This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

An Upload-Efficient Scheme for Transferring Knowledge From a Server-Side Pre-trained Generator to Clients in Heterogeneous Federated Learning

Jianqing Zhang[1]*, Yang Liu[2,3]†, Yang Hua[4], Jian Cao[1,5]†
[1]Shanghai Jiao Tong University [2]Institute for AI Industry Research (AIR), Tsinghua University
[3]Shanghai Artificial Intelligence Laboratory [4]Queen's University Belfast
[5]Shanghai Key Laboratory of Trusted Data Circulation and Governance in Web3
tsingz@sjtu.edu.cn, liuy03@air.tsinghua.edu.cn, y.hua@qub.ac.uk, cao-jian@sjtu.edu.cn

Abstract

*Heterogeneous Federated Learning (HtFL) enables collaborative learning on multiple clients with different model architectures while preserving privacy. Despite recent research progress, knowledge sharing in HtFL is still difficult due to data and model heterogeneity. To tackle this issue, we leverage the knowledge stored in public pre-trained generators and propose a new upload-efficient knowledge transfer scheme called Federated Knowledge-Transfer Loop (**FedKTL**). Our FedKTL can produce client-task-related prototypical image-vector pairs via the generator's inference on the server. With these pairs, each client can transfer pre-existing knowledge from the generator to its local model through an additional supervised local task. We conduct extensive experiments on four datasets under two types of data heterogeneity with 14 kinds of models including CNNs and ViTs. Results show that our upload-efficient FedKTL surpasses seven state-of-the-art methods by up to **7.31%** in accuracy. Moreover, our knowledge transfer scheme is applicable in scenarios with only one edge client. Code:* https://github.com/TsingZ0/FedKTL

1. Introduction

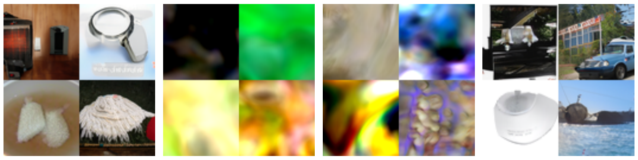
Recently, there has been a growing trend for companies to develop custom models tailored to their specific needs [3, 11, 15, 18, 50]. However, the problem of insufficient data has persistently plagued model training in specific fields, such as medicine [1, 4, 43]. Federated Learning (FL) is a popular approach to tackle this problem by training models collaboratively among multiple clients (*e.g.*, companies or edge devices) while preserving privacy on clients [19, 28]. Traditional FL (tFL) focuses on training a global model for all clients and is unable to fulfill clients' personalized needs due to data heterogeneity among clients [20, 29]. Consequently, personalized FL (pFL) has emerged as a solution to train customized models for each client [30, 58, 67, 69].

*Work done during internship at AIR.
†Corresponding authors.

However, most pFL methods still assume homogeneous client models [30, 67, 69], which may not adequately cater to the specific needs of companies and individuals [61]. Besides, as the size of the model increases, both tFL and pFL incur significant communication costs when transmitting model parameters [76]. Furthermore, exposing clients' model parameters also raises privacy and intellectual property (IP) concerns [27, 55, 63, 70]. Recently, Heterogeneous Federated Learning (HtFL) frameworks have been proposed to consider both data and model heterogeneity [52, 61]. It explores novel knowledge-sharing schemes that go beyond sharing the entire client models.

Most existing HtFL methods adopt knowledge distillation (KD) techniques [13] and design various knowledge-sharing frameworks based on a global dataset [36, 64], a global auxiliary model [57, 71], or global class-wise prototypes [52, 53, 70]. However, global datasets' availability and quality as well as their *relevance to clients' tasks* significantly impact the effectiveness of KD [65]. Directly replacing the global dataset with a pre-trained generator has a minimal impact since most generators are pre-trained to generate unlabeled data *within the domain of their pre-training data* [21, 22]. As for the global auxiliary model, it introduces a substantial communication overhead due to the need to transmit it in each communication iteration. Although sharing class-wise prototypes is communication-efficient, they can only carry limited global knowledge to clients, which is insufficient for clients' model training needs. Furthermore, the prototypes extracted by heterogeneous models are biased, hindering the attainment of uniformly separated global prototypes on the server [70].

Thus, we propose an upload-efficient knowledge transfer scheme called *Federated Knowledge-Transfer Loop (**FedKTL**)*, which takes advantage of the compactness of prototypes and the pre-existing knowledge from a server-side public pre-trained generator. FedKTL can (1) use the generator on the server to produce a handful of global prototypical image-vector pairs tailored to clients' tasks, and (2) transfer pre-existing common knowledge from the generator to each client model via an additional *supervised* local



(a) Valid vecs (b) Random vecs (c) Prototypes (d) Aligned vecs

Figure 1. The images (64 × 64) generated by StyleGAN-XL [48] with different kinds of inputs. "vecs" is short for vectors.

task using these image-vector pairs. We develop FedKTL by addressing the following three questions. **Q1**: *How to upload unbiased prototypes while maintaining upload efficiency?* **Q2 (the core challenge)**: *How to adapt any given pre-trained generator to clients' tasks without fine-tuning it?* **Q3**: *How to transfer the generator's knowledge to client models regardless of the semantics of the generated images?*

For **Q1**, inspired by FedETF [33], we replace each client's classifier with an ETF (equiangular tight frame) classifier [33, 59] to let clients generate unbiased prototypes. Then, we upload these unbiased prototypes to the server for efficiency. For **Q2**, we align the domain formed by prototypes with the generator's inherent valid latent domain to generate informative images, as *these two domains are not naturally aligned*. As shown in Fig. 1, the generator can generate clear images given valid vectors. However, it tends to generate blurry and uninformative images given invalid latent vectors (such as random vectors or prototypes). To generate prototype-induced clear images, we propose a *lightweight trainable feature transformer* on the server to convert prototypes to aligned vectors within the valid input domain, while preserving the class-wise discrimination relevant to clients' classification tasks. For **Q3**, we first aggregate aligned vectors for each class to obtain latent centroids and generate corresponding images to form image-vector pairs. Then we conduct an additional supervised local task to only enhance the client model's feature extraction ability using these pairs, thereby reducing the semantic relevance requirements between the generated images and local data.

We evaluate our FedKTL via extensive experiments on four datasets with two types of data heterogeneity and 14 model architectures using a StyleGAN [21–23, 48] or a Stable Diffusion [45] on the server. Our FedKTL can outperform seven state-of-the-art methods by at most **7.31%** in accuracy. We also show that FedKTL is upload-efficient and one prototypical image-vector pair per class is sufficient for knowledge transfer, which only demands minimal inference of the generator on the server in each iteration.

2. Related Work

2.1. Heterogeneous Federated Learning (HtFL)

HtFL offers the advantage of preserving both privacy and model IP while catering to personalized model architecture requirements [10, 52, 61]. In terms of the level of model heterogeneity, we classify existing HtFL methods into three categories: group heterogeneity, partial heterogeneity, and full heterogeneity.

Group-heterogeneity-based HtFL methods distribute multiple groups of homogeneous models to clients, considering their diverse communication and computing capabilities [8, 36]. They typically form groups by sampling submodels from a server model [8, 14, 56]. In this paper, we do not consider this kind of model heterogeneity due to IP protection concerns and client customization limitations.

Partial-heterogeneity-based HtFL methods, *e.g.*, LG-FedAvg [35], FedGen [75], and FedGH [61], allow the main parts of the clients' models to be heterogeneous but assume the remaining (small) parts to be homogeneous. However, clients can only access limited global knowledge through the small global part. Despite training a global representation generator, FedGen primarily utilizes it to introduce global knowledge for the small classifier rather than the remaining main part (*i.e.*, the feature extractor). Therefore, the data insufficiency problem still exists for the main part.

Full-heterogeneity-based HtFL methods do not impose restrictions on the architectures of client models. Classic KD-based HtFL approaches [26, 62] share model outputs on a global dataset. However, obtaining such a dataset can be difficult in practice [65]. Instead of relying on a global dataset, FML [49] and FedKD [57] utilize mutual distillation [73] between a small auxiliary global model and client models. However, in the early iterations when both the auxiliary model and client models have poor performance, there is a risk of transferring uninformative knowledge between each other [34]. Another approach is to share class prototypes, like FedDistill [17], FedProto [52], and FedPCL [53]. However, the phenomenon of classifier bias has been extensively observed in FL when dealing with heterogeneous data [33, 38]. The bias becomes more pronounced when both the models and the data exhibit heterogeneity, leading to biased prototypes, thereby posing challenges in aggregating class-wise global knowledge [70].

2.2. ETF Classifier

When training a model on balanced data reaches its terminal stage, the neural collapse [42] phenomenon occurs. In this phenomenon, prototypes and the classifier vectors converge to form a simplex ETF, where the vectors are normalized, and the pairwise angles between them are maximized and identical (balanced). Since a simplex ETF represents an ideal classifier, some centralized methods [59, 60] propose generating a random simplex ETF matrix to replace the original classifier and guiding the feature extractor training using the fixed ETF classifier in imbalanced scenarios. To address the data heterogeneity issue in FL, FedETF [33] also proposes to replace the original classifier for each client with a fixed ETF classifier. However, FedETF assumes the presence of homogeneous models and follows FedAvg to transfer global knowledge. Inspired by these methods, we utilize the ETF classifier to enable heterogeneous client models to generate unbiased prototypes and facilitate class-wise global knowledge aggregation on the server.

3. Method

3.1. Preliminaries

Several concepts in various generators, such as StyleGAN [21] and Stable Diffusion [45], share similarities when generating contents, despite potential differences in their nomenclature. Without loss of generality, we primarily focus on introducing the generator components based on StyleGAN's architecture here for convenience. Most existing StyleGANs contain two components: a mapping network G_m and a synthesis network G_s. The space formed by the latent vectors between G_m and G_s is called "$\mathcal{W}$ space".

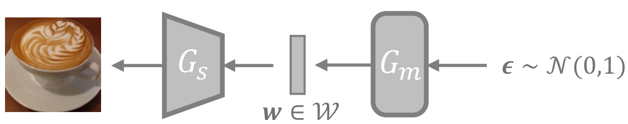


Figure 2. An illustration of the generating process (from right to left) when utilizing StyleGAN-XL as an example. The solid border of G_s and G_m means "with frozen parameters".

In Fig. 2, we show an example of the StyleGAN-XL [48] employed in our FedKTL. Given a vector $\boldsymbol{\epsilon}$ (typically a normally distributed noise vector) as the input, it transforms $\boldsymbol{\epsilon}$ to a latent vector $\boldsymbol{w} \in \mathbb{R}^H$ through G_m, *i.e.*, $\boldsymbol{w} = G_m(\boldsymbol{\epsilon}) \in \mathcal{W}$. Then, it generates an image I by further transforming $\boldsymbol{w}$ with G_s, *i.e.*, $I = G_s(\boldsymbol{w})$. $\boldsymbol{w}$ is the only factor that controls the content of I. While the valid vectors in $\mathcal{W}$ can produce clear and informative images, not all vectors in $\mathbb{R}^H$ are valid and possess the same capability.

3.2. Problem Statement

In HtFL, one server and N clients collaborate to train client models for a multi-classification task of C classes. Client i owns private data $\mathcal{D}_i$ and builds its model g_i (parameterized by $\boldsymbol{W}_i$) with a customized architecture. Formally, the objective is $\min_{\{\boldsymbol{W}_i\}_{i=1}^N} \sum_{i=1}^N \frac{n_i}{n} L_i(\boldsymbol{W}_i, \mathcal{D}_i)$, where $n_i = |\mathcal{D}_i|$, $n = \sum_{i=1}^N n_i$, and L_i is the local loss function.

3.3. Our FedKTL

3.3.1 Overview

In Fig. 3a, we illustrate six key steps of the knowledge-transfer loop in our proposed FedKTL framework. ① After local training, each client generates class-wise prototypes. ② Each client uploads prototypes to the server. ③ The server trains a feature transformer (denoted by F with parameter $\boldsymbol{W}_F$) to transform and align client prototypes to latent vectors. ④ With the trained F, the server first obtains the class-wise latent centroid $\bar{\mathcal{Q}}$, which is the averaged latent vectors within each class, and then generates images $\mathcal{D}_I$ by inputting $\bar{\mathcal{Q}}$ into G_s. ⑤ Each client downloads the prototypical image-vector pairs $\{\mathcal{D}_I, \bar{\mathcal{Q}}\}$ from the server. ⑥ Each client locally trains g_i and h'_i using $\mathcal{D}_i$, $\mathcal{D}_I$, and $\bar{\mathcal{Q}}$, where h'_i is an additional linear projection layer (parameterized by $\boldsymbol{W}_{h'_i}$) used to change the dimension of feature representations. Notice that $|\bar{\mathcal{Q}}| = |\mathcal{D}_I| = C \ll |\mathcal{D}_i|$.

3.3.2 ETF Classifier and Prototype Generation

The local loss L_i consists of two components: L_i^A, which is the loss corresponding to $\mathcal{D}_i$, and L_i^M, which is the loss for knowledge transfer using $\mathcal{D}_I$ and $\bar{\mathcal{Q}}$. For clarity, we only describe L_i^A here and leave the details of L_i^M to Sec. 3.3.4.

To address the biased prototype issue, inspired by FedETF [33], we replace the original classifiers of given model architectures with identical ETF classifiers and add a linear projection layer (one Fully Connected (FC) layer) h_i to the feature extractor f_i. In this way, we encourage each local model g_i to generate unbiased prototypes that are aligned with the globally identical ETF classifier vectors. f_i and h_i have parameters $\boldsymbol{W}_{f_i}$ and $\boldsymbol{W}_{h_i}$, respectively. Thus, we have $g_i = h_i \circ f_i$ and $\boldsymbol{W}_i = \{\boldsymbol{W}_{f_i}, \boldsymbol{W}_{h_i}\}$.

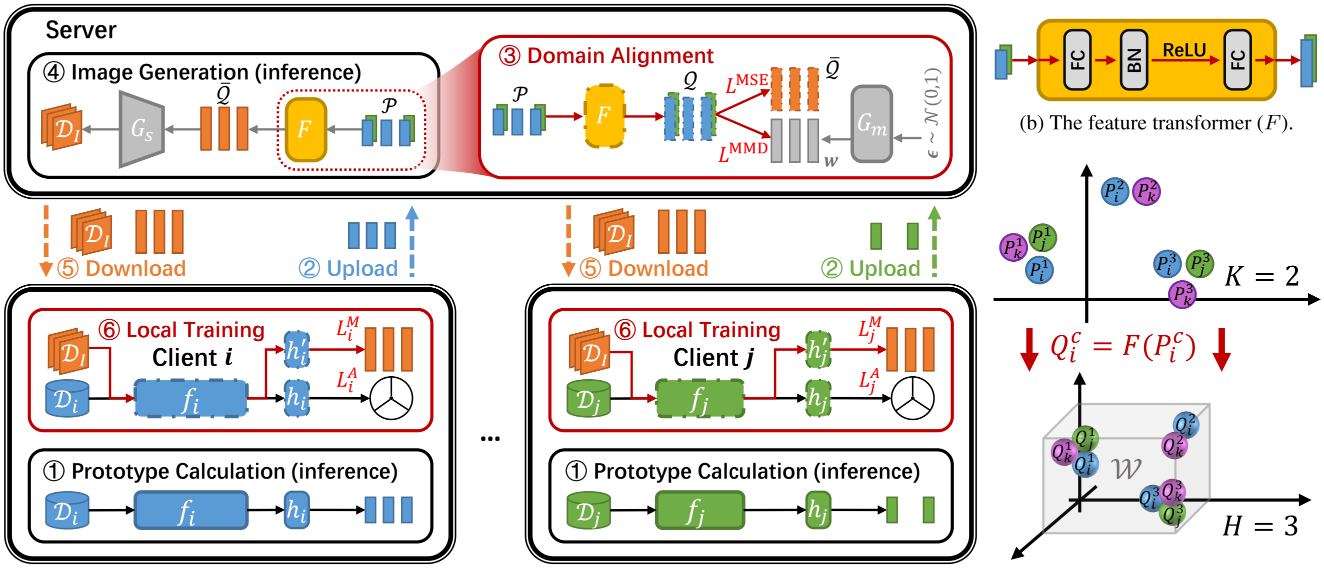
Specifically, we first synthesize a simplex ETF $\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_C]$, where $\boldsymbol{V} = \sqrt{\frac{C}{C-1}}\boldsymbol{U}(\boldsymbol{I}_C - \frac{1}{C}\boldsymbol{1}_C\boldsymbol{1}_C^T) \in \mathbb{R}^{K\times C}$ and the dimension of the ETF space $K \geq C-1$. $\forall c \in [C], \boldsymbol{v}_c \in \mathbb{R}^K$ and the L_2-norm $||\boldsymbol{v}_c||_2 = 1$. $\boldsymbol{U}$ allows a rotation, $\boldsymbol{U} \in \mathbb{R}^{K\times C}$, $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_C$, $\boldsymbol{I}_C$ is an identity matrix, and $\boldsymbol{1}_C$ is a vector with all ones. Besides, $\forall c_1, c_2 \in [C]$ and $c_1 \neq c_2$, we have $\cos\theta = -\frac{1}{C-1}$, where θ is the angle between $\boldsymbol{v}_{c_1}$ and $\boldsymbol{v}_{c_2}$. Furthermore, θ is also the maximum angle to equally separate C vectors [33, 42, 59]. Then, we distribute $\boldsymbol{V}$ to all clients.

Next, for a given input $\boldsymbol{x}$ on client i, we compute logits by measuring the cosine similarity [40] between $g_i(\boldsymbol{x})$ and each vector in $\boldsymbol{V}$. As the ArcFace loss [7] is popular for enhancing supervised learning when using cosine similarity for classification, we apply it during local training:

$$L_i^A = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i} - \log \frac{e^{s(\cos(\theta_y + m))}}{e^{s(\cos(\theta_y + m))} + \sum_{c=1,c\neq y}^C e^{s\cos\theta_c}}, \tag{1}$$

where θ_y is the angle between $g_i(\boldsymbol{x})$ and $\boldsymbol{v}_y$, s and m are the re-scale and additive hyperparameters [7], respectively.

After local training, we fix g_i and collect prototypes $\mathcal{P}_i = \{\boldsymbol{P}_i^c\}_{c\in\mathcal{C}_i}$ in the ETF space, where $\mathcal{C}_i$ is a set of class labels on client i. Formally, $\boldsymbol{P}_i^c = \mathbb{E}_{(\boldsymbol{x},c)\sim\mathcal{D}_i^c}\, g_i(\boldsymbol{x}) \in \mathbb{R}^K$, where $\mathcal{D}_i^c$ refers to the subset of $\mathcal{D}_i$ containing data points



(a) The framework of our FedKTL in one communication iteration for HtFL.

(b) The feature transformer (F).

(c) A domain alignment example.

Figure 3. An example of our FedKTL for a 3-class classification task. (a) Rounded and slender rectangles denote models and representations, respectively; dash-dotted and solid borders denote updating and frozen components, respectively; the segmented circle represents the ETF classifier. (b) The feature transformer (F) contains two FC layers and one Batch Normalization [16] (BN) layer. (c) An example of the domain alignment step with $K = 2$ and $H = 3$; one cluster represents one class. *Best viewed in color.*

belonging to class c. Uploading $\mathcal{P}_i$ to the server only requires $|\mathcal{C}_i| \times K$ elements to communicate, where $|\mathcal{C}_i| \leq C$.

3.3.3 Domain Alignment and Image Generation

For simplicity, we assume full client participation here, although FedKTL supports partial participation. With clients' prototypes $\mathcal{P} = \{\boldsymbol{P}_i^c\}_{i\in[N],c\in\mathcal{C}_i}$ on the server, we devise a trainable feature transformer F (see Fig. 3b) to convert $\mathcal{P}$ into valid latent vectors $\mathcal{Q} = \{\boldsymbol{Q}_i^c\}_{i\in[N],c\in\mathcal{C}_i}$, where $\boldsymbol{Q}_i^c = F(\boldsymbol{P}_i^c) \in \mathbb{R}^H$, in $\mathcal{W}$ space. To maintain $\mathcal{Q}$'s relationship with clients' classification tasks, we first preserve $\mathcal{Q}$'s class-wise discrimination by training F with

$$L^{\mathrm{MSE}} = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{|\mathcal{M}_c|}\sum_{i\in\mathcal{M}_c}\ell(F(\boldsymbol{P}_i^c), \boldsymbol{Q}^c), \tag{2}$$

where $\mathcal{M}_c$ is the client set owning class c, the global class-wise centroid $\boldsymbol{Q}^c = \frac{1}{|\mathcal{M}_c|}\sum_{j\in\mathcal{M}_c} F(\boldsymbol{P}_j^c)$, and ℓ is the Mean Squared Error (MSE) [54] between two vectors. Then, we use the Maximum Mean Discrepancy (MMD) loss [31] to align the domain formed by $\mathcal{Q}$ with the valid input domain of G_s in $\mathcal{W}$:

$$L^{\mathrm{MMD}} = ||\mathbb{E}_{\boldsymbol{Q}\sim\mathcal{Q}}\ \phi(\boldsymbol{Q}) - \mathbb{E}_{\boldsymbol{w}\sim\mathcal{W}}\ \phi(\boldsymbol{w})||^2_{\mathcal{H}}. \tag{3}$$

$\boldsymbol{w}$ is randomly sampled using G_m, ϕ is a feature map induced by a kernel function κ, *i.e.*, $\kappa(\boldsymbol{a}, \boldsymbol{b}) = \langle\phi(\boldsymbol{a}), \phi(\boldsymbol{b})\rangle$, and $\mathcal{H}$ is a reproducing kernel Hilbert space [31, 37]. We combine these two losses to form the server loss $L = L^{\mathrm{MMD}} + \lambda L^{\mathrm{MSE}}$, where λ is a hyper-parameter. We show a domain alignment example in Fig. 3c.

After training F on the server, we generate one image per class by inputting global centroids $\bar{\mathcal{Q}} = \{\boldsymbol{Q}^c\}_{c=1}^C$ into G_s, so only C times of inference for G_s is required in each iteration. Formally, we generate $\mathcal{D}_I = \{I^c\}_{c=1}^C$, where $I^c = G_s(\boldsymbol{Q}^c)$, and distribute paired class-wise $\mathcal{D}_I$ and $\bar{\mathcal{Q}}$ to clients for additional local supervised learning.

3.3.4 Transferring Pre-existing Global Knowledge

Then, client i conducts local training with the integrated local loss $L_i = L_i^A + \mu L_i^M$, where μ is a hyper-parameter. L_i^M is the additional supervised task to transfer pre-existing knowledge from the generator and inject common and shared information into the feature extractor. Formally,

$$L_i^M = \frac{1}{C}\sum_{c=1}^{C}\ell(h_i'(f_i(I^c)), \boldsymbol{Q}^c), \tag{4}$$

where h_i' is a linear projection layer that outputs vectors with dimension H. Since $\mathcal{D}_I$ and $\bar{\mathcal{Q}}$ are the output-input pairs of G_s and serve as the input-output pairs for $h_i' \circ f_i$, we can transfer common knowledge from G_s to $h_i' \circ f_i$. Since h_i' is mainly used for dimension transformation rather than knowledge learning, we initialize $\boldsymbol{W}_{h_i'}$ in an identical way for all clients in each iteration, which does not introduce additional communication costs. This approach minimizes the biased knowledge acquired by h_i' and facilitates the transfer of common knowledge from G_s to f_i.

3.3.5 Privacy-Preserving Discussion

Our FedKTL preserves privacy in three folds. (1) We introduce an identical ETF classifier for all clients to generate unbiased prototypes, which contain little private information. (2) The generated images belong to the generator's inherent output domain, so they are much different from the client's local data (see Fig. 4). (3) We keep all the model parameters locally on clients without sharing. *See the Appendix for further analysis and experimental results.*

4. Experiments

4.1. Setup

Datasets and baseline methods. In this paper, we evaluate our FedKTL on four image datasets, *i.e.*, Cifar10 [25], Cifar100 [25], Tiny-ImageNet [5], and Flowers102 [41] (8K images with 102 classes). Besides, we compare FedKTL with seven state-of-the-art HtFL methods, including LG-FedAvg [35], FedGen [75], FedGH [61], FML [49], FedKD [57], FedDistill [17], and FedProto [52].

Model heterogeneity scenarios. LG-FedAvg, FedGen, and FedGH assume the classifier to be homogeneous. Unless explicitly specified, we consider model heterogeneity for the main model part, *i.e.*, using Heterogeneous Feature Extractors (HtFE), for a fair comparison. Specifically, we denote the model heterogeneity scenarios by "HtFE$_X$", where the suffix number X represents the degree of model heterogeneity, and we utilize a total of X model architectures in HtFL. The larger the X is, the more heterogeneous the scenario is. Given N clients, we distribute the $(i \mod X)$th model architecture to client $i, i \in [N]$ and reinitialize its parameters. For instance, we use HtFE$_8$ by default, which includes eight model architectures: 4-layer CNN [39], GoogleNet [51], MobileNet_v2 [47], ResNet18, ResNet34, ResNet50, ResNet101, and ResNet152 [12]. The model architectures in HtFE$_8$ cover both small and large models. The feature dimensions K' before classifiers are different in these model architectures, which cannot meet the assumptions of FedGH, FedKD, and FedProto, so we add an average pooling layer [51] before classifiers and set $K' = 512$ by default for all model architectures.

Data heterogeneity. Following prior arts [39, 67, 75] in the FL field, we consider two data heterogeneity scenarios, including the pathological setting [52, 68, 70] and the practical setting [53, 66, 69]. In the pathological setting, following FedALA [69], we assign unbalanced data of 2/10/10/20 classes to each client from a total of 10/100/102/200 classes from Cifar10/Cifar100/Flowers102/Tiny-ImageNet datasets without overlap. As for the practical setting, following GPFL [67], we assign a proportion $q_{c,i}$ of data from a subset that contains all the data belonging to class c in a public dataset to client i, where $q_{c,i} \sim Dir(\beta)$, $Dir(\beta)$ is Dirichlet distribution and β is typically set to 0.1 [36].

General Implementation Details. We combine the above model and data heterogeneity to simulate HtFL scenarios. Besides, we split the local data into a training set and a test set with a ratio of 3:1 following [68, 69]. The performance of clients' models is assessed using their respective test sets, and these results (*e.g.*, test accuracy) are then averaged to gauge the performance of an HtFL method. Following FedAvg, we set the client batch size to 10 and run one training epoch with SGD [72], *i.e.*, $\lfloor \frac{n_i}{10} \rfloor$ SGD steps, on the client in each iteration. Besides, we set the client learning rate $\eta_c = 0.01$ and the total communication iterations to 1000. We run three trials and report the mean and standard deviation of the numerical results. We simulate HtFL scenarios on 20 clients with a client participation ratio $\rho = 1$, and we experiment on 50, 100, and 200 clients with $\rho = 0.5$.

Implementation Details for Our FedKTL. We set $\mu = 50$, $\lambda = 1$, $K = C$, $\eta_S = 0.01$, $B_S = 100$, and $E_S = 100$ by default on all tasks, where η_S, B_S, and E_S represent the learning rate, batch size, and number of epochs for training F on the server. Besides, we use Adam [24] for F training following FedGen and set $s = 64$ and $m = 0.5$ following ArcFace loss [7]. By default, we use a public pre-trained StyleGAN-XL [48] as the server-side generator (not used during clients' inference), which is one of the latest StyleGANs. It has approximately 0.13 billion model parameters and is trained on a large-scale ImageNet dataset [6] to generate images with a resolution of 64×64. To ensure compatibility with clients' models, we rescale the generated images on the server to match the resolution of clients' data before downloading them. *See the Appendix for the experiments using Stable Diffusion or only one edge client.*

4.2. Performance Comparison

We show the test accuracy of all the methods on four datasets in Tab. 1, where FedKTL achieves superior performance than baselines in HtFL scenarios. Specifically, our FedKTL outperforms counterparts by up to **5.40%** in test accuracy on Cifar100 in the practical setting. Besides, our FedKTL demonstrates greater superiority in the practical setting compared to the pathological setting. The number of generated images in $\mathcal{D}_I$ equals the number of classes C, so $|\mathcal{D}_I|$ is 10/100/102/200 for Cifar10/Cifar100/Flowers102/Tiny-ImageNet. Even with only 10 images in $\mathcal{D}_I$, our FedKTL can still perform excellently on Cifar10 in two data heterogeneous settings.

4.3. Impact of Model Heterogeneity

We further assess FedKTL on the other five scenarios with incremental model heterogeneity. Specifically, we con-

Settings	Pathological Setting				Practical Setting			
Datasets	Cifar10	Cifar100	Flowers102	Tiny-ImageNet	Cifar10	Cifar100	Flowers102	Tiny-ImageNet
LG-FedAvg	86.82±0.26	57.01±0.66	58.88±0.28	32.04±0.17	84.55±0.51	40.65±0.07	45.93±0.48	24.06±0.10
FedGen	82.83±0.65	58.26±0.36	59.90±0.15	29.80±1.11	82.55±0.49	38.73±0.14	45.30±0.17	19.60±0.08
FedGH	86.59±0.23	57.19±0.20	59.27±0.33	32.55±0.37	84.43±0.31	40.99±0.51	46.13±0.17	24.01±0.11
FML	87.06±0.24	55.15±0.14	57.79±0.31	31.38±0.15	85.88±0.08	39.86±0.25	46.08±0.53	24.25±0.14
FedKD	87.32±0.31	56.56±0.27	54.82±0.35	32.64±0.36	86.45±0.10	40.56±0.31	48.52±0.28	25.51±0.35
FedDistill	87.24±0.06	56.99±0.27	58.51±0.34	31.49±0.38	86.01±0.31	41.54±0.08	49.13±0.85	24.87±0.31
FedProto	83.39±0.15	53.59±0.29	55.13±0.17	29.28±0.36	82.07±1.64	36.34±0.28	41.21±0.22	19.01±0.10
FedKTL	**88.43±0.13**	**62.01±0.28**	**64.72±0.62**	**34.74±0.17**	**87.63±0.07**	**46.94±0.23**	**53.16±0.08**	**28.17±0.18**

Table 1. The test accuracy (%) on four datasets in the pathological and practical settings using HtFE_8.

Settings	Different Degrees of Model Heterogeneity					Large Client Amount ($\rho = 0.5$)		
	HtFE_2	HtFE_3	HtFE_4	HtFE_9	HtM_{10}	50 Clients	100 Clients	200 Clients
LG-FedAvg	46.61±0.24	45.56±0.37	43.91±0.16	42.04±0.26	—	37.81±0.12	35.14±0.47	27.93±0.04
FedGen	43.92±0.11	43.65±0.43	40.47±1.09	40.28±0.54	—	37.95±0.25	34.52±0.31	28.01±0.24
FedGH	46.70±0.35	45.24±0.23	43.29±0.17	43.02±0.86	—	37.30±0.44	34.32±0.16	29.27±0.39
FML	45.94±0.16	43.05±0.06	43.00±0.08	42.41±0.28	39.87±0.09	38.47±0.14	36.09±0.28	30.55±0.52
FedKD	46.33±0.24	43.16±0.49	43.21±0.37	42.15±0.36	40.36±0.12	38.25±0.41	35.62±0.55	31.82±0.50
FedDistill	46.88±0.13	43.53±0.21	43.56±0.14	42.09±0.20	40.95±0.04	38.51±0.36	36.06±0.24	31.26±0.13
FedProto	43.97±0.18	38.14±0.64	34.67±0.55	32.74±0.82	36.06±0.10	33.03±0.42	28.95±0.51	24.28±0.46
FedKTL	**48.06±0.19**	**49.83±0.44**	**47.06±0.21**	**50.33±0.35**	**45.84±0.15**	**43.16±0.82**	**39.73±0.87**	**34.24±0.45**

Table 2. The test accuracy (%) on Cifar100 in the practical setting with different degrees of model heterogeneity or large client amounts.

sider HtFE_2, HtFE_3, HtFE_4, HtFE_9, and HtM_{10}. HtFE_2 includes 4-layer CNN and ResNet18. HtFE_3 includes ResNet10 [74], ResNet18, and ResNet34. HtFE_4 includes 4-layer CNN, GoogleNet, MobileNet_v2, and ResNet18. HtFE_9 includes ResNet4, ResNet6, and ResNet8 [74], ResNet10, ResNet18, ResNet34, ResNet50, ResNet101, and ResNet152. HtM_{10} contains all the model architectures in HtFE_8 plus another two architectures ViT-B/16 [9] and ViT-B/32 [9]. "HtM" is short for heterogeneous models, where classifiers are also heterogeneous. LG-FedAvg, FedGen, and FedGH are *not applicable* for HtM_{10} due to the different classifier architectures of ResNets and ViTs. We allocate model architectures in HtM_{10} to clients using the method introduced for HtFE_X. We show the test accuracy in Tab. 2. For almost all the baselines, their performance deteriorates as model heterogeneity increases, resulting in an accuracy drop of at least **3.53%** from HtFE_2 to HtFE_9. In contrast, FedKTL attains its best performance with HtFE_9, outperforming baselines by **7.31%**.

4.4. Partial Participation with More Clients

To study the scalability of our FedKTL in HtFL settings with more clients, we introduce three scenarios with 50, 100, and 200 clients on HtFE_8, respectively, by splitting the Cifar100 dataset differently. With 200 participating clients, each class has an average of only eight samples for training. We consider partial client participation and set $\rho = 0.5$ in each iteration in these three scenarios. Notice that comparing the accuracy between these scenarios is unreasonable because both the number of clients and the amount of client data change when splitting Cifar100 into different numbers of clients' datasets. As shown in Tab. 2, our FedKTL maintains its superiority even with a large number of clients and partial client participation.

4.5. Impact of Number of Client Training Epochs

	$E = 5$	$E = 10$	$E = 20$
LG-FedAvg	40.33±0.15	40.46±0.08	40.93±0.23
FedGen	40.00±0.41	39.66±0.31	40.07±0.12
FedGH	41.09±0.25	39.87±0.27	40.22±0.41
FML	39.08±0.27	37.97±0.19	36.02±0.22
FedKD	41.06±0.13	40.36±0.20	39.08±0.33
FedDistill	41.02±0.30	41.29±0.23	41.13±0.41
FedProto	38.04±0.52	38.13±0.42	38.74±0.51
FedKTL	**46.18±0.34**	**45.70±0.27**	**45.57±0.23**

Table 3. The test accuracy (%) on Cifar100 in the practical setting using HtFE_8 with large E.

Training more epochs on clients before uploading can save communication resources [39]. Here, we increase the number of client training epochs and study its effects. From Tab. 3, we observe that most of the methods, except for FML and FedKD, can maintain their performance even with a large value of E. Notably, our FedKTL maintains its superior performance across different values of E. Since FML and FedKD learn an auxiliary model following the scheme of FedAvg, the auxiliary model tends to learn more biased information during local training with a larger value of E, which may deteriorate the auxiliary model aggregation [44].

4.6. Impact of Feature Dimensions

	$K' = 64$	$K' = 256$	$K' = 1024$
LG-FedAvg	39.69±0.25	40.21±0.11	40.46±0.01
FedGen	39.78±0.36	40.38±0.36	40.83±0.25
FedGH	39.93±0.45	40.80±0.40	40.19±0.37
FML	39.89±0.34	40.95±0.09	40.26±0.16
FedKD	41.06±0.18	41.14±0.35	40.72±0.25
FedDistill	41.69±0.10	41.66±0.15	40.09±0.27
FedProto	30.71±0.65	37.16±0.42	31.21±0.27
FedKTL	**46.46±0.41**	**47.81±0.43**	**45.91±0.54**

Table 4. The test accuracy (%) on Cifar100 in the practical setting using HtFE_8 with different K'.

Here, we study the impact of K' on the test accuracy. Most of the methods achieve their best performance when setting $K' = 256$, except for the methods that share classifiers, such as LG-FedAvg and FedGen. Using a larger value of K', FedProto can generate prototypes with dimension K' and upload more client information to the server. In contrast, our FedKTL generates prototypes after the projection layer ($h_i, i \in [N]$) with another dimension of $K = C < K'$. This dimension is fixed, *i.e.*, $K = 100$, for the 100-classification problem on Cifar100.

4.7. Communication Cost

Our FedKTL exhibits excellent performance while maintaining an affordable communication cost, as shown in Tab. 5. Specifically, FedKTL exhibits lower upload and download costs compared to FedGen, FML, and FedKD. Notably, the upload cost of our approach is the lowest among all the baselines, since we set $K = C$ for our FedKTL. Besides, the upload overhead required by FedKTL is much less than the download one, which is suitable for real-world scenarios, where the uplink speed is typically lower than the downlink speed [32]. The upload-efficient characteristic of FedKTL highlights its practicality for knowledge transfer in HtFL.

	Upload	Download	Accuracy
LG-FedAvg	1.03M	1.03M	40.65±0.07
FedGen	1.03M	7.66M	38.73±0.14
FedGH	0.46M	1.03M	40.99±0.51
FML	18.50M	18.50M	39.86±0.25
FedKD	16.52M	16.52M	40.56±0.31
FedDistill	0.09M	0.20M	41.54±0.08
FedProto	0.46M	1.02M	36.34±0.28
FedKTL	0.09M	7.17M	**46.94±0.23**

Table 5. The upload and download overhead per iteration using HtFE_8 on Cifar100 with 20 clients in the practical setting. "M" is short for million. The accuracy column is referred from Tab. 1.

	$\lambda = 0.05$	$\lambda = 0.1$	$\lambda = 0.5$
AFHQv2	26.82±0.32	**27.05±0.26**	26.32±0.52
Bench	27.71±0.25	**28.36±0.42**	27.56±0.50
FFHQ-U	**27.28±0.23**	27.21±0.35	26.59±0.47
WikiArt	27.37±0.51	**27.48±0.33**	27.30±0.15

Table 6. The test accuracy (%) on Tiny-ImageNet in the practical setting using HtFE_8 with different pre-trained StyleGAN3s, which are represented by the names of the pre-training datasets.

4.8. Adapting to Various Pre-Trained StyleGAN3s

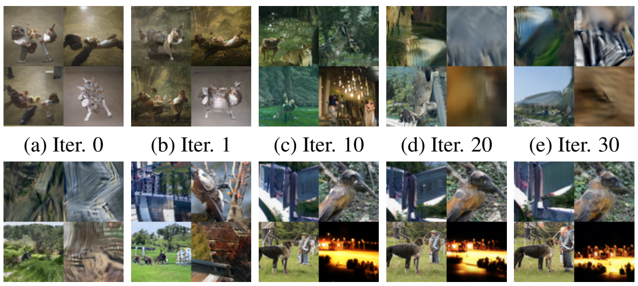
Although we adopt the pre-trained StyleGAN-XL by default as the server generator, our FedKTL is also applicable to other StyleGANs due to the adaptable ability of our feature transformer (F). Here we consider utilizing the popular StyleGAN3 [23], which has nearly $\frac{1}{3}$ of the parameter count compared to StyleGAN-XL. Specifically, we use several public StyleGAN3s pre-trained on four datasets with different resolutions: AFHQv2 (512×512) [23], Benches (512×512) [2], FFHQ-U (256×256) [23], and WikiArt (1024×1024) [46]. To adapt to different pre-trained generators, we re-tune the hyperparameter λ. According to Tab. 1 and Tab. 6, our FedKTL maintains excellent performance even when using other generators with different pre-training datasets. In FedKTL, we prioritize the class-wise discrimination of the generated images over their semantic content. Thus, the knowledge-transfer loop remains valuable when generated images are distinguishable by classes but do not share semantic relevance with clients' data (see Fig. 4).

4.9. Iterative Domain Alignment Process

The training process in HtFL is iterative, so the domain alignment in our FedKTL is also an iterative process. Here we demonstrate the generated images throughout HtFL's training process in Fig. 5 to show the iterative domain alignment process. In the early iterations, as shown in Fig. 5a and

(a) Client #1 (b) AFHQv2 (c) Benches (d) FFHQ-U (e) WikiArt

Figure 4. (a): Four images (one image per class) on client #1. (b), (c), (d), and (e): The images generated by different StyleGAN3s corresponding to the aforementioned four classes.



(a) Iter. 0 (b) Iter. 1 (c) Iter. 10 (d) Iter. 20 (e) Iter. 30

(f) Iter. 50 (g) Iter. 100 (h) Iter. 110 (i) Iter. 120 (j) Iter. 130

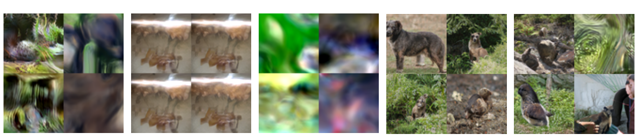
Figure 5. The images generated by StyleGAN-XL corresponding to four classes at different iterations.

Fig. 5b, the generated images ($\mathcal{D}_I$) corresponding to class-wise latent centroids ($\bar{\mathcal{Q}}$) appear similar, since clients cannot generate discriminative prototypes. As HtFL's training process continues, the generated images become increasingly class-discriminative and clear. The generated images in iterations 110, 120, and 130 hardly change for each class, showing the convergence of F and client models' training.

4.10. Ablation Study

FedKTL	-L_i^M	-L^{MSE}	-L^{MMD}	-*ETF*	-$\bar{\mathcal{Q}}$	+*CS*
28.17	24.39	21.70	20.14	21.02	20.69	24.13

Table 7. The test accuracy (%) of our FedKTL's variants on Tiny-ImageNet in the practical setting using HtFE$_8$.



(a) -L_i^M (b) -L^{MSE} (c) -L^{MMD} (d) -*ETF* (e) -$\bar{\mathcal{Q}}$

Figure 6. The images generated by StyleGAN-XL corresponding to four classes in our FedKTL's variants when variants converge.

Here, we remove L_i^M, L^{MMD}, and L^{MSE} from FedKTL and denote these variants "-L_i^M", "-L^{MMD}", and "-L^{MSE}", respectively. Moreover, we create the following three variants. (1) "-*ETF*": we remove h_i and replace the ETF classifier with the original classifier of each model architecture. (2) "-$\bar{\mathcal{Q}}$": we remove L_i^M and mix the generated class-discriminative data $\mathcal{D}_I$ with local data $\mathcal{D}_i$. (3) Besides the common practice of using noise $\boldsymbol{\epsilon}$ to generate images, StyleGAN-XL offers a conditional version that can generate images belonging to any class from the ImageNet dataset. Using the Conditional StyleGAN-XL (CS), we create a variant "+*CS*" by disabling step ② Upload and step ③ Domain Alignment, and directly generating C image-vector pairs for C randomly selected ImageNet classes.

The poor results of these variants in Tab. 7 and Fig. 6 demonstrate the effectiveness of each key component in our FedKTL. Below, we analyze them one by one. (1) -L_i^M: removing L_i^M means training solely on the local dataset $\mathcal{D}_i$ without collaboration, leading to a **3.78%** accuracy drop and distorted generated images (unused). (2) -L^{MSE}: removing L^{MSE} causes the generated images to become indiscriminative, thus misleading the local extractor and causing an accuracy drop of **6.47%**. (3) -L^{MMD}: without the MMD loss for domain alignment, it is hard for $\bar{\mathcal{Q}}$ to be valid latent input vectors for the generator, leading to blurry images and a notable accuracy decrease. (4) -*ETF*: biased classifiers make prototypes of different classes overlap, resulting in a loss of class-wise discrimination of the generated images. In Fig. 6d, three out of the four images depict dogs and grass. (5) -$\bar{\mathcal{Q}}$: without $\bar{\mathcal{Q}}$, only using $\mathcal{D}_I$ on clients cannot transfer knowledge from the generator and mixing $\mathcal{D}_I$ and $\mathcal{D}_i$ perturb the semantics of local data, thus achieving poor performance and generating images with strange contents. (6) +*CS*: using a conditional generator to produce class-wise image-vector pairs without adapting to clients' tasks can harm local training, as evidenced by a **0.26%** decrease in accuracy compared to -L_i^M (no collaboration). (7) Interestingly, the variants -L^{MSE}, -L^{MMD}, -ETF, and -$\bar{\mathcal{Q}}$ perform worse than -L_i^M, which indicates that all key components are crucial and assist each other in FedKTL.

5. Conclusion

We propose FedKTL to promote client training in HtFL by (1) producing image-vector pairs that are related to clients' tasks through a pre-trained generator's inference on the server, and (2) transferring pre-existing knowledge from the generator to clients' heterogeneous models. Extensive experiments show the effectiveness, efficiency, and practicality of our FedKTL in various scenarios.

Acknowledgements

This work was supported by the National Key R&D Program of China under Grant No.2022ZD0160504, the Program of Technology Innovation of the Science and Technology Commission of Shanghai Municipality (Granted No. 21511104700), and Tsinghua University(AIR)-Asiainfo Technologies (China) Inc. Joint Research Center.

References

[1] Enes Ayan and Halil Murat Ünver. Data augmentation importance for classification of skin lesions via deep learning. In *2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT)*, pages 1–4. IEEE, 2018. 1

[2] Marion Bowman. Trees, benches and contemporary commemoration: When the ordinary becomes extraordinary. *Journal for the Study of Religious Experience*, 7(3):33–49, 2021. 7

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1

[4] Sema Candemir, Xuan V Nguyen, Les R Folio, and Luciano M Prevedello. Training strategies for radiology deep learning models in data-limited scenarios. *Radiology: Artificial Intelligence*, 3(6):e210014, 2021. 1

[5] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A Downsampled Variant of Imagenet as an Alternative to the Cifar Datasets. *arXiv preprint arXiv:1707.08819*, 2017. 5

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A Large-Scale Hierarchical Image Database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 5

[7] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3, 5

[8] Enmao Diao, Jie Ding, and Vahid Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. In *International Conference on Learning Representations (ICLR)*, 2020. 2

[9] 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. In *International Conference on Learning Representations (ICLR)*, 2020. 6

[10] Xiuwen Fang, Mang Ye, and Xiyuan Yang. Robust heterogeneous federated learning under data corruption. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[11] Zhida Feng, Zhenyu Zhang, Xintong Yu, Yewei Fang, Lanxin Li, Xuyi Chen, Yuxiang Lu, Jiaxiang Liu, Weichong Yin, Shikun Feng, et al. Ernie-vilg 2.0: Improving text-to-image diffusion model with knowledge-enhanced mixture-of-denoising-experts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[14] Samuel Horvath, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos Venieris, and Nicholas Lane. Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

[15] Linmei Hu, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, and Juanzi Li. A survey of knowledge enhanced pre-trained language models. *IEEE Transactions on Knowledge and Data Engineering*, 2023. 1

[16] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International Conference on Machine Learning (ICML)*, 2015. 4

[17] 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. 2, 5

[18] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, 2020. 1

[19] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and Open Problems in Federated Learning. *arXiv preprint arXiv:1912.04977*, 2019. 1

[20] 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 (ICML)*, 2020. 1

[21] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 3

[22] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[23] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 7

[24] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 5

[25] Alex Krizhevsky and Hinton Geoffrey. Learning Multiple Layers of Features From Tiny Images. *Technical Report*, 2009. 5

[26] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019. 2

[27] 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 Pri-

vacy and Protection. *IEEE Transactions on Knowledge and Data Engineering*, 2021. 1

[28] 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

[29] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated Optimization in Heterogeneous Networks. In *Conference on Machine Learning and Systems (MLSys)*, 2020. 1

[30] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and Robust Federated Learning Through Personalization. In *International Conference on Machine Learning (ICML)*, 2021. 1

[31] Xin-Chun Li, De-Chuan Zhan, Yunfeng Shao, Bingshuai Li, and Shaoming Song. FedPHP: Federated Personalization with Inherited Private Models. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML)*, 2021. 4

[32] Zhenhua Li, Xingyao Li, Xinlei Yang, Xianlong Wang, Feng Qian, and Yunhao Liu. Fast uplink bandwidth testing for internet users. *IEEE/ACM Transactions on Networking*, 2023. 7

[33] Zexi Li, Xinyi Shang, Rui He, Tao Lin, and Chao Wu. No fear of classifier biases: Neural collapse inspired federated learning with synthetic and fixed classifier. In *IEEE International Conference on Computer Vision (ICCV)*, 2023. 2, 3

[34] Ziyun Li, Xinshao Wang, Neil M Robertson, David A Clifton, Christoph Meinel, and Haojin Yang. Smkd: Selective mutual knowledge distillation. In *International Joint Conference on Neural Networks (IJCNN)*, 2023. 2

[35] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020. 2, 5

[36] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:2351–2363, 2020. 1, 2, 5

[37] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning (ICML)*, 2015. 4

[38] Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No Fear of Heterogeneity: Classifier Calibration for Federated Learning with Non-IID data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

[39] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017. 5, 7

[40] Hieu V Nguyen and Li Bai. Cosine similarity metric learning for face verification. In *Asian Conference on Computer Vision (ACCV)*, 2010. 3

[41] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE, 2008. 5

[42] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020. 2, 3

[43] Francesco Ponzio, Gianvito Urgese, Elisa Ficarra, and Santa Di Cataldo. Dealing with lack of training data for convolutional neural networks: the case of digital pathology. *Electronics*, 8(3):256, 2019. 1

[44] Liangqiong Qu, Yuyin Zhou, Paul Pu Liang, Yingda Xia, Feifei Wang, Ehsan Adeli, Li Fei-Fei, and Daniel Rubin. Rethinking architecture design for tackling data heterogeneity in federated learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 7

[45] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[46] Babak Saleh and Ahmed Elgammal. Large-scale classification of fine-art paintings: Learning the right metric on the right feature. *arXiv preprint arXiv:1505.00855*, 2015. 7

[47] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5

[48] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 conference proceedings*, pages 1–10, 2022. 2, 3, 5

[49] 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. 2, 5

[50] Kaili Sun, Xudong Luo, and Michael Y Luo. A survey of pretrained language models. In *International Conference on Knowledge Science, Engineering and Management*, 2022. 1

[51] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 5

[52] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated Prototype Learning across Heterogeneous Clients. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2022. 1, 2, 5

[53] 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 (NeurIPS)*, 2022. 1, 2, 5

[54] Michael Tuchler, Andrew C Singer, and Ralf Koetter. Minimum Mean Squared Error Equalization using A Priori Information. *IEEE Transactions on Signal Processing*, 50(3): 673–683, 2002. 4

[55] Lianyu Wang, Meng Wang, Daoqiang Zhang, and Huazhu Fu. Model barrier: A compact un-transferable isolation domain for model intellectual property protection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1

[56] 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

[57] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1): 2032, 2022. 1, 2, 5

[58] Xiyuan Yang, Wenke Huang, and Mang Ye. Dynamic personalized federated learning with adaptive differential privacy. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 1

[59] Yibo Yang, Shixiang Chen, Xiangtai Li, Liang Xie, Zhouchen Lin, and Dacheng Tao. Inducing neural collapse in imbalanced learning: Do we really need a learnable classifier at the end of deep neural network? *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 3

[60] Yibo Yang, Haobo Yuan, Xiangtai Li, Zhouchen Lin, Philip Torr, and Dacheng Tao. Neural collapse inspired feature-classifier alignment for few-shot class-incremental learning. In *International Conference on Learning Representations (ICLR)*, 2022. 2

[61] 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*, 2023. 1, 2, 5

[62] Qiying Yu, Yang Liu, Yimu Wang, Ke Xu, and Jingjing Liu. Multimodal federated learning via contrastive representation ensemble. In *International Conference on Learning Representations (ICLR)*, 2022. 2

[63] Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph Stoecklin, Heqing Huang, and Ian Molloy. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the 2018 on Asia conference on computer and communications security*, pages 159–172, 2018. 1

[64] Jie Zhang, Song Guo, Xiaosong Ma, Haozhao Wang, Wenchao Xu, and Feijie Wu. Parameterized Knowledge Transfer for Personalized Federated Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 1

[65] Jie Zhang, Song Guo, Jingcai Guo, Deze Zeng, Jingren Zhou, and Albert Zomaya. Towards data-independent knowledge transfer in model-heterogeneous federated learning. *IEEE Transactions on Computers*, 2023. 1, 2

[66] 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. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 5

[67] 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 *IEEE International Conference on Computer Vision (ICCV)*, 2023. 1, 5

[68] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Fedcp: Separating feature information for personalized federated learning via conditional policy. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2023. 5

[69] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. FedALA: Adaptive Local Aggregation for Personalized Federated Learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2023. 1, 5

[70] 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. *arXiv preprint arXiv:2401.03230*, 2024. 1, 2, 5

[71] Lin Zhang, Li Shen, Liang Ding, Dacheng Tao, and Ling-Yu Duan. Fine-Tuning Global Model Via Data-Free Knowledge Distillation for Non-IID Federated Learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[72] Sixin Zhang, Anna E Choromanska, and Yann LeCun. Deep Learning with Elastic Averaging SGD. *Advances in Neural Information Processing Systems (NeurIPS)*, 2015. 5

[73] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[74] Zilong Zhong, Jonathan Li, Lingfei Ma, Han Jiang, and He Zhao. Deep residual networks for hyperspectral image classification. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 1824–1827. IEEE, 2017. 6

[75] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-Free Knowledge Distillation for Heterogeneous Federated Learning. In *International Conference on Machine Learning (ICML)*, 2021. 2, 5

[76] Weiming Zhuang, Chen Chen, and Lingjuan Lyu. When foundation model meets federated learning: Motivations, challenges, and future directions. *arXiv preprint arXiv:2306.15546*, 2023. 1