.*, by regularizing and personalizing the design of clients to correct the update direction of locals [25, 16, 27, 24, 9, 31]), or modify the aggregation scheme [41, 38].

However, in many real-world applications, local data in each client may be collected by different devices and from

different environments. Consequently, there may be *multiple mixed unknown sub-domains within one client* [3], i.e., intra-client data heterogeneity. As discussed in Sec. 1, both intra-client and inter-client data heterogeneity will lead to the performance degradation of FL. Nonetheless, existing methods mainly focus on inter-client data heterogeneity, leaving *intra-client heterogeneity remain an uninvestigated issue*. Given this, this work presents a novel FedIns algorithm to handle intra-client data heterogeneity, which can also be readily deployed to alleviate inter-client data heterogeneity.

Parameter-efficient Fine-tuning. In the recent few years, we have witnessed the promising performance of pre-training models in many downstream tasks. Accordingly, in the field of FL [32, 2], pre-trained models can also serve as a strong baseline. However, the number of parameters of pre-trained models is usually very large [37, 11], and simply fine-tuning the full model undoubtedly yields a huge amount of communication cost in FL algorithms. Therefore, some works have attempted to explore large-scale pre-training models in a parameter-efficient fine-tuning manner [17]. In this way, the backbone network is frozen during training, and only a small number of parameters can be learned or tuned to “understand” downstream tasks. In particular, prompt [21, 14, 22], adapter [13, 12, 33], and SSF [28] have been suggested to leverage the representation abilities of large-scale pre-training models to achieve good performance on downstream tasks by fine-tuning a few trainable parameters.

Among these parameter-efficient fine-tuning methods, most prompt and adapter-based methods [13, 15] introduce additional parameters and computational costs in the inference stage. In addition, prompt tuning is *sensitive* to the number of prompts; e.g., different tasks need to adopt different numbers of prompts, and an inappropriate number will reduce the accuracy or increase the redundancy of the calculation. In contrast, SSF [28] does not bring extra parameters and FLOPs during the inference phase since it only adds learnable parameters during the training phase and merges them into the original pre-trained model weights via reparameterization after training. Despite these progress, it remains not studied for the effectiveness of SSF in a distributed framework with intra- and inter-client data heterogeneity. Therefore, different from the above centralized works, in this paper we focus on exploring how to *make SSF work for decentralized FL frameworks* to effectively handle data heterogeneity.

3. Methodology

3.1. Federated Learning

As growing privacy concerns arise, federated learning has received intensive recent interest by training and de-

ploying deep neural network models in a distributed manner [30]. Suppose there are K local clients, where each of them has their local dataset \mathcal{D}^k . The distributed paradigm FL aims to learn a global model \mathbf{w} over the whole training data $\mathcal{D} = \{\mathcal{D}^1, \mathcal{D}^2, \dots, \mathcal{D}^K\}$ using a central server without exchanging local private data. Formally, such a process can be expressed as

$$\arg \min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) = \sum_{k=1}^K \frac{|\mathcal{D}^k|}{|\mathcal{D}|} \mathcal{L}_k(\mathbf{w}), \quad (1)$$

where $|\mathcal{D}|$ denotes the number of samples in \mathcal{D} , and \mathbf{w} denotes the model parameters. $\mathcal{L}_k(\mathbf{w})$ is the empirical loss of client k which can be expressed as

$$\mathcal{L}_k(\mathbf{w}) = \mathbb{E}_{(\mathbf{x}, \mathbf{y}) \in \mathcal{D}^k} \ell_k(\mathbf{x}; \mathbf{w}), \quad (2)$$

where ℓ_k denotes the local loss term, e.g., cross-entropy loss, and \mathbf{x} denote a sample of client k .

3.2. Learning Federated SSF

In federated learning, the local data \mathcal{D}^k of a client may not be sufficient to train a large scale deep network. Pre-trained models can thus be introduced to compensate for the deficiency of local data [32, 2]. However, pre-trained model usually has a large number of model parameters. Direct fine-tuning the full model consequently gives rise to significant communication costs between the server and the client. In this work, we adopt a recent parameter-efficient fine-tuning paradigm, i.e., SSF [28], which trains only a small number of learnable parameters and brings no inference overhead by reparameterizing them into the original pre-trained model weights. In the following, we will introduce federated SSF, which incorporates SSF into the FL framework.

In SSF, given a pre-trained model with parameters θ , we remodulate features by insert SSF with the scale γ and shift β factors after each operation (OP) [28], i.e., multi-head self-attention (MSA), MLP and layer normalization (LN), etc. During the fine-tuning phase, the model parameters of SSF can be represented as $\mathbf{w} = \{\gamma, \beta, \mathbf{h}, \theta\}$, where \mathbf{h} is the parameters of the classification head. In particular, the pre-trained weights are kept frozen, and only the SSF and classification head are kept updated. Once the training is accomplished, $\{\gamma, \beta, \mathbf{h}\}$ can then be merged into θ to obtain the updated model parameter θ' .

As shown in Fig. 2 (b), when incorporating SSF into FL, we only require to update the client-specific SSF and classification head, i.e., $\delta = \{\gamma, \beta, \mathbf{h}\}$. Thus, Eq. (1) can be modified as,

$$\delta_g = \arg \min_{\delta} \mathcal{L}(\delta) = \sum_{k=1}^K \frac{|\mathcal{D}^k|}{|\mathcal{D}|} \mathcal{L}_k(\delta). \quad (3)$$

Federated SSF only requires a small number of parameters δ_k of local clients to be updated and commu-

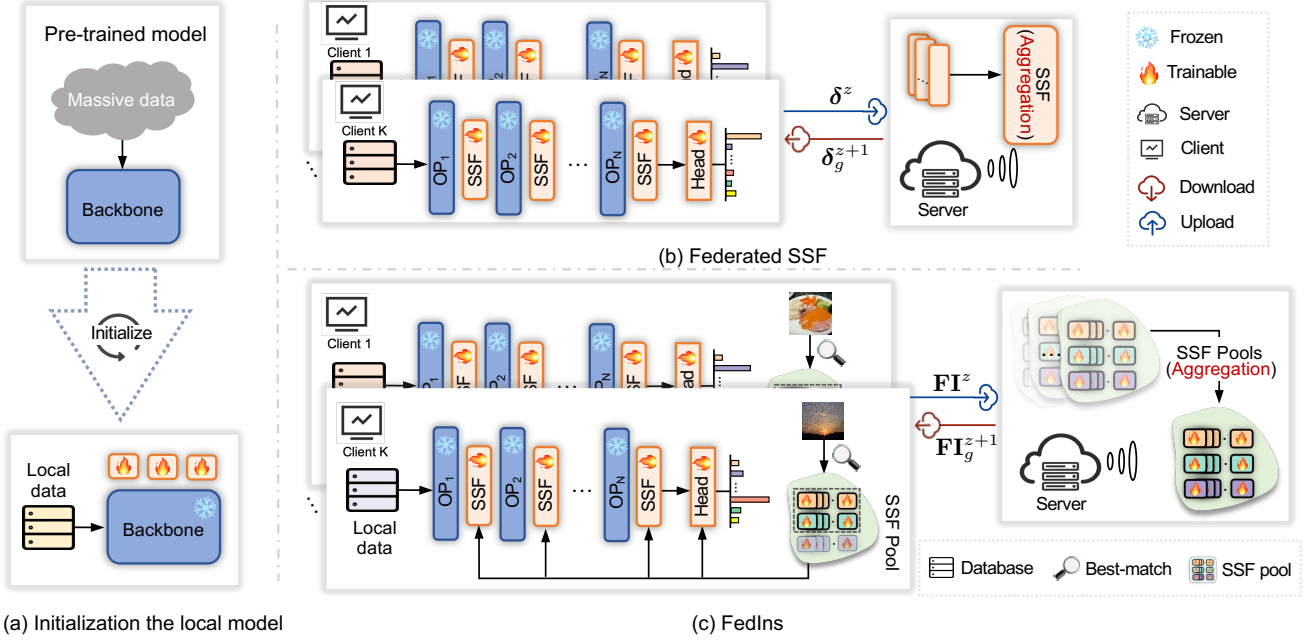


Figure 2: **Overall pipeline** of our proposed **FedIns**. (a) Pre-train the backbone on the massive data *offline* and fine-tune a small number of learnable parameters on local data in a parameter-efficient manner. (b) For federated SSF, we train an SSF for each client and aggregate the SSFs from all clients to obtain the global SSF (see Sec. 3.2). (c) As for FedIns, we train a federated SSF pool for *each client* and aggregate the SSF pools from all clients to obtain the global SSF pool (see Sec. 3.3).

icated with the server, and thus is both parameter- and communication-efficient for FL.

Local Update Step: Assume there are Z rounds of communication with T local updates per round. The clients are optimized using the following update rules with a learning rate of η_k for each communication round $z = \{1, 2, \dots, Z\}$:

$$\delta_k^{z,t+1} \leftarrow \delta_k^{z,t} - \eta_k \nabla \ell_k(\mathbf{x}^k; \delta_k^{z,t}), \quad (4)$$

where t denotes the t -th update of the local clients.

Server Update Step: The server performs aggregation every round by receiving the updated parameters of all participated clients after the local updates within each round. Formally, we have

$$\delta_g^{z+1} \leftarrow \sum_{k=1}^K \frac{|\mathcal{D}^k|}{|\mathcal{D}|} \delta_k^z, \quad (5)$$

where δ_g^{z+1} denotes the global updated parameters of round $z + 1$. Then, we can obtain a robust global model parameterized by δ_g after Z rounds of communication without disclosing any local private data.

When the training is accomplished, we can reparameterize the SSF by merging it into the original parameter space (*i.e.*, model weight θ). As a result, federated SSF is not only efficient in terms of communication costs, but also does not introduce any extra parameters during the inference phase.

3.3. FedIns

In this subsection, we further present FedIns for handling intra-client data heterogeneity. In many complex real-world scenarios, the data in a local client may contain multiple unknown mixed sub-domains [3]. Despite its merit on communication cost, federated SSF is still limited in alleviating the intra-client data heterogeneity issue. To address this issue, we extend federated SSF to SSF pool for enabling instance-adaptive inference in the FL framework, resulting in our FedIns algorithm. As shown in Fig. 2 (c), we train an SSF pool for each client, and aggregate them into the federated SSF pool on the server. For a given instance, we dynamically find the best-matched SSF subsets from the pool and aggregate them to generate an adaptive SSF for instance-adaptive inference. In the following, we will introduce FedIns in more detail.

To extend federated SSF, we allow each client have an SSF pool $\Delta_k = \{\delta_k^1, \delta_k^2, \dots, \delta_k^M\}$ of a set of δ_k^m s, where M denotes the size of pool. Furthermore, we introduce a pool of learnable Keys $\mathbf{K}_k = \{\mathbf{k}_k^1, \mathbf{k}_k^2, \dots, \mathbf{k}_k^M\}$ corresponding to Δ_k . Taking both Δ_k and \mathbf{K}_k into account, the learnable parameters of FedIns of each client can be expressed as,

$$\mathbf{FI}_k = \{\mathbf{K}_k, \Delta_k\}. \quad (6)$$

Then, the local update in FedIns can be written as,

$$\mathbf{FI}_k^{z,t+1} \leftarrow \mathbf{FI}_k^{z,t} - \eta_k \nabla \ell_k(\mathbf{x}^k; \mathbf{FI}_k^{z,t}). \quad (7)$$

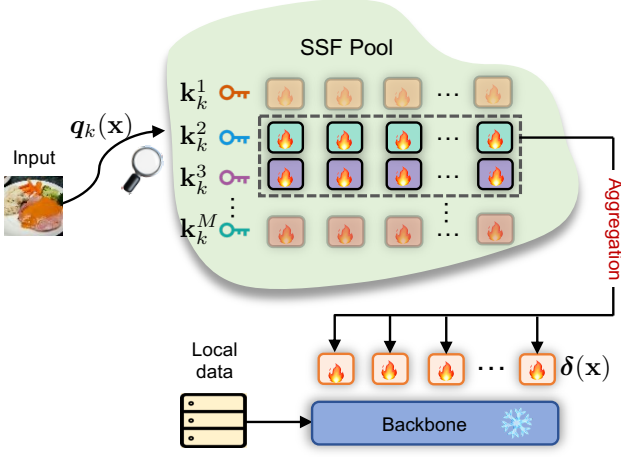


Figure 3: **Illustration** of instance-adaptive inference. For a given instance, we use the pre-trained model to generate a query, and then dynamically select the C best-matched SSFs, which are then aggregated to form an instance-adaptive SSF for enabling instance-adaptive inference.

After T local updates, the server update can be expressed as,

$$\mathbf{FI}_g^{z+1} \leftarrow \sum_{k=1}^K \frac{|\mathcal{D}^k|}{|\mathcal{D}|} \mathbf{FI}_k^z. \quad (8)$$

After Z rounds of communication, we can obtain the global model of FedIns, which is also parameterized by an SSF pool Δ_g and a set of keys \mathbf{K}_g . The detailed FedIns algorithm is given in Algorithm 1. In contrast to federated SSF, during the inference phase, FedIns allow to dynamically find the best matched SSF subsets from the SSF pool for a given instance, thereby making instance-adaptive inference feasible.

3.4. Instance-Adaptive Inference

When the training of FedIns is accomplished, the learned federated SSF pool Δ_g and the corresponding global Keys \mathbf{K}_g will be distributed to each client. As shown in Fig. 3, to enable instance-adaptive inference using FedIns, we resort to generating an instance-adaptive SSF from Δ_g and \mathbf{K}_g during the inference phase.

To this end, for a given instance \mathbf{x} of a client, we first use the pre-trained model to generate a query $q_k(\mathbf{x})$ which has the same dimension as the keys. Based on the cosine similarity between $q_k(\mathbf{x})$ and each keys in \mathbf{K}_g , we select the C best-matched SSFs, *i.e.*, $\{\delta_g^{m_1}, \delta_g^{m_2}, \dots, \delta_g^{m_C}\}$ from Δ_g corresponding to the C most similar keys. The instance-adaptive SSF of \mathbf{x} can then be given by,

$$\delta(\mathbf{x}) = \sum_{c=1}^C \frac{1}{C} \delta_g^{m_c}. \quad (9)$$

We note that $\delta(\mathbf{x})$ is determined by \mathbf{x} and is instance-adaptive. Thus, instance-adaptive inference can be ful-

Algorithm 1: FedIns

Input: Local datasets of K clients:

$\mathcal{D}^1, \mathcal{D}^2, \dots, \mathcal{D}^K$, local updates T , communication rounds Z , pre-trained model parameters θ , learnable parameters $\mathbf{FI}_k = \{\mathbf{K}_k, \Delta_k\}$, learning rate η , hyperparameter c , and M ;

Output: The final global model w ;

1 // **ServerExecution:**

2 Initialize global SSF pool $\Delta_k = \{\delta_k^1, \delta_k^2, \dots, \delta_k^M\}$

with Keys of $\mathbf{K}_k = \{\mathbf{k}_k^1, \mathbf{k}_k^2, \dots, \mathbf{k}_k^M\}$;

3 **for each communication round** $z \in \{1, 2, \dots, Z\}$ **do**

4 **for each client** $k \in \{1, 2, \dots, K\}$ **in parallel do**

5 $\mathbf{FI}_k^z \leftarrow \mathbf{FI}_g^z$;

6 Local updating with regard to each input instance \mathbf{x} :

$\mathbf{FI}_g^{z,t+1} \leftarrow \text{LocalUpdate}(k, \mathbf{x}, \mathbf{FI}_g^{z,t})$;

7 **end**

8 $\mathbf{FI}_g^{z+1} \leftarrow \sum_{k=1}^K \frac{|\mathcal{D}^k|}{|\mathcal{D}|} \mathbf{FI}_k^z$;

9 **end**

10 **return** \mathbf{FI}_g^{z+1}

11 // **LocalUpdate** ($k, \mathbf{x}, \mathbf{FI}_k^{z,t}$):

12 **for each local epoch** $t \in \{1, 2, \dots, T\}$ **do**

13 $\mathbf{FI}_k^{z,t+1} \leftarrow \mathbf{FI}_k^{z,t} - \eta_k \nabla \ell_k(\mathbf{x}^k; \mathbf{FI}_k^{z,t})$;

14 **end**

15 **return** $\mathbf{FI}_k^{z,t+1}$

filled by merging $\delta(\mathbf{x})$ into θ to obtain the instance-adaptive model parameter $\theta'(\mathbf{x})$.

4. Experiments

4.1. Experimental Setup

Implementation Details. We implement our method with Pytorch on one NVIDIA RTX 3090Ti GPU. During the federated training, all participants adopt the same hyperparameter settings, *e.g.*, $M = 25$, $C = 3$. Both collaborative and local updating use the stochastic gradient descent (SGD) optimizer with a batch size of 32 and a learning rate of 0.01.

Datasets. Experiments are conducted on two scenarios [27, 23], including Label Shift: **CIFAR-100** [18], and **Tiny-ImageNet**¹, and Feature Shift: **DomainNet** [34]. To simulate the FL scenario, we use Dirichlet distribution to split the training data of CIFAR-100 and Tiny-ImageNet into multiple non-i.i.d. clients, respectively, and evaluate the performance on the test data [24]. DomainNet consists of data from six different domains with the same class labels. Following [27], we deploy the data from each

¹<https://www.kaggle.com/c/tiny-imagenet>.

Table 1: **Accuracy** % of state-of-the-art FL methods on two scenarios, including Label Shift: **CIFAR-100** [18] and **Tiny-ImageNet**, and Feature Shift: **DomainNet** [34], where **# Com.cost** is the communication cost. w.Full indicates that the local model of a FL algorithm is fully fine-tuned. The arrow \uparrow and \downarrow indicate improvements and decrements compared with FedAvg (w.Full), respectively. Detailed analyses are provided in Sec. 4.2.

Method	# Com.cost	DomainNet	CIFAR-100	Tiny-ImageNet
SOLO	—	62.18(17.84) \downarrow	33.81(47.41) \downarrow	17.17(61.85) \downarrow
FedAvg ₂₀₁₇ (w.Full) [30]	85.80 M	80.02(0.00)	81.22(0.00)	79.02(0.00)
FedProx ₂₀₂₀ [25]	85.80 M	78.73(1.29) \downarrow	81.56(0.34) \uparrow	79.57(0.55) \uparrow
SCAFFOLD ₂₀₂₀ [16]	85.80 M	80.31(0.29) \uparrow	78.02(3.20) \downarrow	76.76(2.26) \downarrow
FedBN ₂₀₂₁ [27]	85.76 M	80.07(0.05) \uparrow	81.24(0.02) \uparrow	79.82(0.80) \uparrow
MOON ₂₀₂₁ [24]	85.80 M	81.65(1.64) \uparrow	81.92(0.70) \uparrow	81.38(2.18) \uparrow
FedDC ₂₀₂₂ [9]	85.80 M	79.92(0.10) \downarrow	78.44(2.78) \downarrow	79.81(0.79) \uparrow
FedIns (w. SSF Pool)	5.35 M	82.34 (2.32) \uparrow	84.11 (2.89) \uparrow	86.29 (7.27) \uparrow
FedIns (Ours)	5.35 M	83.12 (3.10) \uparrow	84.83 (3.61) \uparrow	86.79 (7.77) \uparrow

domain as a specific client. To demonstrate the effectiveness of our proposed approach, each client in our experiments has only a small number of images, *i.e.*, **DomainNet** is divided into six clients with only 26 images for each of them; **CIFAR-100** is divided into five clients with 123, 209, 168, 222, and 278 images, respectively; and **Tiny-ImageNet** is divided into five clients with 322, 360, 343, 466, and 509 images, respectively.

Baselines. We compare our method, FedIns, with various state-of-the-art FL algorithms, including: (1) FedProx [25], which alleviates the data heterogeneity by applying a proximal term to the local objective function; (2) SCAFFOLD [16], which corrects the client-drift by a series of control variates; (3) MOON [24], which corrects the local update by computing the similarity between model representations; (4) FedBN [27], which alleviates the client-shift by using batch normalization on each local client; (5) FedDC [9], which bridges the gap between the local and global model parameter by an auxiliary local drift variable and the classical FL algorithm; and (6) FedAvg [30], which trains a global model by averaging parameters from all the participating clients. In comparison, we also add (7) SOLO, where participants train a model on each client and their private data without FL. For a fair comparison, we use ViT-B/16 [5] pre-trained on ImageNet-21K as the backbone of all the methods in our experiments.

4.2. Comparison with State-of-the-arts

To assess the effectiveness of our method, FedIns, we compared it to the above state-of-the-art FL algorithms, which aim to alleviate the data heterogeneity. For a fair comparison, all methods are retrained using the same model with their best hyperparameters. Specifically, for MOON, the hyperparameter of μ is set to 1. For FedProx [25], the hyperparameter to control the weight of its proximal term is set to 0.001. For FedDC, we set the hyperparameter of α to 0.01. Here, all the competing methods are trained with 300 communication rounds and 10 local

epochs for each round. As the results shown in Table 1, our FedIns consistently outperform baselines in all settings, achieving state-of-the-art results in most cases. In particular, on **DomainNet**, **CIFAR-100**, and **Tiny-ImageNet** datasets, our method improves the accuracy of FedAvg from 80.02%, 81.22%, and 79.02% to **83.12%**, **84.83%**, and **86.79%**, respectively. These results indicate the robustness of our method in the two scenarios, *i.e.*, Label Shift and Feature Shift. The FedAvg (w.Full) [30] is prone to overfitting, even it uses the pre-trained model locally when local data is limited. However, the parameter-efficient method of freezing the backbone network is consistent with the mechanism of personalized FL which is also an effective way to improve FL performance. More importantly, the number of training parameters and communication cost of our method are only 15% of those of FedAvg (w.Full) [30]. It is worth noting that FedProx [25] and SCAFFOLD [16] perform worse than FedAvg [30], *i.e.*, DomainNet: 80.02% \rightarrow **78.73%** and CIFAR-100: 81.22% \rightarrow **78.02%**. This is primarily due to the fact that correcting local updates through proximal terms and control variates will result in some deviations when there are only a few training datasets on the local clients. Differently, FedDC dynamically bridges the gap between the local model and the global model in the parameter aggregation stage, making the global model able to alleviate the data heterogeneity [9]. Nonetheless, even compared with FedDC [9], which is the latest work aiming to address the statistical heterogeneity, our method still achieves a 8.75% improvement on **Tiny-ImageNet**, (*i.e.*, 79.81% \rightarrow **86.79%**).

In contrast, the results of SOLO are lower than those of other FL algorithms, although it is also fine-tuned on a pre-trained model, which demonstrates the benefits of FL. These classical FL algorithms only consider the inter-client data heterogeneity, ignoring the intra-client data heterogeneity, which will also degrade the FL performance. This finding confirms our core idea that the performance of FL algorithms can be improved by simultaneously re-

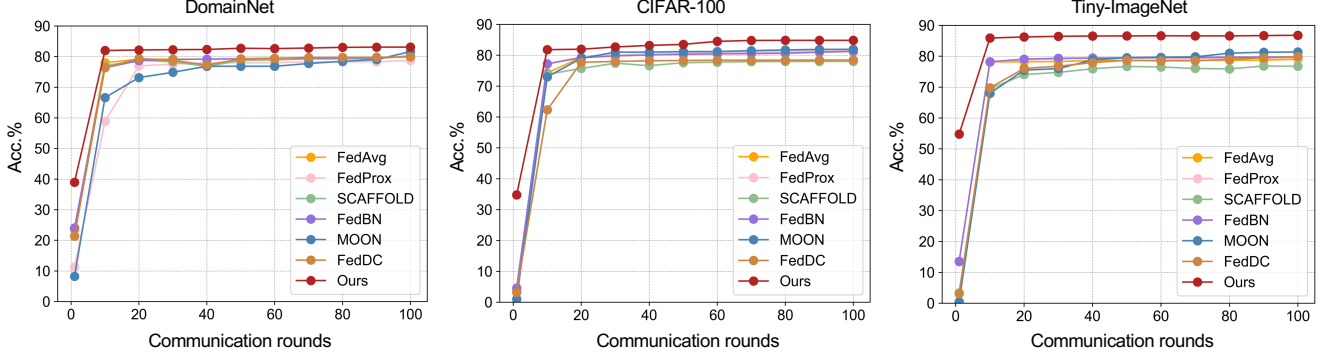


Figure 4: **Accuracy comparison** of different state-of-the-art FL algorithms in terms of different communication rounds on two scenarios, including Label Shift: **CIFAR-100** [18] and **Tiny-ImageNet**, and Feature Shift: **DomainNet** [34] datasets.

Table 2: **Ablation** studies with regard to the *key components* of FedIns on the three datasets, where **#New.Param.** represents the new introduced parameters of each method, ↓ indicates decrements compared with our full model *SSF Pool (Ours)*. Detailed analyses are provided in Sec.4.5.

Variation	Prompt	SSF	Pool	#New.Param.	DomainNet	CIFAR-100	Tiny-ImageNet
<i>Prompt</i>	✓	—	—	0.10 M	79.03(4.09) ↓	80.25(4.58) ↓	81.02(5.77) ↓
<i>Prompt Pool</i>	✓	—	✓	2.32 M	80.17(2.95) ↓	81.39(3.44) ↓	81.98(4.81) ↓
<i>SSF</i>	—	✓	—	0.00 M	80.75(2.37) ↓	83.81(1.02) ↓	84.32(2.47) ↓
<i>SSF Pool (Ours)</i>	—	✓	✓	5.16 M	83.12(0.00)	84.83(0.00)	86.79(0.00)

lieving the inter- and intra-client data heterogeneity. In the next section, we analyze the communication efficiency of FedIns and the influence of FedIns on inter-/intra-client data heterogeneity and local epoch.

4.3. Communication Efficiency Analysis

Our primary goal is to investigate how to efficiently alleviate inter- and intra-client heterogeneity and make instance-adaptive models feasible for FL, leveraging FL to perform well with less local data, a smaller number of parameters, and lower communication costs. Therefore, we evaluate the communication efficiency of the proposed method in terms of communication cost, number of parameters, and different numbers of communication rounds. As shown in Table 1, FedIns only requires 5.35 M of communication, which is 15% of the others. Similarly, the number of learnable parameters of FedIns is only 5.35 M. Therefore, our method has high efficiency in both local client updates and server communication. On the contrary, although the classical federated algorithms MOON [24] and FedBNs [27] can also achieve good accuracy, they require a large amount of local computation and incur high communication costs. In Fig. 4, we compare the accuracy of all methods under different communication rounds to demonstrate the superior communication efficiency of our method. We note that for all competing methods, the local epoch of each method is fixed at 10. As can be seen from this figure, FedIns reached stability in the 10-th round, while the other methods needed to reach stability after 20 rounds. This indicates that our FedIns dynamically guides the local model

to alleviate the intra-client data heterogeneity, enabling the instance-adaptive model feasible in FL.

4.4. Effect of Statistical Heterogeneity

As we mentioned before, our proposed method can improve the performance of FL algorithms by alleviating both inter- and intra-client data heterogeneity. Therefore, here we explore whether the proposed method remains effective as inter- and intra-client data heterogeneity increases. First, we examine the influence of inter-client data heterogeneity on our algorithm by changing the concentration parameter β of the Dirichlet distribution. The smaller β values, the higher inter-client data heterogeneity [24]. Fig. 5 (a) shows the classification accuracies of β values vary in $\{0.1, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5\}$. As can be seen from the figure, the classification accuracies of all the FL algorithms increase with the increasing β value. FedIns, on the other hand, consistently has the highest classification accuracy and is least affected by β . However, with the increase in inter-client data heterogeneity, the performance of FedAvg [30] will rapidly decline. On the contrary, FedIns is least affected by heterogeneity and still holds an accuracy of 85.13% when $\beta = 0.1$. Then, we fix $\beta = 0.2$, and randomly convert some local images to other styles to control the intra-client data heterogeneity [10].

Similarly, the bigger γ values, the higher intra-client data heterogeneity. Fig. 5 (b) shows the classification accuracies of γ values vary in $\{2, 4, 6, \dots, 20\}$. Similar to Fig. 5 (a), the classification accuracy scores of all the FL algorithms decrease with the increasing γ value. These baseline meth-

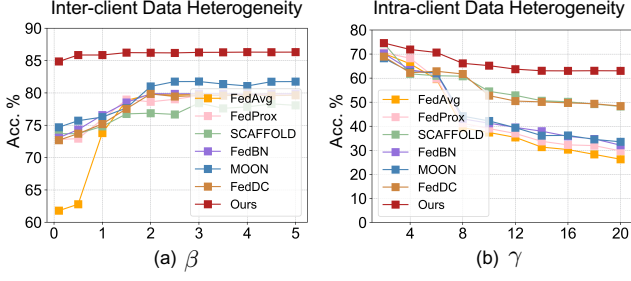


Figure 5: **Accuracy comparison** of different state-of-the-art FL algorithms in terms of various (a) *inter-client heterogeneity* and (b) *intra-client heterogeneity* on **Tiny-ImageNet**, where the smaller β values, the higher inter-client heterogeneity; the larger γ values, the higher intra-client heterogeneity.

ods, in particular, show the most marked downward trend. Nonetheless, when $\gamma = 20$, our proposed method only decreases the accuracy from 75.63% to 64.15%. In contrast, the classical FL algorithm FedBN, which aims to solve the data heterogeneity problem, decreases the accuracy from 68.53% to **31.69%** when $\gamma = 20$.

4.5. Ablation Study

Effectiveness of Core Designs. We first analyze the key components of FedIns on three datasets to evaluate the effectiveness of our core designs. The evaluated components include SSF pool in our model, and SSF, which is the parameter-efficient mechanism in our model. To this end, we built four derived ablation models, where *Prompt* represents that the SSF of our method is replaced by the similar parameter-efficient mechanism prompts [14], *Prompt Pool* represents employing the prompt pool mechanism on *Prompt*, *SSF* indicates that the FedIns model removes the SSF pool mechanism, and only retains the parameter-efficient mechanism of SSF in the federated framework, and *SSF Pool* represents our FedIns that preserves the full components. Following [14], we add the prompt embeddings with a size of $10 \times 12 \times 768$. Table 2 summarizes the classification accuracy of all ablation models. As compared with *Prompt*, we observed that *SSF* improves the performance from 81.02% to **84.32%** on the **Tiny-ImageNet** dataset, indicating that SSF can provide better performance than prompt since prompt is often sensitive to data and tasks and needs to be carefully designed for each client. Additionally, SSF achieves zero overhead by reparameterizing them into the original pre-trained model weights at the inference stage, while prompt-based methods introduced additional parameters, *i.e.*, *Prompt* with 0.10 M and *Prompt Pool* with 2.32 M parameters. This supports our design of using SSF to fine-tune the local model and communicate with the server. *SSF Pool* (Ours) equipped with all components, produces the best classification accuracies, improving the results from 81.02% to **86.79%** on the Tiny-ImageNet dataset. In general, our core designs can help

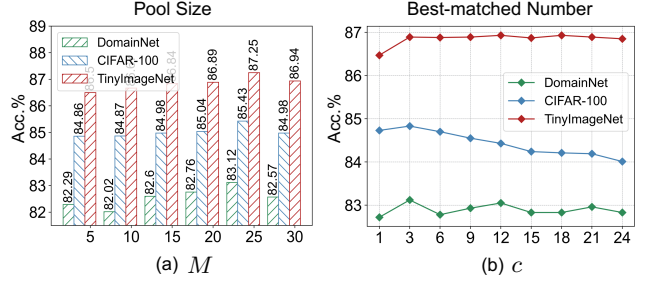


Figure 6: **Ablation studies** versus (a) *Pool size analysis* and (b) *Best-matched number analysis* on **DomainNet**, **CIFAR-100**, and **Tiny-ImageNet** datasets.

to alleviate federated data heterogeneity and enhancing the performance of FL algorithms.

Pool Size Analysis. Here, we focus on analyzing the effect of the pool size on model effectiveness. We can freely select the size with the greatest performance even though the number of network parameters increases with pool size because the number of parameters for SSF is small, *i.e.*, $M = 1$ only requires 0.20 M parameters. Additionally, these parameters can be incorporated into the original pre-trained weights by model reparameterization at the inference stage, thereby avoiding additional parameters for the downstream tasks. We record the performance of different pool sizes on the three datasets in Fig. 6 (a). It can be seen from the figure that as the pool size increases, the performance of FedIns on the three datasets will improve. When $M = 25$, our method yields the best classification performance while only requiring 5.35 M parameters. When M is greater than 25, the performance degrades due to a large amount of redundancy, which affects the update of the local client by instance-adaptive models. It is worth noting that our model always preserves a higher classification accuracy than the baseline, *i.e.*, when $M = 5$, Ours preserve the results of Acc. = **86.50%** *vs.* FedBN: Acc. = 79.81%.

Best-matched Number Analysis. As the selected SSF values for each input will affect the update of the local model, we need to choose an appropriate best-matched number C to train a good SSF pool. In Fig. 6 (b), we record the classification accuracy of c values varying in $\{1, 3, 6, \dots, 24\}$. As can be seen from this figure, we observe that our proposed method obtains the highest classification accuracy when $C = 3$ and the accuracy decreases monotonically on the **CIFAR-100** when the value of C increases, while remaining invariant on the **Tiny-ImageNet** dataset. However, even at the lowest point, our method still produces higher classification accuracy than the baselines in the two scenarios, *i.e.*, Label Shift and Feature Shift (see Table 1). Therefore, this study reveals the effectiveness of our proposed SSF pool mechanism.

5. Conclusion

In this work, we re-examine the problem of data heterogeneity in FL, and find that there is not only inter-client data heterogeneity but also intra-client heterogeneity in many complex real-world scenarios, which also significantly degrades FL performance. To address this issue, we propose a new parameter-efficient fine-tuning FL algorithm, FedIns, which makes it possible to use instance-adaptive inference for FL, thus enabling dynamically guided locals to alleviate intra-client data heterogeneity. Unlike existing approaches, FedIns trains an SSF pool for each client and aggregates them into the federated SSF pool on the server. For a given instance, FedIns dynamically finds the best-matched SSF subsets from the pool, and aggregates them to generate an adaptive SSF for instance-adaptive inference. In addition, FedIns introduces only a small number of learnable parameters with the help of the large-scale pre-training model, and greatly reduces the communication cost. Extensive experiments show the superiority of FedIns in handling the inter- and intra-client data heterogeneity issue.

Acknowledgements: This work was supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-TC-2021-003) and Agency for Science, Technology and Research (A*STAR) through its RIE2020 Health and Biomedical Sciences (HBMS) Industry Alignment Fund Pre-Positioning (IAF-PP) (grant no. H20C6a0032).

References

- [1] Ismaeel Al Ridhawi, Safa Otoum, Moayad Aloqaily, and Azzedine Boukerche. Generalizing ai: challenges and opportunities for plug and play ai solutions. *IEEE Network*, 35(1):372–379, 2020.
- [2] Hong-You Chen, Cheng-Hao Tu, Ziwei Li, Han-Wei Shen, and Wei-Lun Chao. On pre-training for federated learning. *arXiv preprint arXiv:2206.11488*, 2022.
- [3] Ziliang Chen, Jingyu Zhuang, Xiaodan Liang, and Liang Lin. Blending-target domain adaptation by adversarial meta-adaptation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2248–2257, 2019.
- [4] Enmao Diao, Jie Ding, and Vahid Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. *arXiv preprint arXiv:2010.01264*, 2020.
- [5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [6] Chun-Mei Feng, Bangjun Li, Xinxing Xu, Yong Liu, Huazhu Fu, and Wangmeng Zuo. Learning federated visual prompt in null space for mri reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8064–8073, 2023.
- [7] Chun-Mei Feng, Yunlu Yan, Huazhu Fu, Li Chen, and Yong Xu. Task transformer network for joint mri reconstruction and super-resolution. In *Medical Image Computing and Computer Assisted Intervention—MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part VI 24*, pages 307–317. Springer, 2021.
- [8] Chun-Mei Feng, Yunlu Yan, Shanshan Wang, Yong Xu, Ling Shao, and Huazhu Fu. Specificity-preserving federated learning for mr image reconstruction. *IEEE Transactions on Medical Imaging*, 2022.
- [9] Liang Gao, Huazhu Fu, Li Li, Yingwen Chen, Ming Xu, and Cheng-Zhong Xu. Feddc: Federated learning with non-iid data via local drift decoupling and correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10112–10121, 2022.
- [10] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.
- [11] Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. Pre-trained models: Past, present and future. *AI Open*, 2:225–250, 2021.
- [12] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.
- [13] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.
- [14] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. *arXiv preprint arXiv:2203.12119*, 2022.
- [15] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035, 2021.
- [16] 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*, pages 5132–5143. PMLR, 2020.
- [17] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR, 2021.
- [18] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
- [19] Gihun Lee, Minchan Jeong, Yongjin Shin, Sangmin Bae, and Se-Young Yun. Preservation of the global knowledge by not-true distillation in federated learning. In *Advances in Neural Information Processing Systems*, 2022.
- [20] Gihun Lee, Yongjin Shin, Minchan Jeong, and Se-Young

- Yun. Preservation of the global knowledge by not-true self knowledge distillation in federated learning. *arXiv preprint arXiv:2106.03097*, 2021.
- [21] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.
- [22] Haoran Li, Chun-Mei Feng, Tao Zhou, Yong Xu, and Xiaojun Chang. Prompt-driven efficient open-set semi-supervised learning. *arXiv preprint arXiv:2209.14205*, 2022.
- [23] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 965–978. IEEE, 2022.
- [24] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10713–10722, 2021.
- [25] 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.
- [26] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.
- [27] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021.
- [28] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. *arXiv preprint arXiv:2210.08823*, 2022.
- [29] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.
- [30] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.
- [31] Matias Mendieta, Taojiannan Yang, Pu Wang, Minwoo Lee, Zhengming Ding, and Chen Chen. Local learning matters: Rethinking data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8397–8406, 2022.
- [32] John Nguyen, Kshitiz Malik, Maziar Sanjabi, and Michael Rabbat. Where to begin? exploring the impact of pre-training and initialization in federated learning. *arXiv preprint arXiv:2206.15387*, 2022.
- [33] Xing Nie, Bolin Ni, Jianlong Chang, Gaomeng Meng, Chunlei Huo, Zhaoxiang Zhang, Shiming Xiang, Qi Tian, and Chunhong Pan. Pro-tuning: Unified prompt tuning for vision tasks. *arXiv preprint arXiv:2207.14381*, 2022.
- [34] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019.
- [35] Timo Schick and Hinrich Schütze. It’s not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*, 2020.
- [36] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.
- [37] Yang Shu, Zhi Kou, Zhangjie Cao, Jianmin Wang, and Mingsheng Long. Zoo-tuning: Adaptive transfer from a zoo of models. In *International Conference on Machine Learning*, pages 9626–9637. PMLR, 2021.
- [38] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.
- [39] Chencheng Xu, Zhiwei Hong, Minlie Huang, and Tao Jiang. Acceleration of federated learning with alleviated forgetting in local training. *arXiv preprint arXiv:2203.02645*, 2022.
- [40] Haolin Yuan, Bo Hui, Yuchen Yang, Philippe Burlina, Neil Zhenqiang Gong, and Yinzhi Cao. Addressing heterogeneity in federated learning via distributional transformation. In *European Conference on Computer Vision*, pages 179–195. Springer, 2022.
- [41] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazani. Bayesian nonparametric federated learning of neural networks. In *International conference on machine learning*, pages 7252–7261. PMLR, 2019.
- [42] Yuliang Zou, Zizhao Zhang, Chun-Liang Li, Han Zhang, Tomas Pfister, and Jia-Bin Huang. Learning instance-specific adaptation for cross-domain segmentation. *arXiv preprint arXiv:2203.16530*, 2022.

Contents

The following items are included in our supplementary material:

- Detailed illustration of Fig. 1 (c) and (d) in Section A.
- Exemplars of the converted style for the intra-client data heterogeneity in Section B.

A. Detailed Illustration of Fig. 1 (c) and (d)

We redisplay Fig. 1 (c) and (d) in Fig. A1 in details. As can be seen from this figure, the accuracy % comparisons of FedAvg [30], FedBN [27], and FedIns in terms of various (a) *inter-client heterogeneity* and (b) *intra-client heterogeneity* on **DomainNet** are recorded. We use the hyperparameter β of Dirichlet distribution to control the inter-client heterogeneity and γ to control the intra-client heterogeneity. The smaller β values, the higher inter-client heterogeneity; the larger γ values, the higher intra-client heterogeneity, where γ is the number of the converted styles [10]. We provide the exemplars of the converted styles in Sec. B.

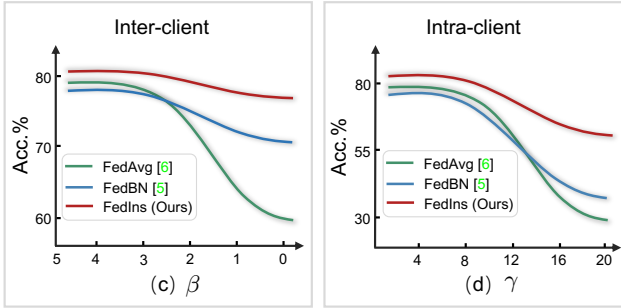


Figure A1: **Accuracy comparisons** of FedAvg [30], FedBN [27], and FedIns in terms of various (a) *inter-client heterogeneity* and (b) *intra-client heterogeneity* on **DomainNet**, where β is hyperparameter of Dirichlet distribution, the smaller β values, the higher inter-client heterogeneity; γ is the number of the converted styles (see Sec. B for details), the larger γ values, the higher intra-client heterogeneity.

B. Exemplars of the Intra-client Data Heterogeneity

As we mentioned, we randomly convert some local images to other styles to control the intra-client data heterogeneity [10]. Here, we show some exemplars of the converted images on the client `clipart` under **DomainNet** dataset in Figure. A2. As can be seen from this figure, different styles show great differences, thereby making the local data preserve higher intra-client data heterogeneity and being able to be used to demonstrate the effectiveness of our method.

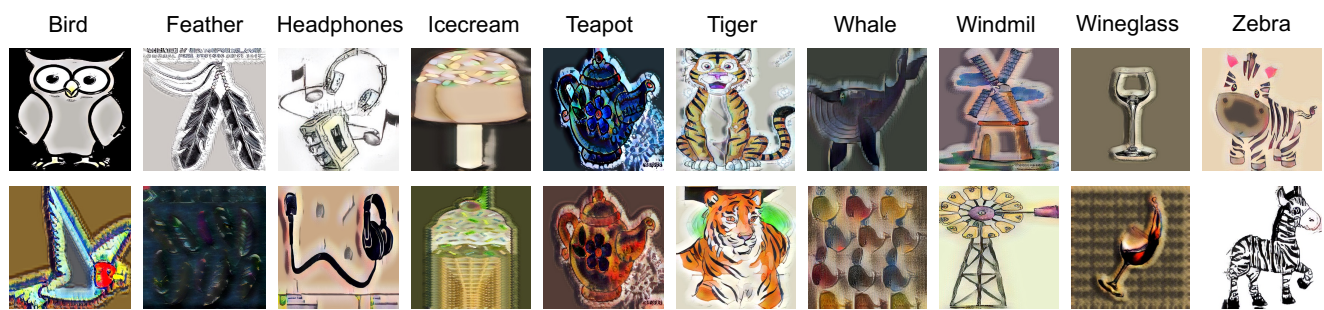


Figure A2: **Visualization** of the convert images with various style on **DomainNet** to control the intra-client data heterogeneity.