FedTGP: Trainable Global Prototypes with Adaptive-Margin-Enhanced Contrastive Learning for Data and Model Heterogeneity in Federated Learning

Jianqing Zhang[1], Yang Liu[2*], Yang Hua[3], Jian Cao[1*]

[1]Shanghai Jiao Tong University
[2]Institute for AI Industry Research, Tsinghua University
[3]Queen's University Belfast
tsingz@sjtu.edu.cn, liuy03@air.tsinghua.edu.cn, y.hua@qub.ac.uk, cao-jian@sjtu.edu.cn

Abstract

Recently, Heterogeneous Federated Learning (HtFL) has attracted attention due to its ability to support heterogeneous models and data. To reduce the high communication cost of transmitting model parameters, a major challenge in HtFL, prototype-based HtFL methods are proposed to solely share class representatives, a.k.a, prototypes, among heterogeneous clients while maintaining the privacy of clients' models. However, these prototypes are naively aggregated into global prototypes on the server using weighted averaging, resulting in suboptimal global knowledge which negatively impacts the performance of clients. To overcome this challenge, we introduce a novel HtFL approach called **FedTGP**, which leverages our **Adaptive-margin-enhanced Contrastive Learning (ACL)** to learn **Trainable Global Prototypes (TGP)** on the server. By incorporating ACL, our approach enhances prototype separability while preserving semantic meaning. Extensive experiments with twelve heterogeneous models demonstrate that our FedTGP surpasses state-of-the-art methods by up to **9.08%** in accuracy while maintaining the communication and privacy advantages of prototype-based HtFL. Our code is available at https://github.com/TsingZ0/FedTGP.

Introduction

With the rapid increase in the amount of data required to train large models today, concerns over data privacy also rise sharply (Shin et al. 2023; Li et al. 2021a). To facilitate training machine learning models while protecting data privacy, Federated Learning (FL) has emerged as a new distributed machine learning paradigm (Kairouz et al. 2019; Li et al. 2020). However, in practical scenarios, traditional FL methods such as FedAvg (McMahan et al. 2017) experience performance degradation when faced with statistical heterogeneity (T Dinh, Tran, and Nguyen 2020; Li et al. 2022). Subsequently, personalized FL methods emerged to address the challenge of statistical heterogeneity by learning personalized model parameters. Nevertheless, most of them still assume the model architectures on all the clients are the same and communicate client model updates to the server to train a shared global model (Zhang et al. 2023d,c,b; Collins et al. 2021; Li et al. 2021b). These methods not only

bring formidable communication cost (Zhuang, Chen, and Lyu 2023) but also expose clients' models, which further raise privacy and intellectual property (IP) concerns (Li et al. 2021a; Zhang et al. 2018a; Wang et al. 2023).

To alleviate these problems, Heterogeneous FL (HtFL) (Tan et al. 2022b) has emerged as a novel FL paradigm that enables clients to possess diverse model architectures and heterogeneous data without sharing private model parameters. Instead, various types of global knowledge are shared among clients to reduce communication and improve model performance. For example, some FL methods adopt knowledge distillation (KD) techniques (Hinton, Vinyals, and Dean 2015) and communicate predicted logits on a public dataset (Li and Wang 2019; Lin et al. 2020; Liao et al. 2023; Zhang et al. 2021) as global knowledge for aggregation at the server. However, these methods highly depend on the availability and quality of the global dataset (Zhang et al. 2023a). Data-free KD-based approaches utilize additional auxiliary models as global knowledge (Wu et al. 2022; Zhang et al. 2022), but the communication overhead for sharing the auxiliary models is still considerable. Alternatively, prototype-based HtFL methods (Tan et al. 2022b,c) propose to share lightweight class representatives, a.k.a, *prototypes*, as global knowledge, significantly reducing communication overhead.

However, existing prototype-based HtFL methods naively aggregate heterogeneous client prototypes on the server using weighted-averaging, which has several limitations. First, the weighted-averaging protocol requires clients to upload class distribution information of private data to the server as weights, which leaks sensitive distribution information about clients' data (Yi et al. 2023). Secondly, the prototypes generated from heterogeneous clients have diverse scales and separation margins. Averaging client prototypes generates uninformative global prototypes with smaller margins than the margins between well-separated prototypes. We demonstrate this "*prototype margin shrink*" phenomenon in Fig. 1(a). However, smaller margins between prototypes diminish their separability, ultimately generating poor prototypes (Zhang and Sato 2023).

To address these limitations, we design a novel HtFL method using **Trainable Global Prototypes (TGP)**, termed **FedTGP**, in which we train the desired global prototypes with our proposed **Adaptive-margin-enhanced Con-**

(a) The prototype margins in FedProto using Cifar10.
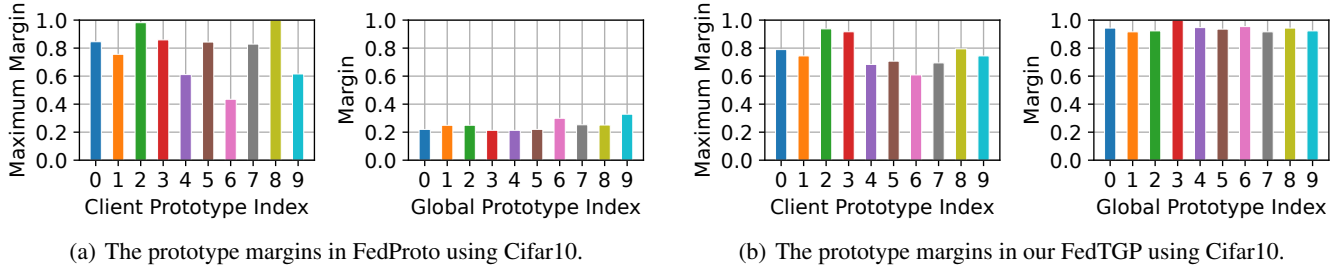
(b) The prototype margins in our FedTGP using Cifar10.

Figure 1: The illustration of the prototype margin change after generating global prototypes. The *prototype margin* is the minimum Euclidean distance between the prototype of a specific class and the prototypes of other classes, and the maximum margin is the maximum prototype margin among all clients for each class. To enhance visualization and eliminate the influence of magnitude, we normalize the margin values for each method in these figures. Different colors represent different classes. (a) The global prototype margin *shrinks* compared to the maximum of clients' prototype margins in FedProto. (b) The global prototype margin *improves* compared to the maximum of clients' prototype margins in our FedTGP.

trastive Learning (ACL). Specifically, we train the global prototypes to be separable while maintaining semantics via contrastive learning (Hayat et al. 2019) with a specified margin. To avoid using an overlarge margin in early iterations and keep the best separability per iteration, we enhance contrastive learning by our adaptive margin, which reserves the maximum prototype margin among all clients in each iteration, as shown in Fig. 1(b). With the guidance of our separable global prototypes, FedTGP can further enlarge the interclass intervals for feature representations on each client.

To evaluate the effectiveness of our FedTGP, we conduct extensive experiments and compare it with six state-of-the-art methods in two popular statistically heterogeneous settings on four datasets using twelve heterogeneous models. Experimental results reveal that FedTGP outperforms FedProto by up to **18.96%** and surpasses other baseline methods by a large gap. Our contributions are:

- We observe that naively averaging prototypes can result in ineffective global prototypes in FedProto-like schemes, as it causes the separation margin to shrink due to model heterogeneity in HtFL.

- We propose an HtFL method called FedTGP that learns trainable global prototypes with our adaptive-margin-enhanced contrastive learning technique to enhance interclass separability.

- Extensive comparison and ablation experiments on four datasets with twelve heterogeneous models demonstrate the superiority of FedTGP over FedProto and other HtFL methods.

Related Work

Heterogeneous Federated Learning

In recent times, Federated Learning (FL) has become a new machine learning paradigm that enables collaborative model training without exposing client data. Although personalized FL methods (T Dinh, Tran, and Nguyen 2020; Zhang et al. 2023e; Yang, Huang, and Ye 2023; Li et al. 2021b; Collins et al. 2021) are proposed soon afterward to tackle the statistical heterogeneity of FL, they are still inapplicable for scenarios where clients own heterogeneous models for their specific tasks. Heterogeneous Federated Learning (HtFL) has emerged as a solution to support both model heterogeneity and statistical heterogeneity simultaneously, protecting both privacy and IP.

One HtFL approach allows clients to sample diverse submodels from a shared global model architecture to accommodate the diverse communication and computing capabilities (Diao, Ding, and Tarokh 2020; Horvath et al. 2021; Wen, Jeon, and Huang 2022). However, concerns over sharing clients' model architectures still exist. Another HtFL approach is to split each client's model architecture and only share the top layers while allowing the bottom layers to have different architectures, *e.g.*, LG-FedAvg (Liang et al. 2020) and FedGen (Zhu, Hong, and Zhou 2021). However, sharing and aggregating top layers may lead to unsatisfactory performance due to statistical heterogeneity (Li et al. 2023a; Luo et al. 2021; Wang et al. 2020). Although learning a global generator can enhance the generalization ability (Zhu, Hong, and Zhou 2021), its effectiveness highly relies on the quality of the generator.

The above HtFL methods still require clients to have codependent model architectures. Alternatively, other methods seek to achieve HtFL with fully independent client models while communicating various kinds of information other than clients' models. Classic KD-based HtFL approaches (Li and Wang 2019; Yu et al. 2022) share predicted knowledge on a global dataset to enable knowledge transfer among heterogeneous clients, but such a global dataset can be difficult to obtain (Zhang et al. 2023a). FML (Shen et al. 2020) and FedKD (Wu et al. 2022) simultaneously train and share a small auxiliary model using mutual distillation (Zhang et al. 2018b) instead of using a global dataset. However, during the early iterations with poor feature-extracting abilities, the client model and the auxiliary model can potentially interfere with each other (Li et al. 2023b). Another popular approach is to share compact class representatives, *i.e.*, prototypes. FedDistill (Jeong et al. 2018) sends the class-wise logits from clients to the server and guides client model training by the globally averaged logits.

FedProto (Tan et al. 2022b) and FedPCL (Tan et al. 2022c) share higher-dimensional prototypes instead of logits. However, all these approaches perform naive weighted-averaging on the clients' prototypes, resulting in subpar global prototypes due to statistical and model heterogeneity in HtFL. While FedPCL applies contrastive learning on each client for projection network training, it relies on pre-trained models, which is hard to satisfy in FL with private model architectures as clients join FL due to data scarcity (Tan et al. 2022a). In this work, we explore methods to enhance the optimization of global prototypes, while maintaining the communication advantages inherent in such prototype-based approaches.

Trainable Prototype Learning

In centralized learning scenarios, trainable prototypes have been explored during model training to improve the intra-class compactness and inter-class discrimination of feature representations through the cross entropy loss (Pinheiro 2018; Yang et al. 2018) and regularizers (Xu et al. 2020; Jin, Liu, and Hou 2010). Besides, some domain adaptation methods (Tanwisuth et al. 2021; Kim and Kim 2020) learn trainable global prototypes to transfer knowledge among domains. However, all these methods assume that data are nonprivate and the prototype learning depends on access to model and feature representations, which are infeasible in the FL setting.

In our FedTGP, we perform prototype learning on the server based solely on the knowledge of clients' prototypes, without accessing client models or features. In this way, the learning process of client models and global prototypes can be fully decoupled while mutually facilitating each other.

Method

Problem Statement and Motivation

We have M clients collaboratively train their models with heterogeneous architectures on their private and heterogeneous data $\{\mathcal{D}_i\}_{i=1}^M$. Following FedProto (Tan et al. 2022b), we split each client i's model into a feature extractor f_i parameterized by θ_i, which maps an input space $\mathbb{R}^D$ to a feature space $\mathbb{R}^K$, and a classifier h_i parameterized by w_i, which maps the feature space to a class space $\mathbb{R}^C$. Clients collaborate by sharing global prototypes $\mathcal{P}$ with a server. Formally the overall collaborative training objective is

$$\min_{\{\{\theta_i, w_i\}\}_{i=1}^M} \frac{1}{M} \sum_{i=1}^M \mathcal{L}_i(\mathcal{D}_i, \theta_i, w_i, \mathcal{P}). \quad (1)$$

In FedProto, each client i first obtains its prototype for each class c:

$$P_i^c = \mathbb{E}_{(\boldsymbol{x},c)\sim\mathcal{D}_{i,c}} f_i(\boldsymbol{x}; \theta_i), \quad (2)$$

where $\mathcal{D}_{i,c}$ denotes the subset of $\mathcal{D}_i$ consisting of all data points belonging to class c. After receiving all prototypes from clients, the server then performs weighted-averaging for each class prototype:

$$\bar{P}^c = \frac{1}{|\mathcal{N}_c|} \sum_{i \in \mathcal{N}_c} \frac{|\mathcal{D}_{i,c}|}{N_c} P_i^c, \quad (3)$$
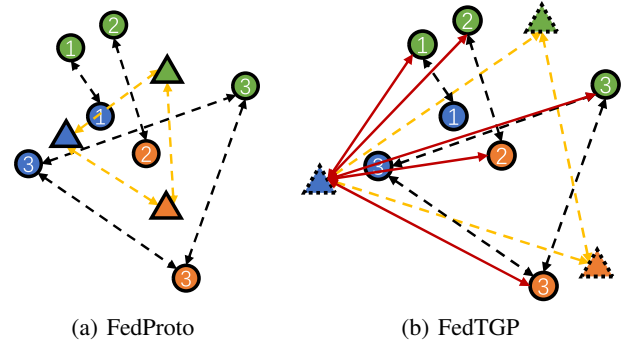

(a) FedProto (b) FedTGP

Figure 2: The global and client prototypes in FedProto and our FedTGP. Different colors and numbers represent classes and clients, respectively. Circles represent the client prototypes and triangles represent the global prototypes. The black and yellow dotted arrows show the inter-class separation among the client and global prototypes, respectively. Triangles with dotted borders represent our TGP. The red arrows show the inter-class intervals between TGP and the client prototypes of other classes in our ACL.

where $\mathcal{N}_c$ and N_c are the client set owning class c and the total number of data of class c among all clients. Next, the server transfers global information $\mathcal{P} = \{\bar{P}^c\}_{c=1}^C$ to each client, who performs guided training with a supervised loss

$$\mathcal{L}_i := \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i}\ell(h_i(f_i(\boldsymbol{x}; \theta_i); w_i), y) + \lambda\mathbb{E}_{c\sim\mathcal{C}_i}\phi(P_i^c, \bar{P}^c), \quad (4)$$

where ℓ is the loss for client tasks, λ is a hyperparameter, and ϕ measures the Euclidean distance. $\mathcal{C}_i$ is a set of classes on the data of client i. Different clients may own different $\mathcal{C}$ in HtFL with heterogeneous data.

We observe that performing simple weighted-averaging to clients' prototypes in a heterogeneous environment may not generate desired information as expected, and we illustrate this phenomenon in Fig. 2(a). Due to the statistical and model heterogeneity, different clients extract much diverse feature representations of different classes with various separability and prototype margins. The weighted-averaging process assigns weights to client prototypes based solely on the amount of data, as indicated by Eq. (3). However, since model performance in a heterogeneous environment can not be fully characterized by the data amount, prototypes generated by a poor client model may still be assigned a larger weight, causing the margin of global prototypes worse than the well-separated prototypes and impairing the training of the client models that previously produce well-separated prototypes.

To address the above problem, we propose FedTGP to (1) use Trainable Global Prototypes (TGP) with a separation objective on the server, (2) guide them to maintain large inter-class intervals with client prototypes while preserving semantics through our Adaptive-margin-enhanced Contrastive Learning (ACL) in each iteration, as shown in Fig. 2(b), and (3) finally improve separability of different classes on each client with the guidance of separable global prototypes.
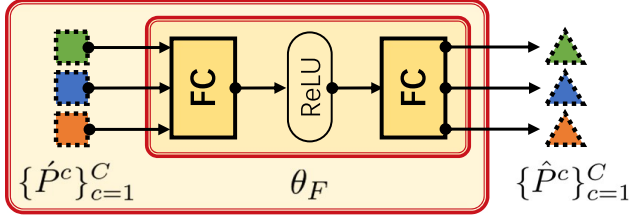
Trainable Global Prototypes

Figure 3: An example of trainable vectors ($\{\acute{P}^c\}_{c=1}^C$) and the further processing model (θ_F). They only exist on the server.

In this section, we aim to learn a new set of global prototypes $\hat{\mathcal{P}} = \{\hat{P}^c\}_{c=1}^C$. Formally, we first randomly initialize a trainable vector $\acute{P}^c \in \mathbb{R}^K$ for each class c. Next, we place a neural network model F, parameterized by θ_F, on the server to further process $\acute{P}^c$ to improve its training ability. The model F transforms a given trainable vector to a global prototype with the same shape, *i.e.*, $\forall c \in [C], \hat{P}^c = F(\acute{P}^c; \theta_F)$, $\hat{P}^c \in \mathbb{R}^K$, as shown in Fig. 3. F consists of two Fully-Connected (FC) layers with a ReLU activation function in between. This structure is widely used for the server model in FL (Chen and Chao 2021; Shamsian et al. 2021; Ma et al. 2022). In other words, the trainable global prototype $\hat{P}^c$ is parameterized by $\{\acute{P}^c, \theta_F\}$, and prototypes of different classes share the same parameter θ_F.

In order to learn effective prototypes, the trainable global prototype of class c needs to achieve two goals: (1) closely align with the client prototypes of class c to retain semantic information, and (2) maintain a significant distance from the client prototypes of other classes to enhance separability. The compactness and separation characteristics of contrastive learning (Hayat et al. 2019; Deng et al. 2019) meet these two targets simultaneously. Thus, we can learn $\hat{\mathcal{P}}$ by

$$\min_{\hat{\mathcal{P}}} \sum_{c=1}^C \mathcal{L}_P^c, \tag{5}$$

$$\mathcal{L}_P^c = \sum_{i \in \mathcal{I}^t} -\log \frac{e^{-\phi(P_i^c, \hat{P}^c)}}{e^{-\phi(P_i^c, \hat{P}^c)} + \sum_{c'} e^{-\phi(P_i^c, \hat{P}^{c'})}}, \tag{6}$$

where $c' \in [C], c' \neq c$, and $\mathcal{I}^t$ is the participating client set at tth iteration with client participation ratio ρ. Notice that all C trainable global prototypes participate in the contrastive learning term in Eq. (6), which means they share pair-wise interactions with each other when performing gradient updates, and the gradient updates can be performed even with partial client participation.

Adaptive-Margin-Enhanced Contrastive Learning

Although the standard contrastive loss Eq. (6) can improve compactness and separation, it does not reduce intra-class variations. Moreover, the learned inter-class separation boundary may still lack clarity (Choi, Som, and Turaga 2020). To further improve the separability of global prototypes, we enforce a margin between classes when learning

Algorithm 1: The learning process of FedTGP.

Input: M clients with their heterogeneous models and data, trainable global prototypes $\hat{\mathcal{P}}$ on the server, η: learning rate, T: total communication iterations.
Output: Well-trained client models.
1: **for** iteration $t = 1, \ldots, T$ **do**
2:     Server randomly samples a client subset $\mathcal{I}^t$.
3:     Server sends $\hat{\mathcal{P}}$ to $\mathcal{I}^t$.
4:     **for** Client $i \in \mathcal{I}^t$ in parallel **do**
5:         Client i updates its model with Eq. (11).
6:         Client i calculates prototypes $\mathcal{P}_i$ by Eq. (2).
7:         Client i sends $\mathcal{P}_i$ to the server.
8:     Server obtains $\delta(t)$ through Eq. (9)
9:     Server updates $\hat{\mathcal{P}}$ with Eq. (10).
10: **return** Client models.

$\hat{\mathcal{P}}$. Inspired by the additive angular margin of ArcFace (Deng et al. 2019) used in an angular space for face recognition, we introduce a scalar δ to Eq. (6) in our considered Euclidean space and rewrite $\mathcal{L}_P^c$ as

$$\mathcal{L}_P^c = \sum_{i \in \mathcal{I}^t} -\log \frac{e^{-(\phi(P_i^c, \hat{P}^c)+\delta)}}{e^{-(\phi(P_i^c, \hat{P}^c)+\delta)} + \sum_{c'} e^{-\phi(P_i^c, \hat{P}^{c'})}}, \tag{7}$$

where $\delta > 0$. According to (Schroff, Kalenichenko, and Philbin 2015; Hayat et al. 2019), minimizing $\mathcal{L}_P^c$ is equivalent to minimizing $\tilde{\mathcal{L}}_P^c$,

$$\mathcal{L}_P^c \propto \tilde{\mathcal{L}}_P^c := \sum_{i \in \mathcal{I}^t} \sum_{c'} e^{\phi(P_i^c, \hat{P}^c) - \phi(P_i^c, \hat{P}^{c'}) + \delta}, \tag{8}$$

which reduces the distance between P_i^c and $\hat{P}^c$ while increasing the distance between P_i^c and $\hat{P}^{c'}$ with a margin δ.

However, we observe that setting a large δ in early iterations may also mislead both the prototype training and the client model training because the feature extraction abilities of heterogeneous models are poor in the beginning. To retain the best separability of client prototypes within the semantic region in each iteration, we set the adaptive $\delta(t)$ to be the maximum cluster margin among client prototypes of different classes with a threshold τ,

$$\delta(t) = \min(\max_{c \in [C], c' \in [C], c \neq c'} \phi(Q_t^c, Q_t^{c'}), \tau), \tag{9}$$

where $Q_t^c = \frac{1}{|\mathcal{P}_t^c|} \sum_{i \in \mathcal{I}^t} P_i^c, \forall c \in [C]$ represents the cluster center of the client prototypes for each class, and it differs from the weighted average $\bar{P}^c$ which adopts private distribution information as weights. $\mathcal{P}_t^c = \{P_i^c\}_{i \in \mathcal{I}^t}$, and τ is used to keep the margin from growing to infinite. Thus, we have

$$\mathcal{L}_P^c = \sum_{i \in \mathcal{I}^t} -\log \frac{e^{-(\phi(P_i^c, \hat{P}^c)+\delta(t))}}{e^{-(\phi(P_i^c, \hat{P}^c)+\delta(t))} + \sum_{c'} e^{-\phi(P_i^c, \hat{P}^{c'})}}. \tag{10}$$

FedTGP Framework

We show the entire learning process of our FedTGP framework in Algorithm 1. With the well-trained separable global

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
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
FedTGP	**90.02±0.30**	**61.86±0.30**	**68.98±0.43**	**34.56±0.27**	**88.15±0.43**	**46.94±0.12**	**53.68±0.31**	**27.37±0.12**

Table 1: The test accuracy (%) on four datasets in the pathological and practical settings using the HtFE$_8$ model group.

prototypes, we send them to participating clients in the next iteration and guide client training with them to improve separability locally among feature representations of different classes by minimizing the client loss $\mathcal{L}_i$ for client i,

$$\mathcal{L}_i := \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i}\ell(h_i(f_i(\boldsymbol{x};\theta_i);w_i),y)+\lambda\mathbb{E}_{c\sim\mathcal{C}_i}\phi(P_i^c,\hat{P}^c), \tag{11}$$

which is similar to Eq. (4) but using the well-trained separable global prototypes $\hat{P}^c$ instead of $\bar{P}^c$. Following FedProto, we also utilize the global prototypes for inference on clients. Specifically, for a given input on one client, we calculate the ϕ distance between the feature representation and C global prototypes, and then this input belongs to the class of the closest global prototype.

Since our FedTGP follows the same communication protocol as FedProto by transmitting only compact 1D-class prototypes, it naturally brings benefits to both privacy preservation and communication efficiency. Specifically, no model parameter is shared and the generation of low-dimensional prototypes is irreversible, preventing data leakage from inversion attacks. In addition, our FedTGP does not require clients to upload the private class distribution information (i.e., $|\mathcal{D}_{i,c}|$ in Eq. (3)) to the server anymore, leading to less information revealed than FedProto.

Experiments

Setup

Datasets. We evaluate four popular image datasets for the multi-class classification tasks, including Cifar10 and Cifar100 (Krizhevsky and Geoffrey 2009), Tiny-ImageNet (Chrabaszcz, Loshchilov, and Hutter 2017) (100K images with 200 classes), and Flowers102 (Nilsback and Zisserman 2008) (8K images with 102 classes).

Baseline methods. To evaluate our proposed FedTGP, we compare it with six popular methods that are applicable in HtFL, including LG-FedAvg (Liang et al. 2020), Fed-Gen (Zhu, Hong, and Zhou 2021), FML (Shen et al. 2020), FedKD (Wu et al. 2022), FedDistill (Jeong et al. 2018), and FedProto (Tan et al. 2022b).

Model heterogeneity. Unless explicitly specified, we evaluate the model heterogeneity regarding Heterogeneous Feature Extractors (HtFE). We use "HtFE$_X$" to denote the HtFE setting, where X is the number of different model architectures in HtFL. We assign the $(i \mod X)$th model architecture to client i. For our main experiments, we use the

"HtFE$_8$" model group with eight architectures including the 4-layer CNN (McMahan et al. 2017), GoogleNet (Szegedy et al. 2015), MobileNet_v2 (Sandler et al. 2018), ResNet18, ResNet34, ResNet50, ResNet101, and ResNet152 (He et al. 2016). To generate feature representations with an identical feature dimension K, we add an average pooling layer (Szegedy et al. 2015) after each feature extractor. By default, we set $K = 512$.

Statistical heterogeneity. We conduct extensive experiments with two widely used statistically heterogeneous settings, the pathological setting (McMahan et al. 2017; Tan et al. 2022b) and the practical setting (Tan et al. 2022c; Li, He, and Song 2021; Zhu, Hong, and Zhou 2021). For the pathological setting, following FedAvg (McMahan et al. 2017), we distribute non-redundant and unbalanced data of 2/10/10/20 classes to each client from a total of 10/100/102/200 classes on Cifar10/Cifar100/Flowers102/Tiny-ImageNet datasets. For the practical setting, following MOON (Li, He, and Song 2021), we first sample $q_{c,i} \sim Dir(\beta)$ for class c and client i, then we assign $q_{c,i}$ proportion of data points from class c in a given dataset to client i, where $Dir(\beta)$ is the Dirichlet distribution and β is set to 0.1 by default (Lin et al. 2020).

Implementation Details. Unless explicitly specified, we use the following settings. We simulate a federation with 20 clients and a client participation ratio $\rho = 1$. Following FedAvg, we run one training epoch on each client in each iteration with a batch size of 10 and a learning rate $\eta = 0.01$ for 1000 communication iterations. We split the private data into a training set (75%) and a test set (25%) on each client. We average the results on clients' test sets and choose the best averaged result among iterations in each trial. For all the experiments, we run three trials and report the mean and standard deviation. We set $\lambda = 0.1$ (the same as FedProto), $\tau = 100$, and $S = 100$ (the number of server training epochs) for our FedTGP on all tasks. Please refer to the Appendix for more results and details.

Performance

As shown in Tab. 1, FedTGP outperforms all the baselines on four datasets by up to **9.08%** in accuracy. Specifically, using our TGP with ACL on the server, our FedTGP can improve FedProto by up to **13.85%**. The improvement is attributed to the enhanced separability of global prototypes. Besides, FedTGP shows better performance in relatively

Settings	Heterogeneous Feature Extractors				Heterogeneous Classifiers		Large Client Amount	
	$HtFE_2$	$HtFE_3$	$HtFE_4$	$HtFE_9$	$Res34\text{-}HtC_4$	$HtFE_8\text{-}HtC_4$	50 Clients	100 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
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
FML	45.94±0.16	43.05±0.06	43.00±0.08	42.41±0.28	41.03±0.20	39.23±0.42	38.47±0.14	36.09±0.28
FedKD	46.33±0.24	43.16±0.49	43.21±0.37	42.15±0.36	39.77±0.42	40.59±0.51	38.25±0.41	35.62±0.55
FedDistill	46.88±0.13	43.53±0.21	43.56±0.14	42.09±0.20	44.72±0.13	41.67±0.06	38.51±0.36	36.06±0.24
FedProto	43.97±0.18	38.14±0.64	34.67±0.55	32.74±0.82	32.26±0.18	25.57±0.72	33.03±0.42	28.95±0.51
FedTGP	**49.82±0.29**	**49.65±0.37**	**46.54±0.14**	**48.05±0.19**	**48.18±0.27**	**44.53±0.16**	**43.17±0.23**	**41.57±0.30**

Table 2: The test accuracy (%) on Cifar100 in the practical setting using heterogeneous feature extractors, heterogeneous classifiers, or a large number of clients ($\rho = 0.5$) with the $HtFE_8$ model group. "Res" is short for ResNet.

harder tasks with more classes, as more classes mean more client prototypes, which benefits our global prototype training. However, the generator in FedGen does not consistently yield improvements in HtFL, as FedGen cannot outperform LG-FedAvg in all cases in Tab. 1.

Impact of Model Heterogeneity

To examine the impact of model heterogeneity in HtFL, we assess the performance of FedTGP on four additional model groups with increasing model heterogeneity without changing the data distribution on clients: "$HtFE_2$" including the 4-layer CNN and ResNet18; "$HtFE_3$" including ResNet10 (Zhong et al. 2017), ResNet18, and ResNet34; "$HtFE_4$" including the 4-layer CNN, GoogleNet, MobileNet_v2, and ResNet18; "$HtFE_9$" including ResNet4, ResNet6, and ResNet8 (Zhong et al. 2017), ResNet10, ResNet18, ResNet34, ResNet50, ResNet101, and ResNet152. We show results in Tab. 2.

Our FedTGP consistently outperforms other FL methods across various model heterogeneities by up to **5.64%**, irrespective of the models' sizes. We observe that all methods perform worse with larger model heterogeneity in HtFL. However, our FedTGP only drops 1.77%, while the decrease for the counterparts is 3.53%∼15.04%, showing that our proposed TGP with ACL is more robust and less impacted by model heterogeneity.

We further evaluate the scenarios with four Heterogeneous Classifiers (HtC_4)[1] and create another two model groups: "$Res34\text{-}HtC_4$" uses the ResNet34 to build homogeneous feature extractors while both the feature extractors and classifiers are heterogeneous in "$HtFE_8\text{-}HtC_4$". We allocate classifiers to clients using the method introduced in $HtFE_X$. Since LG-FedAvg and FedGen require using homogeneous classifiers, these methods are not applicable here. In Tab. 2, our FedTGP still keeps the superiority in these scenarios. In the most heterogeneous scenario $HtFE_8\text{-}HtC_4$, our FedTGP surpasses FedProto by **18.96%** in accuracy with our proposed TGP and ACL.

Partial Participation with More Clients

Additionally, we evaluate our method on the Cifar100 dataset with 50 and 100 clients, respectively, using par-

[1]Please refer to the Appendix for the details.

tial client participation. When assigning Cifar100 to more clients using $HtFE_8$, the data amount on each client decreases, so all the methods perform worse with a larger client amount. Besides, we only sample half of the clients to participate in training in each iteration, i.e., $\rho = 0.5$. In Tab. 2, the superiority of our FedTGP is more obvious with more clients. Specifically, our FedTGP outperforms other methods by 4.66% and 5.48% with 50 clients and 100 clients, respectively.

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
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
FedTGP	**46.44±0.26**	**46.59±0.31**	**46.65±0.29**

Table 3: The test accuracy (%) on Cifar100 in the practical setting using the $HtFE_8$ model group with a different number of client training epochs (E).

During collaborative learning in FL, clients can alleviate the communication burden by conducting more client model training epochs before transmitting their updated models to the server (McMahan et al. 2017). However, we notice that increasing the number of client training epochs leads to reduced accuracy in methods such as FML and FedKD, which employ an auxiliary model. This decrease in accuracy can be attributed to the increased heterogeneity in the parameters of the shared auxiliary model before server aggregation. In contrast, other methods such as our proposed FedTGP, can maintain their performance with more client training epochs.

Impact of Feature Dimensions

We also vary the feature dimension K to evaluate its impact on model performance, as shown in Tab. 4. We find that most methods show better performance with increasing feature dimensions from $K = 64$ to $K = 256$, but the performance

	$K = 64$	$K = 256$	$K = 1024$
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
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
FedTGP	**46.28±0.59**	**46.30±0.39**	**45.98±0.38**

Table 4: The test accuracy (%) on Cifar100 in the practical setting using the HtFE$_8$ model group with different feature dimensions (K).

degrades with an excessively large feature dimension, such as $K = 1024$, as it becomes more challenging to train classifiers with too large feature dimension. In Tab. 4, our FedTGP achieves competitive performance with $K = 64$, while Fed-Proto lags by 6.45% compared to $K = 256$.

Communication Cost

	Theory	Practice
LG-FedAvg	$\sum_{i=1}^{M} \|w_i\| \times 2$	2.05M
FedGen	$\sum_{i=1}^{M}(\|w_i\| \times 2 + \|\Theta\|)$	8.69M
FML	$M \times (\|\theta_g\| + \|w_g\|) \times 2$	36.99M
FedKD	$M \times (\|\theta_g\| + \|w_g\|) \times 2 \times r$	33.04M
FedDistill	$\sum_{i=1}^{M} C \times (C_i + C)$	0.29M
FedProto	$\sum_{i=1}^{M} K \times (C_i + C)$	1.48M
FedTGP	$\sum_{i=1}^{M} K \times (C_i + C)$	1.48M

Table 5: The communication cost per iteration using the HtFE$_8$ model group on Cifar100 in the practical setting. Θ represents the parameters for the auxiliary generator in FedGen. θ_g and w_g denote the parameters of the auxiliary feature extractor and classifier, respectively, in FML and FedKD. r is a compression rate introduced by SVD for parameter factorization in FedKD. $\|\theta_g\| \gg K \times C$. C_i denotes the number of classes on client i. "M" is short for million.

We show the communication cost in Tab. 5. Specifically, we calculate the communication cost in both theory and practice. In Tab. 5 FML and FedKD cost the most overhead in communication as they additionally transmit an auxiliary model. Although FedKD reduces the communication overhead through singular value decomposition (SVD) on the auxiliary model parameters, its communication cost is still much larger than prototype-based methods. In FedGen, downloading the generator from the server brings noticeable communication overhead. Although FedDistill costs $5.12\times$ less communication overhead than our FedTGP, the information capacity of the logits is $5.12\times$ less than the prototypes, so FedDistill achieves lower accuracy than FedTGP. In summary, our FedTGP achieves higher accuracy while preserving communication-efficient characteristics.

Ablation and Hyperparameter Study

	SCL	FM	w/o F	Proto	TGP
Cifar100	40.11	43.46	40.37	36.34	**46.94**
Flowers102	46.81	52.03	49.39	41.21	**53.68**
Tiny-ImageNet	22.26	26.13	23.12	19.01	**27.37**

Table 6: The test accuracy (%) in the practical setting using the HtFE$_8$ model group for ablation study. "Fed" is omitted in the method name due to limited space.

We replace ACL with the standard contrastive loss (Eq. (6)), denoted by "SCL". Besides, we modify ACL and TGP by using a fixed margin (Eq. (7)) and removing the further processing model F but only train $\{\acute{P}^c\}_{c=1}^{C}$, denoted by "FM" and "w/o F", respectively. According to Tab. 6, without utilizing a margin to improve separability, SCL shows a mere improvement of 5.60% for FedProto on Cifar100, whereas the improvement reaches 10.82% for FM with a margin. Nevertheless, our adaptive margin can further enhance FM and improve 12.47% for FedProto on Cifar100. Without sufficient trainable parameters in TGP, the performance of w/o F decreases up to 6.57% compared to our FedTGP, but it still outperforms FedProto by a large gap.

Different τ				Different S			
1	10	100	1000	1	10	100	1000
43.23	44.81	**46.94**	46.09	43.41	44.62	46.94	**47.01**

Table 7: The test accuracy (%) on Cifar100 in the practical setting using the HtFE$_8$ model group with different τ or S. Recall that we set $\tau = 100$ and $S = 100$ by default.

We study the hyperparameters of FedTGP by varying the hyperparameters τ and S in our FedTGP, and the results are shown in Tab. 7. Our FedTGP performs better with a larger threshold τ ranging from 1 to 100. However, the accuracy slightly drops when using $\tau = 1000$, because an excessively large τ leads to unstable prototype guidance on clients, and $\delta(t)$ may keep growing during the later stage of training. Unlike τ, increasing the number of server training epochs S leads to higher accuracy in our FedTGP. As the improvement from $S = 100$ to $S = 1000$ is negligible, we adopt $S = 100$ to save computation. Even with $\tau = 1$ or $S = 1$, our FedTGP can achieve at least 43.23% in accuracy, which is still higher than baseline methods' accuracy as shown in Tab. 1 (Practical setting, Cifar100) but setting $S = 1$ can save a lot of computation.

Conclusion

In this work, we propose a novel HtFL method called FedTGP, which shares class-wise prototypes among the server and clients and enhances the separability of different classes via our TGP and ACL. Extensive experiments with two statistically heterogeneous settings and twelve heterogeneous models show the superiority of our FedTGP over other baseline methods.

Acknowledgments

References

Chen, H.-Y.; and Chao, W.-L. 2021. On Bridging Generic and Personalized Federated Learning for Image Classification. In *ICLR*.

Choi, H.; Som, A.; and Turaga, P. 2020. AMC-loss: Angular margin contrastive loss for improved explainability in image classification. In *CVPR Workshop*.

Chrabaszcz, P.; Loshchilov, I.; and Hutter, F. 2017. A Downsampled Variant of Imagenet as an Alternative to the Cifar Datasets. *arXiv preprint arXiv:1707.08819*.

Collins, L.; Hassani, H.; Mokhtari, A.; and Shakkottai, S. 2021. Exploiting Shared Representations for Personalized Federated Learning. In *ICML*.

Deng, J.; Guo, J.; Xue, N.; and Zafeiriou, S. 2019. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*.

Diao, E.; Ding, J.; and Tarokh, V. 2020. HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients. In *ICLR*.

Hayat, M.; Khan, S.; Zamir, S. W.; Shen, J.; and Shao, L. 2019. Gaussian affinity for max-margin class imbalanced learning. In *ICCV*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Horvath, S.; Laskaridis, S.; Almeida, M.; Leontiadis, I.; Venieris, S.; and Lane, N. 2021. Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout. *NeurIPS*.

Jeong, E.; Oh, S.; Kim, H.; Park, J.; Bennis, M.; and Kim, S.-L. 2018. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*.

Jin, X.-B.; Liu, C.-L.; and Hou, X. 2010. Regularized margin-based conditional log-likelihood loss for prototype learning. *Pattern Recognition*, 43(7): 2428–2438.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2019. Advances and Open Problems in Federated Learning. *arXiv preprint arXiv:1912.04977*.

Kim, T.; and Kim, C. 2020. Attract, perturb, and explore: Learning a feature alignment network for semi-supervised domain adaptation. In *ECCV*.

Krizhevsky, A.; and Geoffrey, H. 2009. Learning Multiple Layers of Features From Tiny Images. *Technical Report*.

Li, D.; and Wang, J. 2019. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*.

Li, Q.; Diao, Y.; Chen, Q.; and He, B. 2022. Federated Learning on Non-IID Data Silos: An Experimental Study. In *ICDE*.

Li, Q.; He, B.; and Song, D. 2021. Model-Contrastive Federated Learning. In *CVPR*.

Li, Q.; Wen, Z.; Wu, Z.; Hu, S.; Wang, N.; Li, Y.; Liu, X.; and He, B. 2021a. A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection. *IEEE Transactions on Knowledge and Data Engineering*.

Li, T.; Hu, S.; Beirami, A.; and Smith, V. 2021b. Ditto: Fair and Robust Federated Learning Through Personalization. In *ICML*.

Li, T.; Sahu, A. K.; Talwalkar, A.; and Smith, V. 2020. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine*, 37(3): 50–60.

Li, Z.; Shang, X.; He, R.; Lin, T.; and Wu, C. 2023a. No Fear of Classifier Biases: Neural Collapse Inspired Federated Learning with Synthetic and Fixed Classifier. *arXiv preprint arXiv:2303.10058*.

Li, Z.; Wang, X.; Robertson, N. M.; Clifton, D. A.; Meinel, C.; and Yang, H. 2023b. SMKD: Selective Mutual Knowledge Distillation. In *IJCNN*.

Liang, P. P.; Liu, T.; Ziyin, L.; Allen, N. B.; Auerbach, R. P.; Brent, D.; Salakhutdinov, R.; and Morency, L.-P. 2020. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*.

Liao, Y.; Ma, L.; Zhou, B.; Zhao, X.; and Xie, F. 2023. DraftFed: A Draft-Based Personalized Federated Learning Approach for Heterogeneous Convolutional Neural Networks. *IEEE Transactions on Mobile Computing*.

Lin, T.; Kong, L.; Stich, S. U.; and Jaggi, M. 2020. Ensemble distillation for robust model fusion in federated learning. *NeurIPS*.

Luo, M.; Chen, F.; Hu, D.; Zhang, Y.; Liang, J.; and Feng, J. 2021. No Fear of Heterogeneity: Classifier Calibration for Federated Learning with Non-IID data. In *NeurIPS*.

Ma, X.; Zhang, J.; Guo, S.; and Xu, W. 2022. Layer-wised model aggregation for personalized federated learning. In *CVPR*.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*.

Nilsback, M.-E.; and Zisserman, A. 2008. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, 722–729. IEEE.

Pinheiro, P. O. 2018. Unsupervised domain adaptation with similarity learning. In *CVPR*.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*.

Schroff, F.; Kalenichenko, D.; and Philbin, J. 2015. Facenet: A unified embedding for face recognition and clustering. In *CVPR*.

Shamsian, A.; Navon, A.; Fetaya, E.; and Chechik, G. 2021. Personalized federated learning using hypernetworks. In *ICML*.

Shen, T.; Zhang, J.; Jia, X.; Zhang, F.; Huang, G.; Zhou, P.; Kuang, K.; Wu, F.; and Wu, C. 2020. Federated mutual learning. *arXiv preprint arXiv:2006.16765*.

Shin, K.; Kwak, H.; Kim, S. Y.; Ramström, M. N.; Jeong, J.; Ha, J.-W.; and Kim, K.-M. 2023. Scaling law for recommendation models: Towards general-purpose user representations. In *AAAI*.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *CVPR*.

T Dinh, C.; Tran, N.; and Nguyen, T. D. 2020. Personalized Federated Learning with Moreau Envelopes. In *NeurIPS*.

Tan, A. Z.; Yu, H.; Cui, L.; and Yang, Q. 2022a. Towards Personalized Federated Learning. *IEEE Transactions on Neural Networks and Learning Systems*. Early Access.

Tan, Y.; Long, G.; Liu, L.; Zhou, T.; Lu, Q.; Jiang, J.; and Zhang, C. 2022b. Fedproto: Federated Prototype Learning across Heterogeneous Clients. In *AAAI*.

Tan, Y.; Long, G.; Ma, J.; Liu, L.; Zhou, T.; and Jiang, J. 2022c. Federated Learning from Pre-Trained Models: A Contrastive Learning Approach. *arXiv preprint arXiv:2209.10083*.

Tanwisuth, K.; Fan, X.; Zheng, H.; Zhang, S.; Zhang, H.; Chen, B.; and Zhou, M. 2021. A prototype-oriented framework for unsupervised domain adaptation. *NeurIPS*.

Wang, H.; Yurochkin, M.; Sun, Y.; Papailiopoulos, D.; and Khazaeni, Y. 2020. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*.

Wang, L.; Wang, M.; Zhang, D.; and Fu, H. 2023. Model Barrier: A Compact Un-Transferable Isolation Domain for Model Intellectual Property Protection. In *CVPR*.

Wen, D.; Jeon, K.-J.; and Huang, K. 2022. Federated dropout—A simple approach for enabling federated learning on resource constrained devices. *IEEE wireless communications letters*, 11(5): 923–927.

Wu, C.; Wu, F.; Lyu, L.; Huang, Y.; and Xie, X. 2022. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1): 2032.

Xu, W.; Xian, Y.; Wang, J.; Schiele, B.; and Akata, Z. 2020. Attribute prototype network for zero-shot learning. *NeurIPS*.

Yang, H.-M.; Zhang, X.-Y.; Yin, F.; and Liu, C.-L. 2018. Robust classification with convolutional prototype learning. In *CVPR*.

Yang, X.; Huang, W.; and Ye, M. 2023. Dynamic Personalized Federated Learning with Adaptive Differential Privacy. In *NeurIPS*.

Yi, L.; Wang, G.; Liu, X.; Shi, Z.; and Yu, H. 2023. FedGH: Heterogeneous Federated Learning with Generalized Global Header. *arXiv preprint arXiv:2303.13137*.

Yu, Q.; Liu, Y.; Wang, Y.; Xu, K.; and Liu, J. 2022. Multimodal Federated Learning via Contrastive Representation Ensemble. In *ICLR*.

Zhang, J.; Gu, Z.; Jang, J.; Wu, H.; Stoecklin, M. P.; Huang, H.; and Molloy, I. 2018a. Protecting intellectual property of deep neural networks with watermarking. In *ASIA-CCS*.

Zhang, J.; Guo, S.; Guo, J.; Zeng, D.; Zhou, J.; and Zomaya, A. 2023a. Towards Data-Independent Knowledge Transfer in Model-Heterogeneous Federated Learning. *IEEE Transactions on Computers*.

Zhang, J.; Guo, S.; Ma, X.; Wang, H.; Xu, W.; and Wu, F. 2021. Parameterized Knowledge Transfer for Personalized Federated Learning. In *NeurIPS*.

Zhang, J.; Hua, Y.; Cao, J.; Wang, H.; Song, T.; XUE, Z.; Ma, R.; and Guan, H. 2023b. Eliminating Domain Bias for Federated Learning in Representation Space. In *NeurIPS*.

Zhang, J.; Hua, Y.; Wang, H.; Song, T.; Xue, Z.; Ma, R.; Cao, J.; and Guan, H. 2023c. GPFL: Simultaneously Learning Global and Personalized Feature Information for Personalized Federated Learning. In *ICCV*.

Zhang, J.; Hua, Y.; Wang, H.; Song, T.; Xue, Z.; Ma, R.; and Guan, H. 2023d. FedALA: Adaptive Local Aggregation for Personalized Federated Learning. In *AAAI*.

Zhang, J.; Hua, Y.; Wang, H.; Song, T.; Xue, Z.; Ma, R.; and Guan, H. 2023e. FedCP: Separating Feature Information for Personalized Federated Learning via Conditional Policy. In *KDD*.

Zhang, K.; and Sato, Y. 2023. Semantic Image Segmentation by Dynamic Discriminative Prototypes. *IEEE Transactions on Multimedia*.

Zhang, L.; Shen, L.; Ding, L.; Tao, D.; and Duan, L.-Y. 2022. Fine-Tuning Global Model Via Data-Free Knowledge Distillation for Non-IID Federated Learning. In *CVPR*.

Zhang, Y.; Xiang, T.; Hospedales, T. M.; and Lu, H. 2018b. Deep mutual learning. In *CVPR*.

Zhong, Z.; Li, J.; Ma, L.; Jiang, H.; and Zhao, H. 2017. Deep residual networks for hyperspectral image classification. In *IEEE international geoscience and remote sensing symposium (IGARSS)*.

Zhu, Z.; Hong, J.; and Zhou, J. 2021. Data-Free Knowledge Distillation for Heterogeneous Federated Learning. In *ICML*.

Zhuang, W.; Chen, C.; and Lyu, L. 2023. When Foundation Model Meets Federated Learning: Motivations, Challenges, and Future Directions. *arXiv preprint arXiv:2306.15546*.