Subspace Constraint and Contribution Estimation for Heterogeneous Federated Learning

Xiangtao Zhang, Sheng Li, Ao Li, Yipeng Liu, Fan Zhang, Ce Zhu, Le Zhang ✉
SICE, UESTC, Chengdu, China
{xiangtaozhang, lisheng2425, aoli}@std.uestc.edu.cn,
{yipengliu, fan.zhang, eczhu, lezhang}@uestc.edu.cn

Abstract

Heterogeneous Federated Learning (HFL) has received widespread attention due to its adaptability to different models and data. The HFL approach utilizing auxiliary models for knowledge transfer can further enhance flexibility. However, existing frameworks face the challenges of local overfitting and aggregation bias. To address these issues, we propose FedSCE. By restricting specific layers of the local model updates to a subspace, FedSCE reduces the degrees of freedom of the update, enhances generalization, and mitigates the risk of overfitting. The subspace is dynamically updated to ensure coverage of the latest model update trajectory. Additionally, FedSCE evaluates client contributions based on the update distance of the auxiliary model in feature space and parameter space, achieving adaptive weighted aggregation. We validate our approach in both feature-skewed and label-skewed scenarios, demonstrating that on Office10, our method exceeds the best baseline by 3.87%. The code will be available at https://github.com/AVC2-UESTC/FedSCE.git.

1. Introduction

Federated Learning (FL) has recently emerged as a collaborative machine learning paradigm that enables model training while preserving local data privacy [30, 31, 49]. However, Traditional Federated Learning (TFL) methods often perform poorly when dealing with heterogeneous data [18, 22, 32]. To solve this problem, Personalized Federated Learning (PFL) was proposed by training customized models on the client [33, 50]. However, most PFL methods maintain a uniform model architecture [33, 54, 55], which may not suit the unique requirements of certain organizations or individuals [52].

Therefore, Heterogeneous Federated Learning (HFL) has been introduced as a paradigm that can handle both data and model heterogeneity [52]. HFL allows clients to use different model architectures, making them adaptable to different hardware and specific task requirements. In addition, these methods explore many novel knowledge transfer strategies beyond simply aggregating model parameters [17, 28, 39, 57].

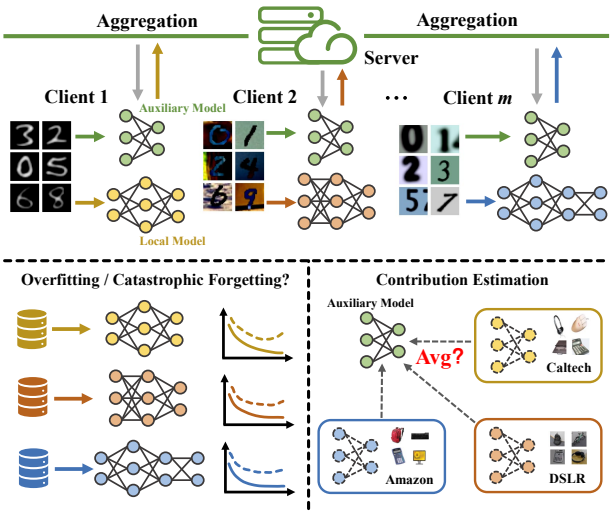


Figure 1. **Challenges** in heterogeneous federated learning based on auxiliary model: 1) *In the local training stage*, local models often suffer from overfitting and catastrophic forgetting, resulting in suboptimal performance. 2) *In the auxiliary model aggregation stage*, assessing the contributions of each client is challenging, which hinders effective information transmission.

Current HFL methods can be broadly categorized into knowledge distillation-based, prototype-based, and auxiliary model-based according to knowledge transfer strategies [57, 58]. Knowledge distillation methods [17, 28] use a global dataset on the server to facilitate knowledge transfer. However, the availability and quality of this dataset significantly impact distillation performance [16, 58]. Prototype-based approaches [42, 43, 57] create a shared class prototype for all clients, which serves as global information. However, these prototypes often lack sufficient expressive-

✉ Corresponding author.

ness. When client data distributions differ, prototypes of the same class may vary widely across clients. This limits the ability of a unified prototype to adapt to different client data. Auxiliary model-based approaches [39, 48] support joint training of local and global models. In this framework, global models are aggregated on the server to facilitate knowledge sharing, while private models are continuously trained locally to capture domain-specific knowledge.

Although auxiliary model-based HFL has shown initial promise, it still faces fundamental challenges inherent in the federated learning framework. FL consists of two key stages: local model training and global knowledge transfer. During local training, models are continuously updated using private data, yet limited data often leads to local overfitting [47, 53, 59] and catastrophic forgetting [17, 36]. To mitigate these issues, existing methods usually incorporate pre-trained models [17] or regularization techniques [23, 32, 46] to guide local training with early-stage information. However, these methods rely on pre-trained models and fail to fully leverage both early and global information. Therefore, ❶ we need to design a strategy that fully utilizes early knowledge to reduce model complexity and alleviate overfitting.

During global knowledge transfer, data heterogeneity introduces biases that complicate auxiliary model aggregation [9, 10, 20, 38]. Traditional average aggregation overlooks client contribution differences, leaving the auxiliary model without sufficient local insights. This raises a crucial question: ❷ how can client contributions be effectively estimated and enable the auxiliary model to integrate knowledge across all clients more accurately?

To tackle the above challenges, we introduce a novel HFL method called ***Fed****erated learning via* ***S****ubspace Constraint and* ***C****ontribution* ***E****stimation (****FedSCE****)*. First, to mitigate local overfitting, we draw inspiration from subspace learning [34, 35], which improves both training efficiency and generalization ability by constraining the network to optimize within a low-dimensional subspace. Reframing subspace learning in terms of model update directions, we find that generalization improves by reducing redundant directions in the update process. In the context of HFL, we consider restricting the update direction of the local model to a low-dimensional subspace, thereby reducing the space complexity of model update. To further minimize the subspace storage cost, we selectively apply this constraint to specific model layers. Furthermore, we dynamically adjust the subspace to cover the entire training trajectory, ensuring that accumulated prior knowledge is effectively integrated into subsequent updates.

Second, to address the model aggregation bias problem, we propose a new weighting strategy, auxiliary model contribution estimation, to focus on client insights lacking in auxiliary models. This strategy evaluates the contribution of the client knowledge to the auxiliary model by estimating the update distance of the auxiliary model in both feature and model space.

We conducted extensive experiments to verify the effectiveness of FedSCE, comparing it with six methods under feature-skewed and label-skewed scenarios. Results demonstrate that FedSCE consistently outperforms other algorithms. Specifically, on the *Office10* dataset, FedSCE achieved an average accuracy improvement of 3.08% at the data level and 1.70% at the client level compared to the best baseline. Our main contributions are as follows:

- We explore the overfitting problem of FL and introduce a hierarchical subspace constrained training scheme. By constraining specific layers to dynamically updated low-rank subspaces, we reduce the degrees of freedom in the model update direction and reduce the subspace storage.
- We propose an auxiliary model contribution estimation strategy. By estimating the update distance in parameter and feature space to reflect the contribution of different clients, we solve the auxiliary model aggregation problem and promote knowledge transfer.
- We conduct comprehensive experiments under feature-skewed and label-skewed scenarios. On *Office10*, our method achieves an accuracy improvement of up to 3.87%.

2. Related Work

2.1. Heterogeneous Federated Learning

We categorize existing federated learning frameworks into two types: data heterogeneity and model heterogeneity.

Data Heterogeneous Federated Learning. The pioneering FedAvg method [37] introduced the idea of aggregating client-specific local model parameters to build a global model. However, FedAvg [37] is primarily suited for homogeneous data and often faces challenges in non-IID (non-independent and identically distributed) settings. Many methods have extended FedAvg [37] to address data heterogeneity. For instance, FedProx [32] and FedDyn [1] introduce regularization terms to limit local updates, while approaches like SCAFFOLD [22] and FedDC [11] utilize global gradient adjustments to counteract local model drift. MOON [29] improves alignment between local and global model representations using contrastive learning, and FedNTD [26] applies local non-true distillation to prevent global knowledge loss during local training. These methods only focus on the global model, which fails to address the personalized needs of individual clients.

Model Heterogeneous Federated Learning. Heterogeneous federated learning methods can be classified into *partially heterogeneous* (same model architecture, different parameters) and *fully heterogeneous* (different architectures and parameters). Personalized federated learning (PFL) is

Description		Description	
M	Client volume	T	Communication rounds
P	Local subspace	$\Delta\hat{\theta}_m^t$	Vectorized layer update
E	Local epochs	$q_{g,i}$	Auxiliary model logits
η	Learning rate	$q_{l,i}$	Local model logits
θ_g	Auxiliary model	z	Feature prototype
θ_m	Local model	p	Number of constraint layers
s	Subspace update gap	R	Rank of subspace
θ_m^t	Local model	$\theta_{g,m}^t$	Auxiliary model of client m

Table 1. A list of notations and corresponding descriptions.

a typical *partially heterogeneous* method [6, 7, 51]. Methods like FedPer [2] and FedRep [7] keep local classifiers while sharing a feature extractor to capture general features. Techniques such as pFedMe [41] and Ditto [33] adjust the objective of personalized model training by adding regularization to balance local and global information. However, these methods rely on a fixed model architecture, which can limit flexibility in meeting varied client needs.

Fully heterogeneous schemes in HFL can be categorized into three main types: knowledge distillation-based, prototype-based, and auxiliary model-based approaches. Knowledge distillation methods, such as FedMD [28] and FCCL [17], achieve knowledge transfer through server-side distillation but are highly dependent on the availability and quality of public datasets. Prototype-based methods, including FedProto [42] and FedTGP [57], establish shared category prototypes across clients; however, their limited expressiveness often hinders adaptability to diverse client data distributions. Similarly, FedDistill [19] enables information exchange through shared category logits, but when client characteristics vary significantly, logits or prototypes for the same category may show considerable deviation. FML [39] and FedKD [48] enhance knowledge sharing through mutual learning between auxiliary models and client models. Although these methods have made some progress, challenges still exist in local fitting. In addition, heterogeneous environments complicate model aggregation, and interference between auxiliary models and the local model may occur in the early stages of training.

2.2. Training in Subspace

Numerous studies have emphasized the inherent low-dimensional structure in neural networks [14, 44, 45]. The seminal work by [27] introduced the idea of training neural networks within a tiny subspace via random projection, showing that only a fraction of the original parameter dimensions are required to reach 90% of the performance achieved by conventional training. Building on this, [13] improves the approach by redrawing the random basis at each step, enhancing the effectiveness of random subspace training. In contrast to random projection, [35] proposed leveraging subspaces derived from prior training trajectories, achieving comparable results to traditional training methods. Additionally, [34] adaptively constructs model parameters by performing a combinatorial coefficient search within the subspace basis, improving generalization. However, current approaches rely on projecting all gradients into a fixed subspace, which becomes infeasible for large-scale networks due to hardware storage burden and fails to adapt to the dynamic nature of network updates.

3. Methodology

3.1. Problem Definition

We consider a standard federated learning setup with M clients (indexed by m), where each client maintains a local model θ_m and a private dataset $D_m = \{x_i, y_i\}_{i=1}^{N_m}$, with N_m denoting the size of the local data. In this setting, the client data usually follows a different distribution, denoted as $P_m(x, y)$. This distribution can be decomposed into $P_m(x|y)P_m(y)$, allowing us to classify data skew into two main types: **label skew** and **feature skew**:

- *Label skew*: The label distribution $P_m(y)$ differs across clients, with a same conditional distribution, defined as:
$$P_m(x|y) = P_n(x|y), \quad \text{while} \quad P_m(y) \neq P_n(y). \quad (1)$$
- *Feature skew*: Clients share the same label space, but the conditional feature distribution $P_m(x|y)$ varies among them, described as:
$$P_m(x|y) \neq P_n(x|y), \quad \text{while} \quad P_m(y) = P_n(y). \quad (2)$$

In HFL, each client customizes an independent model, and the global optimization objective is expressed as:

$$\arg\min_{\theta} \mathcal{L}(\theta) = \sum_{m=1}^{M} W_m \cdot \mathcal{L}_m(\theta_m), \quad (3)$$

where $\mathcal{L}_m(\theta_m) = \mathbb{E}_{(x,y)\sim D_m}[\ell(f(x, \theta_m), y)]$ denotes the local empirical loss, and $\ell(\cdot)$ represents the loss function.

In heterogeneous federated learning, clients have differing local model architectures (*i.e.*, $Shape(\theta_m) \neq Shape(\theta_n)$), making FedAvg-based knowledge transfer infeasible. In this work, we use an auxiliary/global model θ_g to facilitate knowledge transfer across clients. Each model consists of two components: a feature extractor and a classification head. The feature extractor g encodes sample x as a I-dimensional feature vector $z = g(x) \in \mathbb{R}^I$, while the classification head h maps the feature z to the logits $q = h(z) \in \mathbb{R}^C$, where C denotes the number of categories.

3.2. Hierarchical Subspace Constraint

3.2.1. Local Deep Mutual Learning

In a heterogeneous federated framework using an auxiliary model, each client operates two models: a local model customized to the specific data and tasks of the client, and a global model that enables knowledge transfer across clients. Both models are trained on the client's local dataset. Given

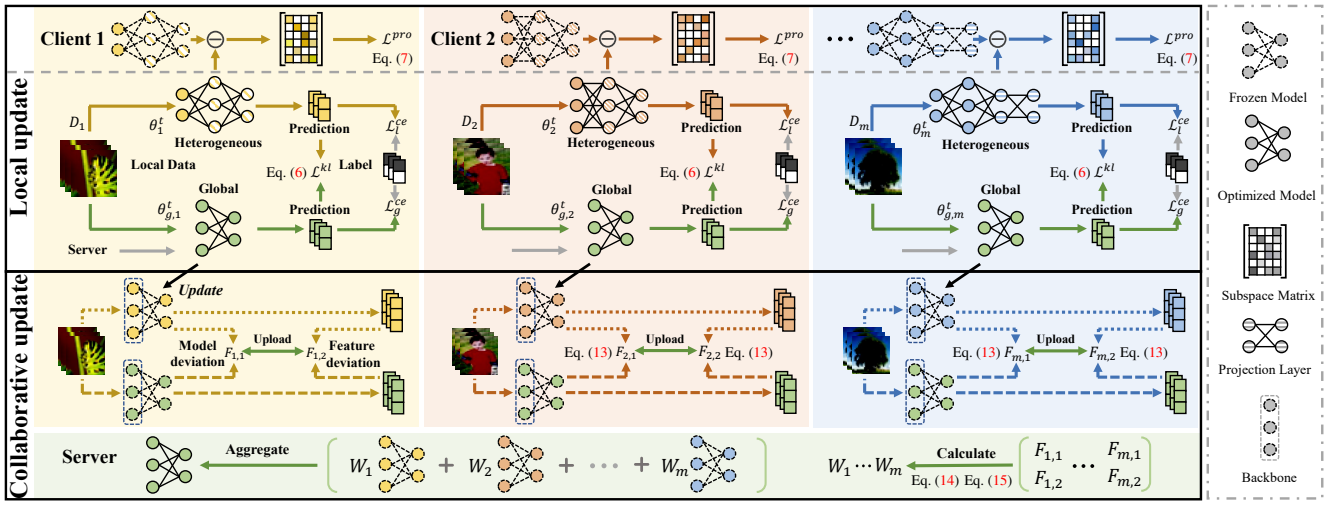


Figure 2. **Schematic diagram** of the FedSCE. First, each client selects the projection layer of the local model and constrains updates within a local subspace using $\mathcal{L}^{pro}$ to reduce the degree of freedom (LSL in Sec. 3.2.2). Then, mutual learning occurs through $\mathcal{L}^{ce}$ and $\mathcal{L}^{kl}$ (DML in Sec. 3.2.2). Finally, each client calculates the update distances $F_{m,1}$ and $F_{m,2}$ in the feature and parameter spaces and uploads them. The server evaluates the contribution of each auxiliary model based on the update bias and aggregates them (CE in Sec. 3.3).

the logits output $q_{g,i} = f(x_i, \theta^t_{g,m}), q_{l,i} = f(x_i, \theta^t_m)$, the local cross-entropy losses are defined as:

$$\begin{aligned} \mathcal{L}^{ce}_g &= -\mathbf{1}_{y_i} \log\left(\texttt{softmax}\left(q_{g,i}\right)\right), \\ \mathcal{L}^{ce}_l &= -\mathbf{1}_{y_i} \log\left(\texttt{softmax}\left(q_{l,i}\right)\right), \end{aligned} \tag{4}$$

where $\mathbf{1}_{y_i}$ denotes the one-hot encoding of y_i and $\texttt{softmax}\left(q_{g,i}\right) = \frac{\exp(q_{g,i})}{\sum_{c'=1}^{C}\exp(q^{c'}_{g,i})}$. In mutual learning, both models are independently trained on private data and consensus is reached by minimizing the difference between the predicted distributions. The distance between the prediction distributions of $\theta^t_{g,m}$ and θ^t_m is measured using *KL* divergence, as shown below:

$$KL(q_{g,i}, q_{l,i}) = \sum_{c'=1}^{C} q^{c'}_{g,i} \log \frac{q^{c'}_{g,i}}{q^{c'}_{l,i}}. \tag{5}$$

Building on the approach in [48], we adaptively tune the *KL* loss coefficient to balance distillation strength and task performance. When predictions from both models are unreliable (*i.e.*, both task losses $\mathcal{L}^{ce}_l$ and $\mathcal{L}^{ce}_g$ are high), the influence of distillation is reduced, allowing the task loss to dominate. As training progresses and task losses decrease, distillation loss plays a more important role, which may also help mitigate overfitting. The distillation losses for each model are defined as:

$$\begin{aligned} \mathcal{L}^{kl}_g &= KL\left(q_{l,i}, q_{g,i}\right) / \left(\mathcal{L}^{ce}_l + \mathcal{L}^{ce}_g\right), \\ \mathcal{L}^{kl}_l &= KL\left(q_{g,i}, q_{l,i}\right) / \left(\mathcal{L}^{ce}_g + \mathcal{L}^{ce}_l\right). \end{aligned} \tag{6}$$

3.2.2. Local Model Subspace Learning

In federated learning scenarios, client data is often limited, increasing the risk of overfitting if the local model is highly complex. To alleviate this problem, we consider reducing the degree of freedom in the model update direction. Specifically, we design a projection loss to limit the update direction of the local model to the low-dimensional subspace, thereby limiting the complexity of the model update, as defined below:

$$\mathcal{L}^{pro} = \left\| \texttt{Proj}\left(\Delta\tilde{\theta}^t_m\right) - \Delta\tilde{\theta}^t_m \right\|^2, \tag{7}$$

here, $\Delta\tilde{\theta}^t_m \in \mathbb{R}^K$ denotes the vectorized model update, calculated as $\texttt{reshape}(\theta^{t,e,i}_m - \theta^{t,0,0}_m)$, where K is the number of model parameters, and e and i represent the epoch and batch index, respectively. The projection $\texttt{Proj}(\Delta\tilde{\theta}^t_m) = PP^T\Delta\tilde{\theta}^t_m$, where $P \in \mathbb{R}^{K\times R}$, defines the subspace, which is initially randomized. During subspace update rounds (when $\texttt{mod}(t, s) == 0$, with s as the update interval), the subspace is refined by concatenating the updated parameter vectors and performing truncated singular value decomposition (*SVD*), as shown in Fig. 3. Specifically, the subspace P is concatenated with the parameter vector $\Delta\tilde{\theta}^t_m$ to form a $K \times (R+1)$ matrix, on which *SVD* is then applied.

$$[U, \Sigma, V] \leftarrow \texttt{SVD}_R\left(\left[P, \Delta\tilde{\theta}^t_m\right]\right), \tag{8}$$

where R is the rank of the subspace. After $\texttt{SVD}_R$ is completed, we set U to the local subspace $P \leftarrow U$.

The above strategy reduces the degree of freedom in the model update direction from K to R. However, in deep networks, the storage cost of subspaces can be significant. To mitigate this, we introduce a hierarchical constraint approach. Specifically, we reduce the subspace storage requirements by selectively applying subspace constraint to specific layers, allowing other layers to train normally. Since later layers in the model are more likely to capture localized distributions, we apply the constraint to the final layers of the network (see the projection layer in Fig. 2). We denote this vectorized projection layer as $\Delta\hat{\theta}^t_m$, replacing $\Delta\tilde{\theta}^t_m$ in Eq. (7) and (8). The modified projection loss

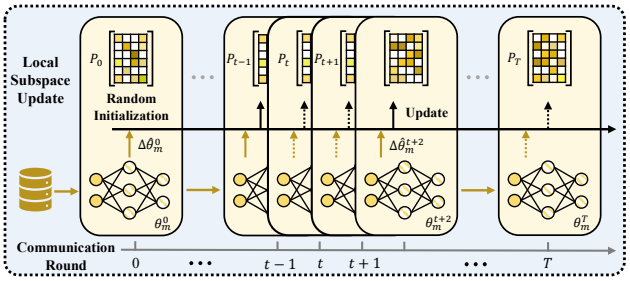


Figure 3. **Illustration** of the **local subspace update** with an interval $s = 3$. Solid arrows indicate rounds where the subspace is updated, while dotted arrows represent rounds without updates. The client updates P by applying $\texttt{SVD}_R$ on the previous subspace P and the parameter changes $\Delta\hat{\theta}_m^t$, as described in Eq. (8).

function is then defined as:

$$\mathcal{L}^{pro} = \left\| \texttt{Proj}\left(\Delta\hat{\theta}_m^t\right) - \Delta\hat{\theta}_m^t \right\|^2. \tag{9}$$

Additionally, since the subspace is updated dynamically, it contains information on previous model update trajectories. By constraining model updates within this space of prior trajectories, the approach may indirectly help to alleviate catastrophic forgetting in the model. During the local update phase, we optimize the following objectives:

$$\begin{aligned} \mathcal{L}_g &= \mathcal{L}_g^{ce} + \mathcal{L}_g^{kl}, \\ \mathcal{L}_l &= \mathcal{L}_l^{ce} + \mathcal{L}_l^{kl} + \mu \cdot \mathcal{L}^{pro}, \end{aligned} \tag{10}$$

where μ balances the impact of projection loss.

3.2.3. Discussion

Let L denote the number of parameters in $\Delta\hat{\theta}_m^t$. In FedSCE, the local computation process involves subspace extraction, where the main computational overhead comes from *SVD*, which has a time complexity of $\mathcal{O}(LR^2)$. This process can be accelerated using GPUs, making the computational cost of subspace extraction negligible compared to that of local training. As for space complexity, the subspace constraint reduces the L-dimensional space to the R-dimensional, bringing the space complexity down to $\mathcal{O}(K-L+R)$. Furthermore, driven by the existing analysis [4, 5, 42, 56, 61], We provide insights into generalization bounds and non-convex convergence rate of subspace constraint.

Assumption 1. *Local loss function is L_1-Lipschitz smooth,* $\|\nabla\mathcal{L}_m^{t_1,e} - \nabla\mathcal{L}_m^{t_2,e}\|_2 \leq L_1\|\theta_m^{t_1,e} - \theta_m^{t_2,e}\|_2, \forall t_1, t_2 > 0$.

Assumption 2. *The stochastic gradient for each client $g_m^{t,e} = \nabla\mathcal{L}(\theta_m^{t,e}, \xi_m^{t,e})$ is an unbiased estimator of the local gradient, $\mathbb{E}_{\xi_m^{t,e}\sim D_m}[g_m^{t,e}] = \nabla\mathcal{L}(\theta_m^{t,e})$, and its variance is bounded by σ^2, $\mathbb{E}[\|g_m^{t,e} - \nabla\mathcal{L}(\theta_m^{t,e})\|_2^2] \leq \sigma^2, \forall t, e$.*

Assumption 3. *The expectation of the stochastic gradient is bounded by G, $\mathbb{E}[\|g_m^{t,e}\|_2^2] \leq G^2, \forall t, e$.*

Proposition 1. *The squared projection error for $\Delta\tilde{\theta}_m^{t,e}$ is bounded, $\mathbb{E}[\|\texttt{Proj}(\Delta\tilde{\theta}_m^{t,e}) - \Delta\tilde{\theta}_m^{t,e}] \leq \rho_R\eta^2e^2G^2, \rho_R = \frac{\sum_{r=R+1}^{K}\sigma_r^2}{\sum_{r=1}^{K}\sigma_r^2}, \forall t, e$. σ_r denotes singular value.*

Theorem 1. *Consider a virtual global data domain $\mathcal{T}$ and a local data domain $\mathcal{T}_m$. Define $\mathcal{T} = \langle\mathcal{D}, c^*\rangle$ with $|\hat{\mathcal{D}}| = n$ and $\mathcal{T}_m = \langle\mathcal{D}_m, c^*\rangle$, where $\mathcal{D}, \mathcal{D}_m \subseteq \mathcal{X}$, and $c^* : \mathcal{X} \mapsto \mathcal{Y}$ is a ground-truth labeling function. Let $\mathcal{G} : \mathcal{X} \mapsto \mathcal{Z}$ be a feature extraction function. Suppose $\mathcal{H}$ represents a hypothesis space constrained by subspace with VC dimension d', and $h : \mathcal{Z} \mapsto \mathcal{Y}$ for each $h \in \mathcal{H}$. Then, with probability at least $1-\delta$, the following holds for any $h \in \mathcal{H}$:*

$$\begin{aligned} \mathcal{L}_{\mathcal{T}_m}(h) \leq \hat{\mathcal{L}}_{\mathcal{T}}(h) + \sqrt{\frac{4}{n}\left(d'\log\frac{2en}{d'} + \log\frac{4}{\delta}\right)} \\ + d_{\mathcal{H}\Delta\mathcal{H}}(\tilde{\mathcal{D}}, \tilde{\mathcal{D}}_m) + \lambda_m, \end{aligned} \tag{11}$$

where $d' < d < 2n$, d represents the VC dimension, which quantifies the model's complexity by indicating the maximum number of points it can shatter. $\lambda_m := \min_h(\mathcal{L}_{\mathcal{T}_m}(h) + \mathcal{L}_{\mathcal{T}}(h))$ denotes an oracle performance, e is the base of the natural logarithm, $\tilde{\mathcal{D}}, \tilde{\mathcal{D}}_m$ are the induced distributions of $\mathcal{D}, \mathcal{D}_m$ under $\mathcal{G}$. $d_{\mathcal{H}\Delta\mathcal{H}}(\tilde{\mathcal{D}}_m, \tilde{\mathcal{D}})$ denotes the divergence measured over a symmetric-difference hypothesis space [5]. $\hat{\mathcal{L}}_{\mathcal{T}}(h)$ is the empirical risk on $\mathcal{T}$.

Theorem 2. *Assuming that Assumption 1 to 3 and Proposition 1 hold, and $\mathcal{L}_m^*$ denotes the local optimum, the following inequality is valid for any client and any $\epsilon > 0$:*

$$\begin{aligned} \frac{1}{TE}\sum_{t=0}^{T-1}\sum_{e=0}^{E-1}\mathbb{E}[\|\nabla\mathcal{L}_m^{t,e}\|_2^2] \leq \frac{L_1\eta\sigma^2 + 2\mu\rho_R\eta EG^2}{2 - L_1\eta} \\ + \frac{2(\mathcal{L}_m^0 - \mathcal{L}_m^*)}{TE\eta(2 - L_1\eta)} < \epsilon. \end{aligned} \tag{12}$$

The *VC* dimension increases with space complexity [3, 40], imposing subspace constraint reduces the *VC* dimension and decreases the generalization gap. When $\eta < 2\epsilon/(L(\epsilon+\sigma^2) + 2\mu\rho_R EG^2)$, the model with subspace constraint can converge at a speed of $\mathcal{O}(1/T)$. See the Appendix for proof.

3.3. Auxiliary Model Contribution Estimates

The global model is aggregated on the server side to collect insights from all clients to guide local training. If the global model captures sufficient information from a client, its updates should show minimal differences (*e.g.*, model convergence). We assume that significant changes in the global model during local training indicate insufficient adaptation to the local data distribution. In such cases, the global model from that client should be given a higher aggregate weight to enhance the injection of client-specific information.

To quantify the local update distance of the global model, we measure the update shift of the global model in both *parameter* and *feature* spaces. Specifically, we calculate the distance between the locally updated global model $\theta_{g,m}^t$ and the pre-update model θ_g^t in these spaces:

$$F_{m,1} = \|\theta_{g,m}^t - \theta_g^t\|_2, \quad F_{m,2} = \|z_{g,m} - z_g\|_2, \tag{13}$$

where $z_{g,m} = g(x, \theta_{g,m}^t)$ and $z_g = g(x, \theta_g^t)$. Since the scales of parameter and feature spaces are different, we can-

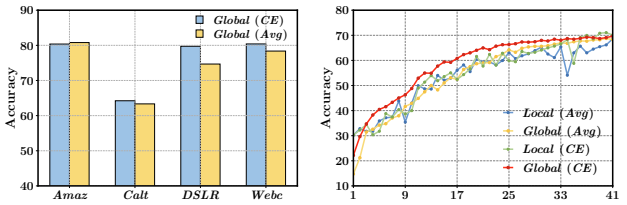
not directly combine them. Therefore, we first normalize the distances in each space independently as follows:

$$F'_{m,1} = \frac{F_{m,1}}{\sum_{m=1}^{M} F_{m,1}}, \quad F'_{m,2} = \frac{F_{m,2}}{\sum_{m=1}^{M} F_{m,2}}. \tag{14}$$

Finally, we combine $F'_{m,1}$ and $F'_{m,2}$ to calculate F_m and then normalize it to obtain the aggregate weight W_m:

$$W_m = \frac{F_m}{\sum_{m=1}^{M} F_m}, \tag{15}$$

where $F_m = F'_{m,1} + F'_{m,2}$. The weights assigned to auxiliary models from different clients are dynamically adjusted based on their contributions. This adaptive weighting strategy emphasizes insights from clients whose information is underrepresented in the auxiliary model. We validated this approach using a toy *CNN* [55] as the auxiliary model, with the accuracy results shown in Fig. 4.



(a) Accuracy of the auxiliary model. (b) Accuracy Curve

Figure 4. **Motivation Illustration** of Auxiliary Model Contribution Estimates. As shown in Fig. 4b, the accuracy of the auxiliary model using **CE** surpasses that of both the local model and the auxiliary model using **Average**.

4. Experiments

4.1. Experimental Setup

Datasets. This study focuses on image classification tasks across various scenarios. For the **feature-skew** setting, we utilize two benchmark datasets: *Digit5* and *Office10*. *Digit5* [60] is a standard benchmark for digit recognition, containing five subsets: *SVHN*, *USPS*, *SynthDigits (SYN)*, *MNIST-M (MNTM)*, and *MNIST (MNT)*. *Office10* [12] is another benchmark dataset, consisting of four subsets: *Amazon (Amaz)*, *Caltech (Calt)*, *DSLR*, and *WebCam (Webc)*.

For **label-skew** scenarios, we employ three popular datasets: *Cifar10* [24], *Cifar100* [24], and *Tiny-Imagenet* [25]. To simulate label skew, we apply two methods. The first involves a pathological heterogeneity setting, where we randomly sample 2, 10, or 20 classes from the available 10, 100, or 200 categories, respectively, for each client. In the second method, we use the Dirichlet distribution (Dir_β) to create heterogeneous data partitions [21].

Model. We consider five heterogeneous model architectures: *ResNet18* [15], *ResNet34* [15], *ResNet50* [15], *ViT-B/16* [8], and *ViT-B/32* [8]. For M clients, we assign each client the $(m \bmod 5)$-th model in sequence. To facilitate

Algorithm 1: FedSCE

Input: Communication rounds T, local epochs E, clients number M, private data $D_m(x, y)$, coefficient μ.

for $t = 0, 1, ..., T-1$ **do**
  /* Client Side: */
  for $m = 1, 2, ..., M$ ***in parallel*** **do**
    $\theta^t_{g,m}, F_{m,1}, F_{m,2} \leftarrow$ **LocalUpdating**(θ^t_g, t)
  /* Server Side: */
  Normalize $F_{m,1}$ and $F_{m,2}$ by Eq. (14).
  Calculate W_m by Eq. (15).
  $\theta^{t+1}_g \leftarrow \sum_{m=1}^{M} W_m \cdot \theta^t_{g,m}$

LocalUpdating(θ^t_g, t)**:**
$\theta^t_{g,m} \leftarrow \theta^t_g$ // Distribute Auxiliary Model
for $e = 0, 1, ..., E-1$ **do**
  for $(x_i, y_i) \in D_m$ **do**
    $q_{l,i} \leftarrow f(x_i, \theta^t_m), q_{g,i} \leftarrow f(x_i, \theta^t_{g,m})$
    /* Local Deep Mutual Learning */
    /* Sec. 3.2.1 */
    $\mathcal{L}^{ce}_l \leftarrow (q_{l,i}, y_i), \mathcal{L}^{ce}_g \leftarrow (q_{g,i}, y_i)$ in Eq. (4).
    $\mathcal{L}^{kl}_l \leftarrow (q_{g,i}, q_{l,i}), \mathcal{L}^{kl}_g \leftarrow (q_{l,i}, q_{g,i})$ in Eq. (6).
    /* Local Subspace Learning */
    /* Sec. 3.2.2 */
    $\mathcal{L}^{pro} \leftarrow (\Delta\hat{\theta}^t_m, P)$ in Eq. (7).
    $\mathcal{L}_g \leftarrow \mathcal{L}^{ce}_g + \mathcal{L}^{kl}_g, \mathcal{L}_l \leftarrow \mathcal{L}^{ce}_l + \mathcal{L}^{kl}_l + \mu \cdot \mathcal{L}^{pro}$
    $\theta^t_m \leftarrow \theta^t_m - \eta\nabla\mathcal{L}_l, \theta^t_{g,m} \leftarrow \theta^t_{g,m} - \eta\nabla\mathcal{L}_g$
/* Auxiliary Model Contribution Estimates */
$F_{m,1} \leftarrow \|\theta^t_{g,m} - \theta^t_g\|_2, F_{m,2} \leftarrow 0$ in Eq. (13).
for $(x_i, y_i) \in D_m$ **do**
  $z^i_{g,m} \leftarrow g(x_i, \theta^t_{g,m}), z^i_g \leftarrow g(x_i, \theta^t_g)$
  $F_{m,2} \leftarrow F_{m,2} + \|z^i_{g,m} - z^i_g\|^2_2$ in Eq. (13).
$F_{m,2} \leftarrow \sqrt{F_{m,2}}$.
/* Local Subspace Update */
if $\mathrm{mod}(t, s) == 0$ **then**
  $P \leftarrow (P, \Delta\hat{\theta}^t_m)$ in Eq. (8).
return $\theta^t_{g,m}, F_{m,1}, F_{m,2}$.

communication between the clients, we select the smallest model as the default auxiliary model.

Baselines. We compare FedSCE with several popular state-of-the-art HFL methods, including FML [39], FedKD [48], FedDistill [19], FedProto [42], and FedTGP [57]. We also evaluate a baseline with only local updates, termed SOLO.

Evaluation Metrics. We evaluate performance using average accuracy at both the data level (***D.Avg***) and client level (***C.Avg***), with *D.Avg* as the default. Specifically, N^{corr}_m represents the number of correctly classified samples per client, and accuracy is calculated as follows:

$$D.Avg = \frac{\sum_{m=1}^{M} N^{corr}_m}{\sum_{m=1}^{M} N_m}, \quad C.Avg = \frac{\sum_{m=1}^{M} N^{corr}_m / N_m}{M}. \tag{16}$$

Implementation Details. For all datasets, we use a batch

Method	*Office10*						*Digit5*						
	Amaz	*Calt*	*DSLR*	*Webc*	***C.Avg***	***D.Avg***	*SYN*	*USPS*	*SVHN*	*MNTM*	*MNT*	***C.Avg***	***D.Avg***
SOLO	80.79	61.03	82.27	90.54	78.66	73.26	92.81	99.83	76.36	86.72	98.27	90.81	90.41
FML [39]	78.91	59.96	78.48	91.89	77.31	72.00	92.94	99.88	75.67	87.45	98.28	90.85	90.45
FedKD [48]	77.45	58.36	79.74	93.24	77.22	70.98	96.41	99.78	85.60	89.08	98.50	93.88	93.62
FedDistill [19]	79.54	60.14	82.27	91.89	78.49	72.56	93.25	99.85	77.62	86.98	98.21	91.19	90.81
FedProto [42]	77.24	56.40	73.41	95.94	75.75	69.95	92.69	99.79	75.29	86.77	98.02	90.52	90.11
FedTGP [57]	78.07	60.85	68.35	93.24	75.13	71.61	94.34	99.77	77.88	87.37	98.01	91.48	91.11
FedSCE	84.55	64.59	79.74	92.57	**80.36**	**76.34**	96.48	99.82	84.87	89.90	98.69	**93.95**	**93.69**

Table 2. Comparison with other state-of-the-art algorithms in **feature-skew** scenarios. The highest average accuracy is marked in **bold**, while the highest accuracy for each client is underlined. Please see details in Sec. 4.2.

Method	*Cifar10*				*Cifar100*				*Tiny-Imagenet*			
	Pat	$Dir_{0.1}$	$Dir_{0.3}$	$Dir_{0.5}$	Pat	$Dir_{0.01}$	$Dir_{0.1}$	$Dir_{0.5}$	Pat	$Dir_{0.01}$	$Dir_{0.1}$	$Dir_{0.5}$
SOLO	82.29	84.27	68.80	57.69	52.51	62.70	39.12	22.04	**30.76**	43.08	28.40	14.80
FML [39]	82.44	84.51	69.74	58.46	52.19	62.03	39.11	22.92	30.20	42.76	28.31	14.53
FedKD [48]	84.47	86.29	71.77	61.39	52.71	63.73	40.37	23.60	30.46	43.12	**28.42**	15.26
FedDistill [19]	82.18	84.07	68.93	57.82	52.60	63.04	38.80	22.92	30.26	42.51	27.72	14.70
FedProto [42]	70.52	69.22	60.91	52.05	49.65	52.92	29.00	17.73	25.73	30.49	15.13	10.62
FedTGP [57]	80.33	83.67	68.61	57.80	50.48	62.68	39.97	22.91	26.20	34.50	20.51	11.10
FedSCE	**84.73**	**86.74**	**72.86**	**62.93**	**54.30**	**64.37**	**41.05**	**23.91**	30.49	**43.61**	28.38	**15.94**

Table 3. Comparison with other state-of-the-art algorithms in **label-skew** scenarios. where Pat stands for pathological heterogeneity. Please see details in Sec. 4.2.

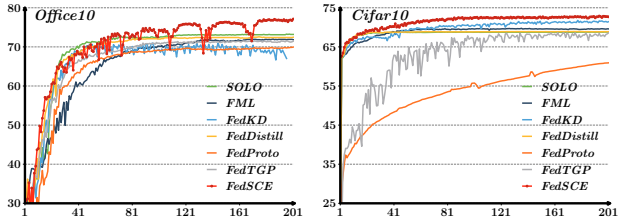
size of 64 for training, except for *Tiny-Imagenet*, where a batch size of 256 is applied. We use the SGD optimizer with a learning rate of 0.01. We set the number of local epochs to 5 and the number of clients to 20. We conduct 200 communication rounds, beyond which most methods have little or no accuracy improvement. Additionally, the training and test datasets are split with a ratio of 0.75 and 0.25, respectively. All experiments are conducted on an NVIDIA GeForce RTX 3090 GPU. Please refer to the Appendix for more results and details of baselines.

4.2. Comparison to State-of-the-Arts

Heterogeneous scenarios. We conducted an extensive comparison of FedSCE with several state-of-the-art methods across various heterogeneous scenarios, with the results presented in Tab. 2 and Tab. 3. For these experiments, we set $\mu = 5$, $p = 5$, and $R = 3$. Notably, in the feature-skew scenario, our approach achieved higher average test accuracy compared to other algorithms. Specifically, in *Office10*, FedSCE outperformed the best baseline test accuracy by 3.08% at the data level and 1.70% at the client level.

In the label-skew scenario, we simulated different levels of heterogeneity. Due to the limited number of categories in *Cifar10*, setting $\beta = 0.01$ was overly extreme, so we adjusted it to $\beta = 0.3$. As shown in Tab. 3, FedSCE continued to achieve top performance in most cases.

Local Epoch. Tab. 6 shows the impact of varying E on FedSCE. With $E = 1$, FedSCE sees limited improvement in average accuracy, but achieves gains of 3.31% and 1.91% at $E = 10$ and $E = 20$, respectively. Smaller E reduces local overfitting through frequent communication, though more rounds are needed for comparable accuracy. Larger E increases the risk of overfitting and catastrophic forgetting, where subspace learning helps mitigate these risks.



(a) Accuracy Curve on ***Office10***. (b) Accuracy Curve on ***Cifar10***.

Figure 5. **Convergence comparison** of average accuracy of different algorithms under feature-skew (*Office10*) and label-skew (*Cifar10*) scenarios. Please see details in Sec. 4.2.

Convergence Comparison. The convergence curves are presented in Fig. 5. As shown, FedSCE exhibits strong performance in both feature-skew and label-skew scenarios, achieving rapid convergence and high accuracy.

t-SNE Visualization. Fig. 7 displays the t-SNE visualizations of FedSCE across various communication rounds, illustrating that FedSCE progressively learns a more generalized decision boundary as training continues.

4.3. Ablation Study

Influence of Components. We assessed the impact of key components: local subspace learning (LSL), local deep mu-

tual learning (DML), and auxiliary model contribution estimation (CE) to demonstrate their individual effectiveness.

1) Effectiveness of LSL. As shown in Tab. 4, adding projection constraint to local model updates, even without any federated techniques, increased average accuracy from 78.66% and 73.26% to 79.93% and 73.90%. This result shows that restricting certain layers to a subspace can improve performance.

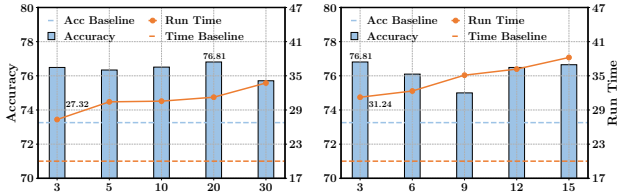
2) Effectiveness of DML. Local deep mutual learning improves accuracy from 73.90% to 75.31% by enabling the auxiliary model to integrate information from other clients, enhancing the local model generalization and supporting collaborative training.

3) Effectiveness of CE. As shown in Tab. 4, using the contribution estimation strategy to adjust the weight of the auxiliary model, the average test accuracy increased by 1.02% and 1.03%. This strategy enables the auxiliary model to better focus on the information it lacks.

Module			*Test Accuracy*					
LSL	DML	CE	*Amaz*	*Calt*	*DSLR*	*Webc*	***C.Avg***	***D.Avg***
			80.79	61.03	82.27	90.54	78.66	73.26
✓			80.58	61.74	84.81	92.57	79.93	73.90
✓	✓		83.29	63.37	78.48	91.89	79.34	75.31
✓	✓	✓	84.55	64.59	79.74	92.57	**80.36**	**76.34**

Table 4. **Ablation** experiments for different components. Please see details in Sec. 4.3.

Influence of Hyperparameters. We present the performance impact of the subspace hyperparameters p and R in Fig. 4.3. In Tab. 5, we evaluate the effect of μ in both feature-skew and label-skew scenarios.



(a) The number of layers p (b) The rank of subspace R

Figure 6. **Hyperparameter Study:** We examine the rank R of the subspace and the number of layers p with projection constraint. Further details are provided in Sec. 4.3.

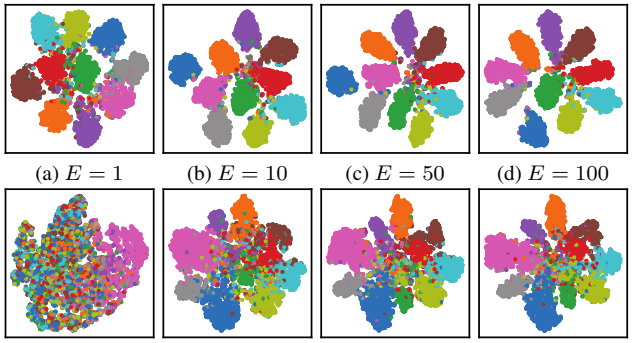
	Office10				*Cifar10*			
μ	0.1	1	5	10	0.1	1	5	10
Acc	76.97	76.81	76.34	76.96	72.89	72.67	72.86	72.80

Table 5. **Hyperparameter Study:** Test accuracy (%) of FedSCE under **different μ** values for the *Office10* and *Cifar10* datasets. Please refer to Sec. 4.3 for a detailed discussion.

Tab. 5 presents the impact of different μ values on *Office10* and *Cifar10*. The results indicate that FedSCE performance remains stable across μ settings, consistently surpassing the best baselines of 73.26% and 71.77%. Fig. 6a demonstrates the effect of the number of constraint layers p on accuracy and runtime, with the highest accuracy of 76.81% at $p = 20$. Choosing the number of layers carefully is essential for balancing performance and computational cost. Fig. 6b illustrates the influence of subspace rank R at $p = 20$. While a higher R provides more flexibility, it does not consistently improve accuracy and may increase runtime, as it can counteract the goal of limiting update directions, leading to diminishing performance gains.

Method	*E=1*		*E=10*		*E=20*	
	C.Avg	***D.Avg***	***C.Avg***	***D.Avg***	***C.Avg***	***D.Avg***
SOLO	72.57	69.18	79.01	73.89	79.41	74.44
FML [39]	67.06	62.53	77.91	72.31	77.27	72.00
FedKD [48]	73.48	68.92	77.21	71.92	78.56	73.50
FedDistill [19]	**74.96**	70.21	79.03	73.50	79.71	74.44
FedProto [42]	66.07	64.74	76.44	70.42	78.68	72.94
FedTGP [57]	67.43	66.08	76.97	73.89	79.11	76.41
FedSCE	72.70	**70.79**	**80.88**	**77.20**	**82.18**	**78.31**

Table 6. Average test accuracy (%) of **different local epochs** on *Office10*. Please see details in Sec. 4.2.



(a) $E = 1$ (b) $E = 10$ (c) $E = 50$ (d) $E = 100$

(e) $E = 1$ (f) $E = 10$ (g) $E = 50$ (h) $E = 100$

Figure 7. **t-SNE visualization** for *SYN* (top) and *SVHN* (bottom) clients in FedSCE at different rounds. See Details in Sec. 4.2.

5. Conclusion

In this paper, we introduce FedSCE to tackle local overfitting and model aggregation bias challenges in heterogeneous federated learning. During local training, FedSCE reduces the degree of freedom of model update and improves generalization ability by constraining the specific layer of local model update in a low-dimensional subspace. FedSCE dynamically updates the subspace to ensure the injection of the latest information. Furthermore, FedSCE evaluates the contributions of different clients based on the update distance of the auxiliary model in feature space and parameter space, and then adaptively aggregates the auxiliary model. Comprehensive experiments demonstrate that FedSCE outperforms several state-of-the-art methods in feature-skewed and label-skewed scenarios.

Acknowledgements. This work was supported by the Key Program for International Cooperation of Ministry of Science and Technology of China (No.2024YFE0100700).

References

[1] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, 2021. 2

[2] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019. 3

[3] Peter L Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019. 5

[4] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19, 2006. 5

[5] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. *Advances in neural information processing systems*, 20, 2007. 5

[6] Yuhang Chen, Wenke Huang, and Mang Ye. Fair federated learning under domain skew with local consistency and domain diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12077–12086, 2024. 3

[7] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International conference on machine learning*, pages 2089–2099. PMLR, 2021. 3

[8] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 6

[9] Xiuwen Fang and Mang Ye. Robust federated learning with noisy and heterogeneous clients. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10072–10081, 2022. 2

[10] Xiuwen Fang, Mang Ye, and Xiyuan Yang. Robust heterogeneous federated learning under data corruption. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5020–5030, 2023. 2

[11] 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. 2

[12] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2066–2073. IEEE, 2012. 6

[13] Frithjof Gressmann, Zach Eaton-Rosen, and Carlo Luschi. Improving neural network training in low dimensional random bases. *Advances in Neural Information Processing Systems*, 33:12140–12150, 2020. 3

[14] Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018. 3

[15] 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*, pages 770–778, 2016. 6

[16] Geoffrey Hinton. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[17] Wenke Huang, Mang Ye, and Bo Du. Learn from others and be yourself in heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10143–10153, 2022. 1, 2, 3

[18] Wenke Huang, Mang Ye, Zekun Shi, He Li, and Bo Du. Rethinking federated learning with domain shift: A prototype view. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16312–16322. IEEE, 2023. 1

[19] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018. 3, 6, 7, 8

[20] Meirui Jiang, Holger R Roth, Wenqi Li, Dong Yang, Can Zhao, Vishwesh Nath, Daguang Xu, Qi Dou, and Ziyue Xu. Fair federated medical image segmentation via client contribution estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16302–16311, 2023. 2

[21] Norman Lloyd Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. *Continuous multivariate distributions*. Wiley New York, 1972. 6

[22] 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. 1, 2

[23] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 2

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[25] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 6

[26] Gihun Lee, Minchan Jeong, Yongjin Shin, Sangmin Bae, and Se-Young Yun. Preservation of the global knowledge by not-true distillation in federated learning. *Advances in Neural Information Processing Systems*, 35:38461–38474, 2022. 2

[27] Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*, 2018. 3

[28] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019. 1, 3

[29] 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. 2

[30] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3347–3366, 2021. 1

[31] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020. 1

[32] 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. 1, 2

[33] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International conference on machine learning*, pages 6357–6368. PMLR, 2021. 1, 3

[34] Tao Li, Zhehao Huang, Qinghua Tao, Yingwen Wu, and Xiaolin Huang. Trainable weight averaging: Efficient training by optimizing historical solutions. In *The Eleventh International Conference on Learning Representations*, 2022. 2, 3

[35] Tao Li, Lei Tan, Zhehao Huang, Qinghua Tao, Yipeng Liu, and Xiaolin Huang. Low dimensional trajectory hypothesis is true: Dnns can be trained in tiny subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3): 3411–3420, 2022. 2, 3

[36] Kangyang Luo, Xiang Li, Yunshi Lan, and Ming Gao. Gradma: A gradient-memory-based accelerated federated learning with alleviated catastrophic forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3708–3717, 2023. 2

[37] 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*, pages 1273–1282. PMLR, 2017. 2

[38] Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *IEEE Transactions on Signal Processing*, 70:1142–1154, 2022. 2

[39] Tao Shen, Jie Zhang, Xinkang Jia, Fengda Zhang, Gang Huang, Pan Zhou, Kun Kuang, Fei Wu, and Chao Wu. Federated mutual learning. *arXiv preprint arXiv:2006.16765*, 2020. 1, 2, 3, 6, 7, 8

[40] Eduardo D Sontag et al. Vc dimension of neural networks. *NATO ASI Series F Computer and Systems Sciences*, 168: 69–96, 1998. 5

[41] Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in neural information processing systems*, 33:21394–21405, 2020. 3

[42] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8432–8440, 2022. 1, 3, 5, 6, 7, 8

[43] Yue Tan, Guodong Long, Jie Ma, Lu Liu, Tianyi Zhou, and Jing Jiang. Federated learning from pre-trained models: A contrastive learning approach. *Advances in neural information processing systems*, 35:19332–19344, 2022. 1

[44] Mark Tuddenham, Adam Prügel-Bennett, and Jonathan Hare. Quasi-newton's method in the class gradient defined high-curvature subspace. *arXiv preprint arXiv:2012.01938*, 2020. 3

[45] Oriol Vinyals and Daniel Povey. Krylov subspace descent for deep learning. In *Artificial intelligence and statistics*, pages 1261–1268. PMLR, 2012. 3

[46] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020. 2

[47] Dingzhu Wen, Ki-Jun Jeon, and Kaibin Huang. Federated dropout—a simple approach for enabling federated learning on resource constrained devices. *IEEE wireless communications letters*, 11(5):923–927, 2022. 2

[48] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1): 2032, 2022. 2, 3, 4, 6, 7, 8

[49] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019. 1

[50] Xiyuan Yang, Wenke Huang, and Mang Ye. Dynamic personalized federated learning with adaptive differential privacy. *Advances in Neural Information Processing Systems*, 36:72181–72192, 2023. 1

[51] Xiyuan Yang, Wenke Huang, and Mang Ye. Fedas: Bridging inconsistency in personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11986–11995, 2024. 3

[52] Liping Yi, Gang Wang, Xiaoguang Liu, Zhuan Shi, and Han Yu. Fedgh: Heterogeneous federated learning with generalized global header. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 8686–8696, 2023. 1

[53] Jie Zhang, Zhiqi Li, Bo Li, Jianghe Xu, Shuang Wu, Shouhong Ding, and Chao Wu. Federated learning with label distribution skew via logits calibration. In *International Conference on Machine Learning*, pages 26311–26329. PMLR, 2022. 2

[54] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, Jian Cao, and Haibing Guan. Gpfl: Simultaneously learning global and personalized feature information for personalized federated learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5041–5051, 2023. 1

[55] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Fedala: Adaptive local aggregation for personalized federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11237–11244, 2023. 1, 6

[56] Jianqing Zhang, Yang Hua, Jian Cao, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Eliminating domain bias for federated learning in representation space. *Ad-*

vances in Neural Information Processing Systems, 36, 2024. 5

[57] Jianqing Zhang, Yang Liu, Yang Hua, and Jian Cao. Fedtgp: Trainable global prototypes with adaptive-margin-enhanced contrastive learning for data and model heterogeneity in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 16768–16776, 2024. 1, 3, 6, 7, 8

[58] Jianqing Zhang, Yang Liu, Yang Hua, and Jian Cao. An upload-efficient scheme for transferring knowledge from a server-side pre-trained generator to clients in heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12109–12119, 2024. 1

[59] Xu Zhang, Yinchuan Li, Wenpeng Li, Kaiyang Guo, and Yunfeng Shao. Personalized federated learning via variational bayesian inference. In *International Conference on Machine Learning*, pages 26293–26310. PMLR, 2022. 2

[60] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 561–578. Springer, 2020. 6

[61] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*, pages 12878–12889. PMLR, 2021. 5