Eliminating Domain Bias for Federated Learning in Representation Space

Jianqing Zhang[1], **Yang Hua**[2], **Jian Cao**[1*], **Hao Wang**[3],
Tao Song[1] **Zhengui Xue**[1], **Ruhui Ma**[1*], **Haibing Guan**[1]
[1]Shanghai Jiao Tong University [2]Queen's University Belfast [3]Louisiana State University
`{tsingz, cao-jian, songt333, zhenguixue, ruhuima, hbguan}@sjtu.edu.cn`
`Y.Hua@qub.ac.uk, haowang@lsu.edu`

We provide more details and results about our work in the appendices. Here are the contents:

- Appendix A: The extended version of the Related Work section in the main body.
- Appendix B: Proofs of Corollary 1 and Corollary 2.
- Appendix C: More details about experimental settings.
- Appendix D: Additional experiments (*e.g.*, a real-world application).
- Appendix E: Broader impacts of our proposed method.
- Appendix F: Limitations of our proposed method.
- Appendix G: Data distribution visualizations for different scenarios in our experiments.

A Related Work

As the number of users and sensors rapidly increases with massive growing services on the Internet, the privacy concerns about private data also draw increasing attention of researchers [34, 59, 61]. Then a new distributed machine learning paradigm, federated learning (FL), comes along with the privacy-preserving and collaborative learning abilities [34, 50, 63]. Although there are horizontal FL [41, 50, 63], vertical FL [47, 56, 63], federated transfer learning[14, 45], *etc*., we focus on the popular horizontal FL and call it FL for short in this paper.

Traditional FL methods concentrate on learning a single global model among a server and clients, but it suffers an accuracy decrease under statistically heterogeneous scenarios, which are common scenarios in practice [42, 50, 58, 66]. Then, many FL methods propose learning personalized models (or modules) for each client besides learning the global model. These FL methods are specifically called personalized FL (pFL) methods [18, 20, 60].

A.1 Traditional Federated Learning

FL methods perform machine learning through iterative communication and computation on the server and clients. To begin with, we describe the FL procedure in one iteration based on FedAvg [50], which is a famous FL method and a basic framework for later FL methods. The FL procedure includes five stages: (1) A server selects a group of clients to join FL in this iteration and sends the current global model to them; (2) these clients receive the global model and initialize their local model by overwriting their local model with the parameters in the global model; (3) these clients train their local models on their own private local data, respectively; (4) these clients send the trained local models to the server; (5) the server receives client models and aggregates them through weighted averaging on model parameters to obtain a new global model.

---

[*] Corresponding author.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Then, massive traditional FL methods are proposed in the literature to improve FedAvg regarding privacy-preserving [43, 52, 70], accuracy [35, 40, 71], fairness [30, 64], overhead [26, 38, 48], *etc*. Here, we focus on the representative traditional FL methods that handle the heterogeneity issues in four categories: update-correction-based FL [23, 35, 54], regularization-based FL [1, 16, 37, 41], model-split-based FL [33, 40], and knowledge-distillation-based FL [25, 31, 67, 71].

Among **update-correction-based FL** methods, SCAFFOLD [35] witnesses the client-drift phenomenon of FedAvg under statistically heterogeneous scenarios due to local training and proposes correcting local update through control variates for each model parameter. Among **regularization-based FL** methods, FedProx [41] modifies the local objective on each client by adding a regularization term to keep local model parameters close to the global model during local training in an element-wise manner. Among **model-split-based FL** methods, MOON [40] observes that local training degenerates representation quality, so it adds a contrastive learning term to let the representations outputted by the local feature extractor be close to the ones outputted by the received global feature extractor given each input during local training. However, input-wise contrastive learning relies on biased local data domains, so MOON still suffers from representation bias. Among **knowledge-distillation-based FL** methods, FedGen [71] learns a generator on the server to produce additional representations, shares the generator among clients, and locally trains the classifier with the combination of the representations outputted by the local feature extractor and the additionally generated representations. In this way, FedGen can reduce the heterogeneity among clients with the augmented representations from the shared generator via knowledge distillation. However, it only considers the local-to-global knowledge transfer for the single global model learning and additionally brings communication and computation overhead for learning and transmitting the generator.

A.2 Personalized Federated Learning

Different from traditional FL, pFL additionally learns personalized models (or modules) besides the global model. In this paper, we consider pFL methods in four categories: meta-learning-based pFL [12, 20], regularization-based pFL [42, 58], personalized-aggregation-based pFL [19, 46, 68], and model-split-based pFL [3, 13, 18, 55].

Meta-learning-based pFL. Meta-learning is a technique that trains deep neural networks (DNNs) on a given dataset for quickly adapting to other datasets with only a few steps of fine-tuning, *e.g.*, MAML [22]. By integrating MAML into FL, Per-FedAvg [20] updates the local models like MAML to capture the learning trends of each client and then aggregates the learning trends by averaging on the server. It obtains personalized models by fine-tuning the global model for each client. Similar to Per-FedAvg, FedMeta [12] also introduces MAML on each client during training and fine-tuning the global model for evaluation. However, it is hard for these meta-learning-based pFL methods to find a consensus learning trend through averaging under statistically heterogeneous scenarios.

Regularization-based pFL. Like FedProx, pFedMe [58] and Ditto [42] also utilize the regularization technique, but they modify the objective for additional personalized model training rather than the one for local model training. In pFedMe and Ditto, each client owns two models: the local model that is trained for global model aggregation and the personalized model that is trained for personalization. Specifically, pFedMe regularizes the model parameters between the personalized model and the local model during training while Ditto regularizes the model parameters between the personalized model and the received global model. Besides, Ditto simply trains the local model similar to FedAvg while pFedMe trains the local model based on the personalized model. Although the local model is initialized by the global model, but the initialized local model gradually loses global information during local training. Thus, the personalized model in Ditto can be aware of more global information than the one in pFedMe. Both pFedMe and Ditto require additional memory space to store the personalized model and double the computation resources at least to train both the local model and the personalized model.

Personalized-aggregation-based pFL. These pFL methods adaptively aggregate the global model and local model according to the local data on each client, *e.g.*, APFL [19], or directly generate the personalized model using other client models through personalized aggregation on each client, *e.g.*, FedFomo [68] and APPLE [46]. Specifically, APFL aggregates the parameters in the global model and the local model with weighted averaging and adaptively updates the scalar weight based on the gradients. On each client, FedFomo generates the client-specific aggregating weights for the received client models through first-order approximation while APPLE adaptively learns these weights based on the local data. Both FedFomo and APPLE require multiple communication overhead than other FL methods, but FedFomo costs less computation overhead than APPLE attributed to approximation.

Model-split-based pFL. These pFL methods split a given model into a feature extractor and a classifier. They treat the feature extractor and the classifier differently. Concretely, FedPer [3] and FedRep [18] keep the classifier locally on each client. FedPer trains the feature extractor and the classifier together while FedRep first fine-tunes the classifier and then trains the feature extractor in each iteration. For FedPer and FedRep, the feature extractor intends to extract representations to cater to these personalized classifiers, thus reducing the generic representation quality. FedRoD [13] trains the local model with the balanced softmax (BSM) loss function [57] and simultaneously learns an additional personalized classifier for each client. However, the BSM loss is useless for missing labels on each client while label missing is a common situation in statistically heterogeneous scenarios [44, 66, 68]. Moreover, the uniform label distribution modified by the BSM cannot reflect the original distribution. The above pFL methods learn personalized models (or modules) in FL, but FedBABU [55] firstly trains the global feature extractor with the frozen classifier during the FL process, then it fine-tunes the global model on each client after FL to obtain personalized models. However, this post-FL fine-tuning is beyond the scope of FL. Almost all the FL methods have multiple fine-tuning variants, *e.g.*, fine-tuning the whole model or only a part of the model. Furthermore, training the feature extractor with the naive and randomly initialized classifier in FL has an uncontrollable risk due to randomness.

B Theoretical Derivations

B.1 Notations and Preliminaries

Following prior arts [19, 49, 60, 71], we consider a binary classification problem in FL here. Recall that $\mathcal{X} \subset \mathbb{R}^D$ is an input space, $\mathcal{Z} \subset \mathbb{R}^K$ is a representation space, and $\mathcal{Y} \subset \{0, 1\}$ is a label space. Let $\mathcal{F} : \mathcal{X} \mapsto \mathcal{Z}$ be a representation function that maps from the input space to the representation space. We denote $\mathcal{D} := \langle \mathcal{U}, c^* \rangle$ as a data domain where the distribution $\mathcal{U} \subseteq \mathcal{X}$ and $c^* : \mathcal{X} \mapsto \mathcal{Y}$ is a ground-truth labeling function. $\tilde{\mathcal{U}}$ is the induced distribution of $\mathcal{U}$ over the representation space $\mathcal{Z}$ under $\mathcal{F}$ [6], *i.e.*, $\tilde{\mathcal{U}} \subseteq \mathcal{Z}$, that satisfies

$$\mathbb{E}_{\boldsymbol{z} \sim \tilde{\mathcal{U}}} \left[\mathcal{B}\left(\boldsymbol{z}\right)\right] = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{U}} \left[\mathcal{B}\left(\mathcal{F}\left(\boldsymbol{x}\right)\right)\right], \tag{1}$$

where $\mathcal{B}$ is a probability event. Given fixed but unknown $\mathcal{U}$ and c^*, the learning task on one domain is to choose a representation function $\mathcal{F}$ and a hypothesis class $\mathcal{H} \subseteq \{h : \mathcal{Z} \mapsto \mathcal{Y}\}$ to approximate the function c^*.

Then, we provide the definition and theorem from Ben-David et al. [6, 7], Blitzer et al. [8], Kifer et al. [36] under their assumptions:

Definition 1. *If a space $\mathcal{Z}$ with $\tilde{\mathcal{U}}^a$ and $\tilde{\mathcal{U}}^b$ distributions over $\mathcal{Z}$, let $\mathcal{H}$ be a hypothesis class on $\mathcal{Z}$ and $\mathcal{Z}_h \subseteq \mathcal{Z}$ be the subset with characteristic function h, the $\mathcal{H}$-divergence between $\tilde{\mathcal{U}}^a$ and $\tilde{\mathcal{U}}^b$ is*

$$d_{\mathcal{H}}\left(\tilde{\mathcal{U}}^a, \tilde{\mathcal{U}}^b\right) = 2 \sup_{h \in \mathcal{H}} \left|\Pr_{\tilde{\mathcal{U}}^a}\left[\mathcal{Z}_h\right] - \Pr_{\tilde{\mathcal{U}}^b}\left[\mathcal{Z}_h\right]\right|,$$

where $\mathcal{Z}_h = \{\boldsymbol{z} \in \mathcal{Z} : h\left(\boldsymbol{z}\right) = 1\}, h \in \mathcal{H}$.

Definition 1 implies that $d_{\mathcal{H}}\left(\tilde{\mathcal{U}}^a, \tilde{\mathcal{U}}^b\right) = d_{\mathcal{H}}\left(\tilde{\mathcal{U}}^b, \tilde{\mathcal{U}}^a\right)$.

Theorem 1. *Consider a source domain $\mathcal{D}_S$ and a target domain $\mathcal{D}_T$. Let $\mathcal{D}_S = \langle \mathcal{U}_S, c^* \rangle$ and $\mathcal{D}_T = \langle \mathcal{U}_T, c^* \rangle$, where $\mathcal{U}_S \subseteq \mathcal{X}$, $\mathcal{U}_T \subseteq \mathcal{X}$, and $c^* : \mathcal{X} \mapsto \mathcal{Y}$ is a ground-truth labeling function. Let $\mathcal{H}$ be a hypothesis space of VC dimension d and $h : \mathcal{Z} \mapsto \mathcal{Y}, \forall\, h \in \mathcal{H}$. Given a feature extraction function $\mathcal{F} : \mathcal{X} \mapsto \mathcal{Z}$ that shared between $\mathcal{D}_S$ and $\mathcal{D}_T$, a random labeled sample of size m generated by applying $\mathcal{F}$ to a random sample from $\mathcal{U}_S$ labeled according to c^*, then for every $h \in \mathcal{H}$, with probability at least $1 - \delta$:*

$$\mathcal{L}_{\mathcal{D}_T}\left(h\right) \leq \mathcal{L}_{\hat{\mathcal{D}}_S}\left(h\right) + \sqrt{\frac{4}{m}\left(d \log \frac{2em}{d} + \log \frac{4}{\delta}\right)} + d_{\mathcal{H}}\left(\tilde{\mathcal{U}}_S, \tilde{\mathcal{U}}_T\right) + \lambda,$$

where $\mathcal{L}_{\hat{\mathcal{D}}_S}$ is the empirical loss on $\mathcal{D}_S$, e is the base of the natural logarithm, and $d_{\mathcal{H}}\left(\cdot, \cdot\right)$ is the $\mathcal{H}$-divergence between two distributions. $\tilde{\mathcal{U}}_S$ and $\tilde{\mathcal{U}}_T$ are the induced distributions of $\mathcal{U}_S$ and $\mathcal{U}_T$

under $\mathcal{F}$, respectively, s.t. $\mathbb{E}_{\boldsymbol{z}\sim\tilde{\mathcal{U}}_S}\left[\mathcal{B}\left(\boldsymbol{z}\right)\right] = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{U}_S}\left[\mathcal{B}\left(\mathcal{F}\left(\boldsymbol{x}\right)\right)\right]$ given a probability event $\mathcal{B}$, and so for $\tilde{\mathcal{U}}_T$. $\tilde{\mathcal{U}}_S \subseteq \mathcal{Z}$ and $\tilde{\mathcal{U}}_T \subseteq \mathcal{Z}$. $\lambda := \min_h \mathcal{L}_{\mathcal{D}_S}\left(h\right) + \mathcal{L}_{\mathcal{D}_T}\left(h\right)$ denotes an oracle performance.

The traditional FL methods, which focus on enhancing the performance of a global model, regard local domains $\mathcal{D}_i, i \in [N]$ and the virtual global domain $\mathcal{D}$ as the source domain and the target domain, respectively [71], which is called local-to-global knowledge transfer by us. In contrast, pFL methods that focus on improving the performance of personalized models regard $\mathcal{D}$ and $\mathcal{D}_i, i \in [N]$ as the source domain and the target domain, respectively [19, 49, 60]. We call this kind of adaptation global-to-local knowledge transfer. The local-to-global knowledge transfer happens on the server while the global-to-local one occurs on the client.

B.2 Derivations of Corollary 1

As we focus on the local-to-global knowledge transfer on the *server side*, in the FL scenario, we can rewrite Theorem 1 to

Theorem 2. *Consider a local data domain $\mathcal{D}_i$ and a virtual global data domain $\mathcal{D}$. Let $\mathcal{D}_i = \langle \mathcal{U}_i, c^* \rangle$ and $\mathcal{D} = \langle \mathcal{U}, c^* \rangle$, where $\mathcal{U}_i \subseteq \mathcal{X}$ and $\mathcal{U} \subseteq \mathcal{X}$. Given a feature extraction function $\mathcal{F} : \mathcal{X} \mapsto \mathcal{Z}$ that shared between $\mathcal{D}_i$ and $\mathcal{D}$, a random labeled sample of size m generated by applying $\mathcal{F}$ to a random sample from $\mathcal{U}_i$ labeled according to c^*, then for every $h \in \mathcal{H}$, with probability at least $1-\delta$:*

$$\mathcal{L}_{\mathcal{D}}\left(h\right) \leq \mathcal{L}_{\hat{\mathcal{D}}_i}\left(h\right) + \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4}{\delta}\right)} + d_{\mathcal{H}}\left(\tilde{\mathcal{U}}_i, \tilde{\mathcal{U}}\right) + \lambda_i,$$

where $\tilde{\mathcal{U}}_i$ and $\tilde{\mathcal{U}}$ are the induced distributions of $\mathcal{U}_i$ and $\mathcal{U}$ under $\mathcal{F}$, respectively. $\tilde{\mathcal{U}}_i \subseteq \mathcal{Z}$ and $\tilde{\mathcal{U}} \subseteq \mathcal{Z}$. $\lambda_i := \min_h \mathcal{L}_{\mathcal{D}_i}\left(h\right) + \mathcal{L}_{\mathcal{D}}\left(h\right)$ denotes an oracle performance.

Corollary 1. *Consider a local data domain $\mathcal{D}_i$ and a virtual global data domain $\mathcal{D}$ for client i and the server, respectively. Let $\mathcal{D}_i = \langle \mathcal{U}_i, c^* \rangle$ and $\mathcal{D} = \langle \mathcal{U}, c^* \rangle$, where $c^* : \mathcal{X} \mapsto \mathcal{Y}$ is a ground-truth labeling function. Let $\mathcal{H}$ be a hypothesis space of VC dimension d and $h : \mathcal{Z} \mapsto \mathcal{Y}, \forall\, h \in \mathcal{H}$. When using `DBE`, given a feature extraction function $\mathcal{F}^g : \mathcal{X} \mapsto \mathcal{Z}$ that shared between $\mathcal{D}_i$ and $\mathcal{D}$, a random labeled sample of size m generated by applying $\mathcal{F}^g$ to a random sample from $\mathcal{U}_i$ labeled according to c^*, then for every $h^g \in \mathcal{H}$, with probability at least $1-\delta$:*

$$\mathcal{L}_{\mathcal{D}}\left(h^g\right) \leq \mathcal{L}_{\hat{\mathcal{D}}_i}\left(h^g\right) + \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4}{\delta}\right)} + d_{\mathcal{H}}\left(\tilde{\mathcal{U}}^g_i, \tilde{\mathcal{U}}^g\right) + \lambda_i,$$

where$\mathcal{L}_{\hat{\mathcal{D}}_i}$ is the empirical loss on $\mathcal{D}_i$, e is the base of the natural logarithm, and $d_{\mathcal{H}}\left(\cdot,\cdot\right)$ is the $\mathcal{H}$-divergence between two distributions. $\lambda_i := \min_{h^g} \mathcal{L}_{\mathcal{D}}\left(h^g\right) + \mathcal{L}_{\mathcal{D}_i}\left(h^g\right)$, $\tilde{\mathcal{U}}^g_i \subseteq \mathcal{Z}$, $\tilde{\mathcal{U}}^g \subseteq \mathcal{Z}$, and $d_{\mathcal{H}}\left(\tilde{\mathcal{U}}^g_i, \tilde{\mathcal{U}}^g\right) \leq d_{\mathcal{H}}\left(\tilde{\mathcal{U}}_i, \tilde{\mathcal{U}}\right)$. $\tilde{\mathcal{U}}^g_i$ and $\tilde{\mathcal{U}}^g$ are the induced distributions of $\mathcal{U}_i$ and $\mathcal{U}$ under $\mathcal{F}^g$, respectively. $\tilde{\mathcal{U}}_i$ and $\tilde{\mathcal{U}}$ are the induced distributions of $\mathcal{U}_i$ and $\mathcal{U}$ under $\mathcal{F}$, respectively. $\mathcal{F}$ is the feature extraction function in the original FedAvg without `DBE`.

Proof. Computing $d_{\mathcal{H}}\left(\cdot,\cdot\right)$ is identical to learning a classifier to achieve a minimum error of discriminating between points sampled from $\tilde{\mathcal{U}}$ and $\tilde{\mathcal{U}}'$, *i.e.*, a binary domain classification problem [6, 7]. The more difficult the domain classification problem is, the smaller $d_{\mathcal{H}}\left(\cdot,\cdot\right)$ is. Unfortunately, computing the error of the optimal hyperplane classifier for arbitrary distributions is a well-known NP-hard problem [5, 6]. Thus, researchers approximate the error by learning a linear classifier for the binary domain classification [5, 8, 9]. Inspired by previous approaches [4, 39, 51], we consider using Linear Discriminant Analysis (LDA) for the binary domain classification. The discrimination ability of LDA is measured by the Fisher discriminant ratio (F1) [10, 28, 62]

$$F1\left(\tilde{\mathcal{U}}^a, \tilde{\mathcal{U}}^b\right) = \max_k \left[\frac{\left(\boldsymbol{\mu}^k_{\tilde{\mathcal{U}}^a} - \boldsymbol{\mu}^k_{\tilde{\mathcal{U}}^b}\right)^2}{\left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}^a}\right)^2 + \left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}^b}\right)^2}\right],$$

where $\boldsymbol{\mu}^k_{\tilde{\mathcal{U}}^a}$ and $\left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}^a}\right)^2$ are the mean and variance of the values in the kth dimension over $\tilde{\mathcal{U}}^a$. The smaller the Fisher discriminant ratio is, the less discriminative the two domains are. Theorem 2 holds with every $h \in \mathcal{H}$, so we omit `PRBM` here. `MR` $(\bar{z}_i^g, \bar{z}^g)$ forces the local domain to be close to the global domain in terms of the mean value at each feature dimension in the feature representation independently, therefore, $\forall\, k \in [K]$,

$$\boldsymbol{\mu}^k_{\tilde{\mathcal{U}}_i^g} - \boldsymbol{\mu}^k_{\tilde{\mathcal{U}}^g} \le \boldsymbol{\mu}^k_{\tilde{\mathcal{U}}_i} - \boldsymbol{\mu}^k_{\tilde{\mathcal{U}}}.$$

As the feature extractors share the same structure with identical parameter initialization and the feature representations are extracted from the same data domain $\mathcal{D}_i$ $(\mathcal{D})$ [17, 32], we assume that $\boldsymbol{\sigma}_{\tilde{\mathcal{U}}_i^g} = \boldsymbol{\sigma}_{\tilde{\mathcal{U}}_i}$ and $\boldsymbol{\sigma}_{\tilde{\mathcal{U}}^g} = \boldsymbol{\sigma}_{\tilde{\mathcal{U}}}$. Thus, $\forall\, k \in [K]$,

$$\frac{\left(\boldsymbol{\mu}^k_{\tilde{\mathcal{U}}_i^g} - \boldsymbol{\mu}^k_{\tilde{\mathcal{U}}^g}\right)^2}{\left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}_i^g}\right)^2 + \left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}^g}\right)^2} \le \frac{\left(\boldsymbol{\mu}^k_{\tilde{\mathcal{U}}_i} - \boldsymbol{\mu}^k_{\tilde{\mathcal{U}}}\right)^2}{\left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}_i}\right)^2 + \left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}}\right)^2}.$$

As this inequality is satisfied in all dimensions including the dimension where the maximum value exists, so for the Fisher discriminant ratio, we have

$$F1\left(\tilde{\mathcal{U}}_i^g, \tilde{\mathcal{U}}^g\right) = \max_k \left[\frac{\left(\boldsymbol{\mu}^k_{\tilde{\mathcal{U}}_i^g} - \boldsymbol{\mu}^k_{\tilde{\mathcal{U}}^g}\right)^2}{\left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}_i^g}\right)^2 + \left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}^g}\right)^2}\right] \le \max_k \left[\frac{\left(\boldsymbol{\mu}^k_{\tilde{\mathcal{U}}_i} - \boldsymbol{\mu}^k_{\tilde{\mathcal{U}}}\right)^2}{\left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}_i}\right)^2 + \left(\boldsymbol{\sigma}^k_{\tilde{\mathcal{U}}}\right)^2}\right] = F1\left(\tilde{\mathcal{U}}_i, \tilde{\mathcal{U}}\right).$$

The smaller the Fisher discriminant ratio is, the less discriminative the two domains are. The less discriminative the two domains are, the smaller $d_{\mathcal{H}}(\cdot, \cdot)$ is. Thus, finally, we have

$$d_{\mathcal{H}}\left(\tilde{\mathcal{U}}_i^g, \tilde{\mathcal{U}}^g\right) \le d_{\mathcal{H}}\left(\tilde{\mathcal{U}}_i, \tilde{\mathcal{U}}\right).$$

□

B.3 Derivations of Corollary 2

When we focus on the global-to-local knowledge transfer on the *client side*, in the FL scenario, we rewrite Theorem 1 as

Theorem 3. *Consider a virtual global data domain $\mathcal{D}$ and a local data domain $\mathcal{D}_i$. Let $\mathcal{D} = \langle \mathcal{U}, c^* \rangle$ and $\mathcal{D}_i = \langle \mathcal{U}_i, c^* \rangle$, where $\mathcal{U} \subseteq \mathcal{X}$ and $\mathcal{U}_i \subseteq \mathcal{X}$. Given a feature extraction function $\mathcal{F}: \mathcal{X} \mapsto \mathcal{Z}$ that shared between $\mathcal{D}$ and $\mathcal{D}_i$, a random labeled sample of size m generated by applying $\mathcal{F}$ to a random sample from $\mathcal{U}$ labeled according to c^*, then for every $h \in \mathcal{H}$, with probability at least $1 - \delta$:*

$$\mathcal{L}_{\mathcal{D}_i}(h) \le \mathcal{L}_{\hat{\mathcal{D}}}(h) + \sqrt{\frac{4}{m}\left(d \log \frac{2em}{d} + \log \frac{4}{\delta}\right)} + d_{\mathcal{H}}\left(\tilde{\mathcal{U}}, \tilde{\mathcal{U}}_i\right) + \lambda_i,$$

where $\tilde{\mathcal{U}}_i$ and $\tilde{\mathcal{U}}$ are the induced distributions of $\mathcal{U}_i$ and $\mathcal{U}$ under $\mathcal{F}$, respectively. $\tilde{\mathcal{U}}_i \subseteq \mathcal{Z}$ and $\tilde{\mathcal{U}} \subseteq \mathcal{Z}$. $\lambda_i := \min_h \mathcal{L}_{\mathcal{D}}(h) + \mathcal{L}_{\mathcal{D}_i}(h)$ denotes an oracle performance.

Corollary 2. *Let $\mathcal{D}_i$, $\mathcal{D}$, $\mathcal{F}^g$, and λ_i defined as in Corollary 1. Given a translation transformation function* `PRBM` *$: \mathcal{Z} \mapsto \mathcal{Z}$ that shared between $\mathcal{D}_i$ and virtual $\mathcal{D}$, a random labeled sample of size m generated by applying $\mathcal{F}'$ to a random sample from $\mathcal{U}_i$ labeled according to c^*, $\mathcal{F}' =$* `PRBM` *$\circ \mathcal{F}^g : \mathcal{X} \mapsto \mathcal{Z}$, then for every $h' \in \mathcal{H}$, with probability at least $1 - \delta$:*

$$\mathcal{L}_{\mathcal{D}_i}(h') \le \mathcal{L}_{\hat{\mathcal{D}}}(h') + \sqrt{\frac{4}{m}\left(d \log \frac{2em}{d} + \log \frac{4}{\delta}\right)} + d_{\mathcal{H}}\left(\tilde{\mathcal{U}}', \tilde{\mathcal{U}}_i'\right) + \lambda_i,$$

where $d_{\mathcal{H}}\left(\tilde{\mathcal{U}}', \tilde{\mathcal{U}}_i'\right) = d_{\mathcal{H}}\left(\tilde{\mathcal{U}}^g, \tilde{\mathcal{U}}_i^g\right) \le d_{\mathcal{H}}\left(\tilde{\mathcal{U}}, \tilde{\mathcal{U}}_i\right) = d_{\mathcal{H}}\left(\tilde{\mathcal{U}}_i, \tilde{\mathcal{U}}\right)$. $\tilde{\mathcal{U}}'$ and $\tilde{\mathcal{U}}_i'$ are the induced distributions of $\mathcal{U}$ and $\mathcal{U}_i$ under $\mathcal{F}'$, respectively.

Proof. `PRBM` is a translation transformation with parameters $\bar{z}_i^p$, *s.t.* $\forall\, \boldsymbol{x}_i \in \mathcal{U}_i$, $\boldsymbol{z}_i = \boldsymbol{z}_i^g + \bar{\boldsymbol{z}}_i^p$, where $\boldsymbol{z}_i = \mathcal{F}'(\boldsymbol{x}_i) \in \tilde{\mathcal{U}}_i'$ and $\boldsymbol{z}_i^g = \mathcal{F}^g(\boldsymbol{x}_i) \in \tilde{\mathcal{U}}_i^g$. In other words, $\forall\, \boldsymbol{z}_i^g \in \tilde{\mathcal{U}}_i^g, \exists!\, \boldsymbol{z}_i \in \tilde{\mathcal{U}}_i'$. Therefore, we have $\Pr_{\tilde{\mathcal{U}}_i^g}[\{\boldsymbol{z} \in \mathcal{Z}\}] = \Pr_{\tilde{\mathcal{U}}_i'}[\{\boldsymbol{z} \in \mathcal{Z}\}]$ and the same applies to the pair of $\tilde{\mathcal{U}}^g$ and $\tilde{\mathcal{U}}'$, *i.e.*,

$\Pr_{\tilde{\mathcal{U}}^g}\left[\{\boldsymbol{z} \in \mathcal{Z}\}\right] = \Pr_{\tilde{\mathcal{U}}'}\left[\{\boldsymbol{z} \in \mathcal{Z}\}\right]$. Then the subtraction of the probability on each side is also equal, *i.e.*,

$$\Pr_{\tilde{\mathcal{U}}_i^g}\left[\{\boldsymbol{z} \in \mathcal{Z}\}\right] - \Pr_{\tilde{\mathcal{U}}^g}\left[\{\boldsymbol{z} \in \mathcal{Z}\}\right] = \Pr_{\tilde{\mathcal{U}}_i'}\left[\{\boldsymbol{z} \in \mathcal{Z}\}\right] - \Pr_{\tilde{\mathcal{U}}'}\left[\{\boldsymbol{z} \in \mathcal{Z}\}\right].$$

$\forall\, h' \in \mathcal{H}, h^g = h' \circ \texttt{PRBM} \in \mathcal{H}$, so $\forall\, \boldsymbol{z}^a \in \mathcal{Z}$ if $h^g\left(\boldsymbol{z}^a\right) = 1$, then $h'\left(\boldsymbol{z}^b\right) = 1$, where $\boldsymbol{z}^b = \boldsymbol{z}^a + \bar{\boldsymbol{z}}_i^p$. Therefore, we have

$$\Pr_{\tilde{\mathcal{U}}_i^g}\left[\mathcal{Z}_{h^g}\right] - \Pr_{\tilde{\mathcal{U}}^g}\left[\mathcal{Z}_{h^g}\right] = \Pr_{\tilde{\mathcal{U}}_i'}\left[\mathcal{Z}_{h'}\right] - \Pr_{\tilde{\mathcal{U}}'}\left[\mathcal{Z}_{h'}\right],$$

where $\mathcal{Z}_{h^g} = \{\boldsymbol{z} \in \mathcal{Z} : h^g(\boldsymbol{z}) = 1\}, h^g \in \mathcal{H}$ and $\mathcal{Z}_{h'} = \{\boldsymbol{z} \in \mathcal{Z} : h'(\boldsymbol{z}) = 1\}, h' \in \mathcal{H}$. According to Definition 1, we have

$$\begin{aligned} d_{\mathcal{H}}\left(\tilde{\mathcal{U}}', \tilde{\mathcal{U}}_i'\right) &= 2 \sup_{h' \in \mathcal{H}} \left|\Pr_{\tilde{\mathcal{U}}_i'}\left[\mathcal{Z}_{h'}\right] - \Pr_{\tilde{\mathcal{U}}'}\left[\mathcal{Z}_{h'}\right]\right| \\ &= 2 \sup_{h^g \in \mathcal{H}} \left|\Pr_{\tilde{\mathcal{U}}_i^g}\left[\mathcal{Z}_{h^g}\right] - \Pr_{\tilde{\mathcal{U}}^g}\left[\mathcal{Z}_{h^g}\right]\right| \\ &= d_{\mathcal{H}}\left(\tilde{\mathcal{U}}^g, \tilde{\mathcal{U}}_i^g\right) \\ &\leq d_{\mathcal{H}}\left(\tilde{\mathcal{U}}, \tilde{\mathcal{U}}_i\right). \end{aligned}$$

□

C Detailed Settings

C.1 Implementation Details

We create the datasets for each client using six public datasets: Fashion-MNIST (FMNIST)[2], Cifar100[3], Tiny-ImageNet[4] (100K images with 200 labels) and AG News[5] (a news classification dataset with four labels, more than 30K samples per label). The MDL is calculated through the public code[6]. We run all experiments on a machine with two Intel Xeon Gold 6140 CPUs (36 cores), 128G memory, eight NVIDIA 2080 Ti GPUs, and CentOS 7.8.

C.2 Hyperparameters of `DBE`

For hyperparameter tuning, we use grid search to find optimal hyperparameters, including κ and μ. Specifically, grid search is performed in the following search space:

- κ: 0, 0.001, 0.01, 0.1, 1, 5, 10, 20, 50, 100, 200, 500
- μ: 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0

In this paper, we set $\kappa = 50, \mu = 1.0$ for the 4-layer CNN, $\kappa = 1, \mu = 0.1$ for the ResNet-18, and $\kappa = 0.1, \mu = 1.0$ for the fastText. We only set different values for the hyperparameters κ and μ on different model architectures but use identical settings for one architecture on all datasets. Different models exhibit diverse capabilities in both feature extraction and classification. Given that our proposed `DBE` operates by integrating itself into a specific model, it is crucial to tune the parameters κ and μ to adapt to the feature extraction and classification abilities of different models.

As for the *criteria for hyperparameter tuning*, κ and μ require different tunning methods according to their functions. Specifically, μ is a momentum introduced along with the widely-used moving average technology in approximating statistics, so for the model architectures that originally contain statistics collection operations (*e.g.*, the batch normalization layers in ResNet-18) one can set a relatively small value by tuning μ from 0 to 1 with a reasonable step size. For other model architectures, one can set a relatively large value for μ by tuning it from 1 to 0. The parameter κ is utilized to regulate the magnitude of the MSE loss in `MR`. However, different architectures generate feature representations with varying magnitudes, leading to differences in the magnitude of the MSE loss. Thus, we tune κ by aligning the magnitude of the MSE loss with the other loss term.

---

[2] `https://pytorch.org/vision/stable/datasets.html#fmnist`

[3] `https://pytorch.org/vision/stable/datasets.html#cifar`

[4] `http://cs231n.stanford.edu/tiny-imagenet-200.zip`

[5] `https://pytorch.org/text/stable/datasets.html#ag-news`

[6] `https://github.com/willwhitney/reprieve`

D Additional Experiments

D.1 Convergence

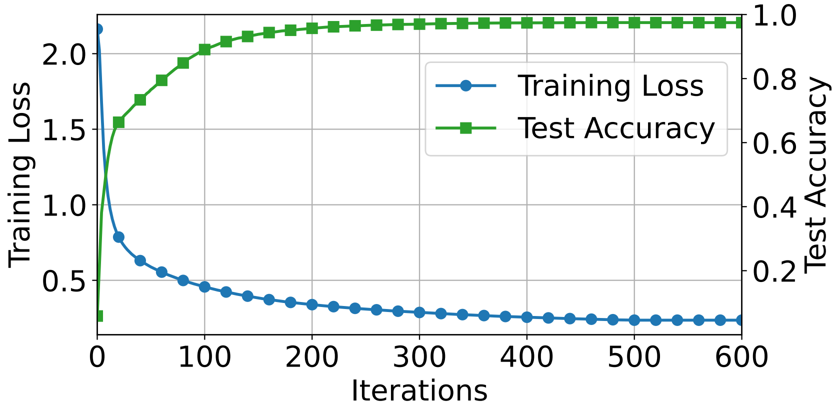


Figure 1: The training loss and test accuracy curve of FedAvg+DBE on FMNIST dataset using the 4-layer CNN in the practical setting.

Recall that our objective is

$$\min_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_N} \mathbb{E}_{i\in[N]}[\mathcal{L}_{\mathcal{D}_i}(\boldsymbol{\theta}_i)], \tag{2}$$

and its empirical version is $\min_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_N} \sum_{i=1}^{N} \frac{n_i}{n} \mathcal{L}_{\hat{\mathcal{D}}_i}(\boldsymbol{\theta}_i)$. Here, we visualize the value of $\sum_{i=1}^{N} \frac{n_i}{n} \mathcal{L}_{\hat{\mathcal{D}}_i}(\boldsymbol{\theta}_i)$ and the corresponding test accuracy during the FL process. Figure 1 shows the convergence of FedAvg+DBE and its stable training procedure. Besides, we also report the total iterations required for convergence on Tiny-ImageNet using ResNet-18 in Table 2. Based on the findings from Table 2, we observe that the utilization of DBE can yield a substantial reduction from 230 to 107 (more than 50%) in the total number of communication iterations needed for convergence, as compared to the original requirements of FedAvg.

D.2 Model-Splitting in ResNet-18

In the main body, we have shown that DBE improves the per-layer MDL and accuracy of FedAvg no matter how we split the 4-layer CNN. In Table 1, we report the per-layer MDL and accuracy when we consider model splitting in ResNet-18, a model deeper than the 4-layer CNN. No matter at which layer we split ResNet-18 to form a feature extractor and a classifier, DBE can also reduce MDL and improve accuracy, showing its general applicability.

Table 1: The MDL (bits, ↓) of layer-wise representations, test accuracy (%, ↑), and the number of trainable parameters (↓) in PRBM when adding DBE to FedAvg on Tiny-ImageNet using ResNet-18 in the practical setting. The "B", "CONV", "POOL", and "FC" means the "block", "convolution block", "average pool layer", and "fully connected layer" in ResNet-18 [27], respectively.

Metrics	**MDL**							**Accuracy**	**Param.**
	CONV→B1	B1→B2	B2→B3	B3→B4	B4→POOL	POOL→FC	Logits		
Original (FedAvg)	4557	4198	3598	3501	3445	3560	3679	19.45	0
CONV→DBE →B1	4332	4050	3528	3407	3292	3347	3493	19.96	16384
B1→DBE →B2	4527	4072	3568	3456	3361	3451	3560	19.50	16384
B2→DBE →B3	4442	4091	3575	3474	3326	3411	3520	19.55	8192
B3→DBE →B4	4447	4073	3511	3414	3259	3346	3467	20.72	4096
B4→DBE →POOL	4424	4030	3391	3304	3284	3511	3612	39.99	2048
POOL→DBE →FC	4432	4035	3359	3298	3209	3454	3594	42.98	512

D.3 Distinguishable Representations

As our primary goal is to demonstrate the elimination of representation bias rather than improving discrimination in Figure 3 (main body), we present the t-SNE visualization for our largest dataset in experiments, Tiny-ImageNet (200 labels). Given that the 200 labels are distributed around the chromatic circle, adjacent labels are assigned similar colors, resulting in Figure 3 (main body) being indistinguishable by the label. Using a dataset AG News with only four labels for t-SNE visualization can clearly show that the representations extracted by the global feature extractor are distinguishable in Figure 2.

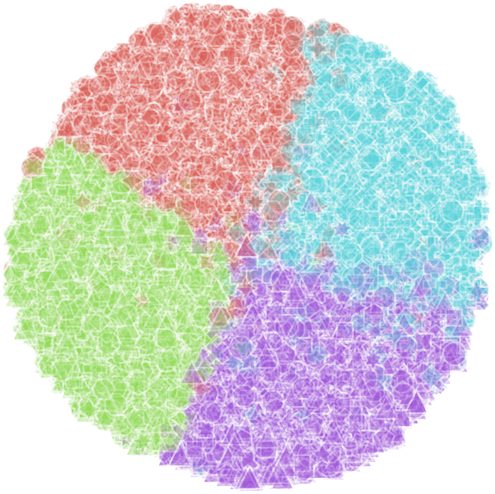


Figure 2: t-SNE visualization for the representations extracted by the global feature extractor on AG News (four labels) in FedAvg+`DBE`. We use *color* and *shape* to distinguish *labels* and *clients*, respectively.

D.4 A Practical Scenario with New Participants

To simulate a practical scenario with new clients joining for future FL, we perform method-specific local training for 10 epochs on new participants for warming up after their local models are initialized by the learned global model (or client models in FedFomo). Since FedAvg, Per-FedAvg, and FedBABU do not generate personalized models during the FL process, we fine-tune the entire global model on new clients for them to obtain test accuracy. Specifically, using Cifar100 and 4-layer CNN, we conduct FL on 80 old clients ($\rho = 0.5$ or $\rho = 0.1$) and evaluate accuracy on 20 new joining clients after warming up. We utilize the data distribution depicted in Figure 6. According to Table 2, FedAvg shows excellent generalization ability with fine-tuning. However, `DBE` can still improve FedAvg by up to **+6.68** with more stable performance for different ρ.

D.5 Large Local Epochs

We also conduct experiments with more local epochs in each iteration on FMNIST using the 4-layer CNN, as shown in Table 2. All the pFL methods perform similarly with the results for one local epoch, except for Per-FedAvg, which degenerates around 1.18 in accuracy (%).

D.6 Real-World Application

We also evaluate the performance of our `DBE` in a real-world application. Specifically, we apply `DBE` to the Internet-of-Things (IoT) scenario on a popular Human Activity Recognition (HAR) dataset [2] with the HAR-CNN [65] model. HAR contains the sensor signal data collected from 30 users who perform six activities (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING) wearing a smartphone on the waist. We show the results in Table 3, where FedAvg+`DBE` still achieves superior performance.

Table 2: The total iterations for convergence and the averaged test accuracy (%, ↑) of pFL methods.

Items	**Iterations**	**New Participants**		**Local Epochs**		
		$\rho = 0.5$	$\rho = 0.1$	1	5	10
Per-FedAvg [20]	34	48.66	48.36	95.10	93.92	93.91
pFedMe [58]	113	41.20	38.39	97.25	97.44	97.32
Ditto [42]	27	36.57	45.06	97.47	97.67	97.64
FedPer [3]	43	39.86	42.39	97.44	97.50	97.54
FedRep [18]	115	38.75	35.09	97.56	97.55	97.55
FedRoD [13]	50	50.10	51.73	97.52	97.49	97.35
FedBABU [55]	513	48.60	42.29	97.46	97.57	97.65
APFL [19]	57	38.19	45.16	97.25	97.31	97.34
FedFomo [68]	71	27.50	27.47	97.21	97.17	97.22
APPLE [46]	45	—	—	97.06	97.07	97.01
FedAvg	230	52.52	49.44	85.85	85.96	85.53
FedAvg+`DBE`	107	**57.62**	**56.12**	**97.69**	**97.75**	**97.78**

Table 3: The test accuracy (%) on the HAR dataset.

Methods	Accuracy
FedAvg	87.20±0.27
SCAFFOLD	91.34±0.43
FedProx	88.34±0.24
MOON	89.86±0.18
FedGen	90.82±0.21
Per-FedAvg	77.12±0.17
pFedMe	91.57±0.12
Ditto	91.53±0.09
FedPer	75.58±0.13
FedRep	80.44±0.42
FedRoD	89.91±0.23
FedBABU	87.12±0.31
APFL	92.18±0.51
FedFomo	63.39±0.48
APPLE	86.46±0.35
FedAvg+`DBE`	**94.53±0.26**

E Broader Impacts

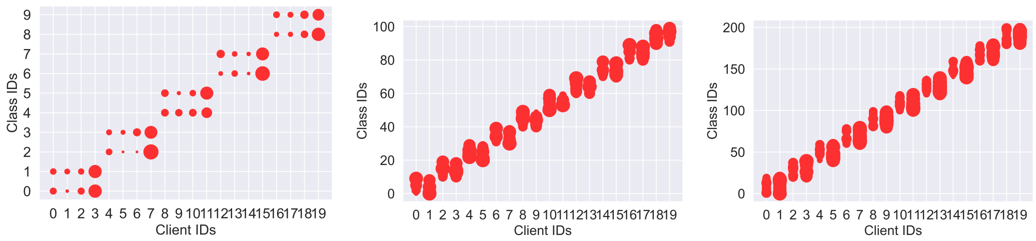
The representation bias and representation degeneration naturally exist in FL under statistically heterogeneous scenarios, which are derived from the inherently separated local data domains on individual clients. In the main body, we show the general applicability of our proposed `DBE` to representative FL methods. More than that, `DBE` can also be applied to other practical fields, such as the Internet of Things (IoT) [24, 29, 53] and digital health [14, 15]. Furthermore, introducing the view of knowledge transfer into FL sheds light on this field.

F Limitations

Although FL comes along for privacy-preserving and collaborative learning, it still suffers from privacy leakage issues with malicious clients [11, 69] or under attacks [21, 47]. We design `DBE` based on FL to improve generalization and personalization abilities, and we only modify the local training procedure without affecting the downloading, uploading, and aggregation processes. Thus, the `DBE`-equipped FL methods still suffer from the originally existing privacy issues like the original version of these FL methods when attacks happen. It requires future work to devise specific methods for privacy-preserving enhancement.

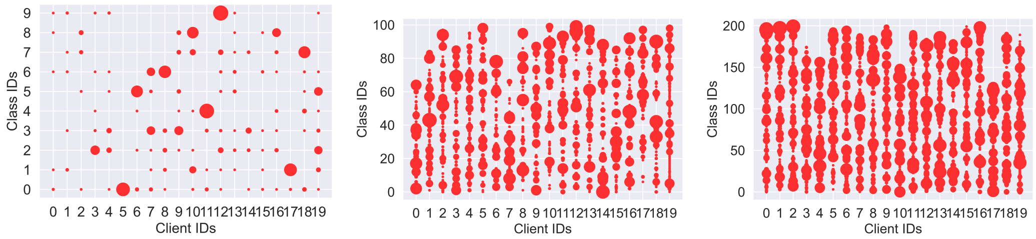
G Data Distribution Visualization

We illustrate the data distributions (including training and test data) in our experiments here.



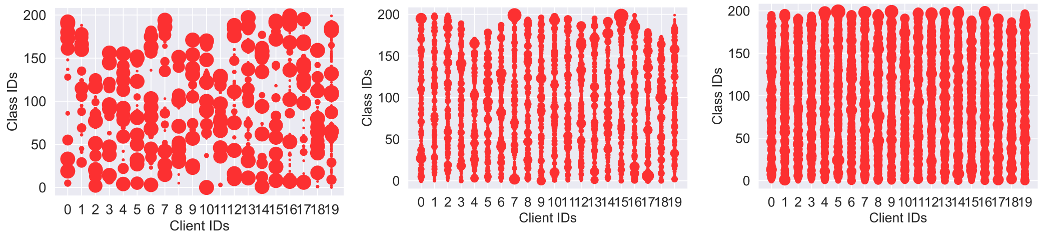
(a) FMNIST (b) Cifar100 (c) Tiny-ImageNet

Figure 3: The data distributions of all clients on FMNIST, Cifar100, and Tiny-ImageNet, respectively, in the pathological settings. The size of a circle represents the number of samples.



(a) FMNIST (b) Cifar100 (c) Tiny-ImageNet

Figure 4: The data distributions of all clients on FMNIST, Cifar100, and Tiny-ImageNet, respectively, in the practical settings ($\beta = 0.1$). The size of a circle represents the number of samples.



(a) $\beta = 0.01$ (b) $\beta = 0.5$ (c) $\beta = 5$

Figure 5: The data distribution on all clients on Tiny-ImageNet in three additional practical settings. The size of a circle represents the number of samples. The degree of heterogeneity decreases as β in $Dir(\beta)$ increases.

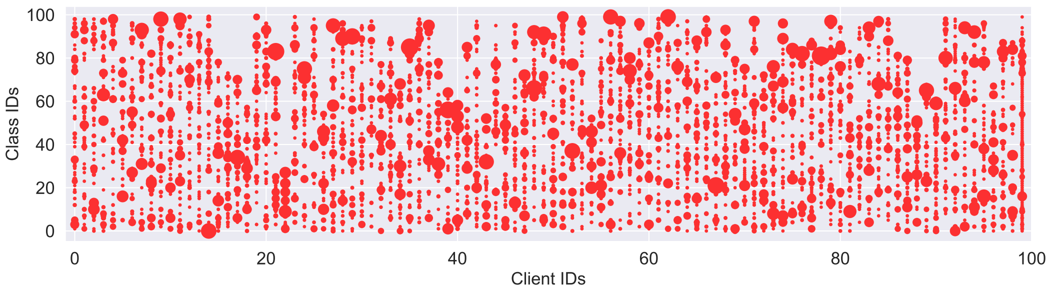


Figure 6: The data distributions of all clients on Cifar100 in the practical setting ($\beta = 0.1$) with 100 clients, respectively. The size of a circle represents the number of samples.

References

[1] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated Learning Based on Dynamic Regularization. In *International Conference on Learning Representations (ICLR)*, 2021.

[2] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *Ambient Assisted Living and Home Care: 4th International Workshop, IWAAL 2012, Vitoria-Gasteiz, Spain, December 3-5, 2012. Proceedings 4*, pages 216–223. Springer, 2012.

[3] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated Learning with Personalization Layers. *arXiv preprint arXiv:1912.00818*, 2019.

[4] Suresh Balakrishnama and Aravind Ganapathiraju. Linear discriminant analysis-a brief tutorial. *Institute for Signal and information Processing*, 18(1998):1–8, 1998.

[5] Shai Ben-David, Nadav Eiron, and Philip M Long. On the difficulty of approximately maximizing agreements. *Journal of Computer and System Sciences*, 66(3):496–514, 2003.

[6] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of Representations for Domain Adaptation. In *International Conference on Advances in Neural Information Processing Systems (NeurIPS)*, 2006.

[7] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79:151–175, 2010.

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

[9] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447, 2007.

[10] José-Ramón Cano. Analysis of data complexity measures for classification. *Expert systems with applications*, 40(12):4820–4831, 2013.

[11] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Provably secure federated learning against malicious clients. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[12] Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated Meta-Learning With Fast Convergence and Efficient Communication. *arXiv preprint arXiv:1802.07876*, 2018.

[13] Hong-You Chen and Wei-Lun Chao. On Bridging Generic and Personalized Federated Learning for Image Classification. In *International Conference on Learning Representations (ICLR)*, 2021.

[14] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

[15] Yiqiang Chen, Wang Lu, Xin Qin, Jindong Wang, and Xing Xie. MetaFed: Federated Learning Among Federations With Cyclic Knowledge Distillation for Personalized Healthcare. *arXiv preprint arXiv:2206.08516*, 2022.

[16] Anda Cheng, Peisong Wang, Xi Sheryl Zhang, and Jian Cheng. Differentially Private Federated Learning with Local Regularization and Sparsification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[17] Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems*, 31, 2018.

[18] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting Shared Representations for Personalized Federated Learning. In *International Conference on Machine Learning (ICML)*, 2021.

[19] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive Personalized Federated Learning. *arXiv preprint arXiv:2003.13461*, 2020.

[20] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach. In *International Conference on Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[21] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *USENIX Security*, 2020.

[22] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *International Conference on Machine Learning (ICML)*, 2017.

[23] Liang Gao, Huazhu Fu, Li Li, Yingwen Chen, Ming Xu, and Cheng-Zhong Xu. FedDC: Federated Learning With Non-IID Data Via Local Drift Decoupling and Correction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[24] Bimal Ghimire and Danda B Rawat. Recent Advances on Federated Learning for Cybersecurity and Cybersecurity for Federated Learning for Internet of Things. *IEEE Internet of Things Journal*, 09(11): 8229 – 8249, 2022.

[25] Xuan Gong, Abhishek Sharma, Srikrishna Karanam, Ziyan Wu, Terrence Chen, David Doermann, and Arun Innanje. Preserving Privacy in Federated Learning With Ensemble Cross-Domain Knowledge Distillation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

[26] Meng Hao, Hongwei Li, Xizhao Luo, Guowen Xu, Haomiao Yang, and Sen Liu. Efficient and Privacy-Enhanced Federated Learning for Industrial Artificial Intelligence. *IEEE Transactions on Industrial Informatics*, 16(10):6532–6542, 2019.

[27] 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.

[28] Tin Kam Ho and Mitra Basu. Complexity measures of supervised classification problems. *IEEE transactions on pattern analysis and machine intelligence*, 24(3):289–300, 2002.

[29] Rui Hu, Yuanxiong Guo, Hongning Li, Qingqi Pei, and Yanmin Gong. Personalized federated learning with differential privacy. *IEEE Internet of Things Journal*, 7(10):9530–9539, 2020.

[30] Tiansheng Huang, Weiwei Lin, Wentai Wu, Ligang He, Keqin Li, and Albert Y Zomaya. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Transactions on Parallel and Distributed Systems*, 32(7):1552–1564, 2020.

[31] Wenke Huang, Mang Ye, and Bo Du. Learn From Others and Be Yourself in Heterogeneous Federated Learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[32] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[33] Meirui Jiang, Zirui Wang, and Qi Dou. Harmofl: Harmonizing Local and Global Drifts in Federated Learning on Heterogeneous Medical Images. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

[34] 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.

[35] 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.

[36] Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *VLDB*, pages 180–191, 2004.

[37] Jinkyu Kim, Geeho Kim, and Bohyung Han. Multi-Level Branched Regularization for Federated Learning. In *International Conference on Machine Learning (ICML)*, 2022.

[38] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated Learning: Strategies for Improving Communication Efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[39] Max Kuhn, Kjell Johnson, Max Kuhn, and Kjell Johnson. Discriminant analysis and other linear classification models. *Applied predictive modeling*, pages 275–328, 2013.

[40] Qinbin Li, Bingsheng He, and Dawn Song. Model-Contrastive Federated Learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[41] 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.

[42] 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.

[43] Zengpeng Li, Vishal Sharma, and Saraju P Mohanty. Preserving Data Privacy via Federated Learning: Challenges and Solutions. *IEEE Consumer Electronics Magazine*, 9(3):8–16, 2020.

[44] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble Distillation for Robust Model Fusion in Federated Learning. In *International Conference on Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[45] Yang Liu, Yan Kang, Chaoping Xing, Tianjian Chen, and Qiang Yang. A secure federated transfer learning framework. *IEEE Intelligent Systems*, 35(4):70–82, 2020.

[46] Jun Luo and Shandong Wu. Adapt to Adaptation: Learning Personalization for Cross-Silo Federated Learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2022.

[47] Xinjian Luo, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. Feature inference attack on model predictions in vertical federated learning. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 181–192. IEEE, 2021.

[48] WANG Luping, WANG Wei, and LI Bo. Cmfl: Mitigating communication overhead for federated learning. In *2019 IEEE 39th international conference on distributed computing systems (ICDCS)*, pages 954–964. IEEE, 2019.

[49] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three Approaches for Personalization with Applications to Federated Learning. *arXiv preprint arXiv:2002.10619*, 2020.

[50] 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.

[51] Sebastian Mika, Gunnar Ratsch, Jason Weston, Bernhard Scholkopf, and Klaus-Robert Mullers. Fisher discriminant analysis with kernels. In *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop (cat. no. 98th8468)*, pages 41–48. Ieee, 1999.

[52] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A Survey on Security and Privacy of Federated Learning. *Future Generation Computer Systems*, 115:619–640, 2021.

[53] Dinh C Nguyen, Ming Ding, Pubudu N Pathirana, Aruna Seneviratne, Jun Li, and H Vincent Poor. Federated Learning for Internet of Things: A Comprehensive Survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.

[54] Yifan Niu and Weihong Deng. Federated Learning for Face Recognition With Gradient Correction. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

[55] Jaehoon Oh, SangMook Kim, and Se-Young Yun. FedBABU: Toward Enhanced Representation for Federated Image Classification. In *International Conference on Learning Representations (ICLR)*, 2022.

[56] Tao Qi, Fangzhao Wu, Chuhan Wu, Lingjuan Lyu, Tong Xu, Hao Liao, Zhongliang Yang, Yongfeng Huang, and Xing Xie. Fairvfl: A fair vertical federated learning framework with contrastive adversarial learning. *Advances in Neural Information Processing Systems*, 35:7852–7865, 2022.

[57] Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced Meta-Softmax for Long-Tailed Visual Recognition. In *International Conference on Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[58] Canh T Dinh, Nguyen Tran, and Tuan Dung Nguyen. Personalized Federated Learning with Moreau Envelopes. In *International Conference on Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[59] Shauhin A Talesh. Data breach, privacy, and cyber insurance: How insurance companies act as "compliance managers" for businesses. *Law & Social Inquiry*, 43(2):417–440, 2018.

[60] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards Personalized Federated Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. Early Access.

[61] Welderufael B Tesfay, Peter Hofmann, Toru Nakamura, Shinsaku Kiyomoto, and Jetzabel Serna. I read but don't agree: Privacy policy benchmarking using machine learning and the eu gdpr. In *Companion Proceedings of the The Web Conference 2018*, pages 163–166, 2018.

[62] Suge Wang, Deyu Li, Xiaolei Song, Yingjie Wei, and Hongxia Li. A feature selection method based on improved fisher's discriminant ratio for text sentiment classification. *Expert Systems with Applications*, 38 (7):8696–8702, 2011.

[63] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated Machine Learning: Concept and Applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2):1–19, 2019.

[64] Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. A fairness-aware incentive scheme for federated learning. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 393–399, 2020.

[65] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *6th international conference on mobile computing, applications and services*, pages 197–205. IEEE, 2014.

[66] 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.

[67] 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.

[68] Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M Alvarez. Personalized Federated Learning with First Order Model Optimization. In *International Conference on Learning Representations (ICLR)*, 2020.

[69] Zaixi Zhang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Fldetector: Defending federated learning against model poisoning attacks via detecting malicious clients. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining*, 2022.

[70] Wennan Zhu, Peter Kairouz, Brendan McMahan, Haicheng Sun, and Wei Li. Federated Heavy Hitters Discovery with Differential Privacy. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

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