# Data-Free Knowledge Distillation for Heterogeneous Federated Learning

**Zhuangdi Zhu** [1] **Junyuan Hong** [1] **Jiayu Zhou** [1]

## Abstract

Federated Learning (FL) is a decentralized machine-learning paradigm, in which a global server iteratively aggregates the model parameters of local users without accessing their data. User *heterogeneity* has imposed significant challenges to FL, which can incur drifted global models that are slow to converge. *Knowledge Distillation* has recently emerged to tackle this issue, by refining the server model using aggregated knowledge from heterogeneous users, other than directly aggregating their model parameters. This approach, however, depends on a proxy dataset, making it impractical unless such prerequisite is satisfied. Moreover, the ensemble knowledge is not fully utilized to guide local model learning, which may in turn affect the quality of the aggregated model. Inspired by prior art, we propose a *data-free knowledge distillation* approach to address heterogeneous FL, where the server learns a lightweight generator to ensemble user information in a data-free manner, which is then broadcasted to users, regulating local training using the learned knowledge as an inductive bias. Empirical studies powered by theoretical implications show that, our approach facilitates FL with better generalization performance using fewer communication rounds, compared with the state-of-the-art

## 1. Introduction

Federated Learning (FL) is an effective machine learning approach that enables the decentralization of computing and data resources. Classical FL, represented by FEDAVG (McMahan et al., 2017), obtains an aggregated model by iteratively averaging the parameters of distributed local user models, therefore omits the need of accessing their data. Serving as a communication-efficient and privacy-preserving learning scheme, FL has shown its potential to facilitate real-world applications, including healthcare (Sheller et al., 2020) and natural language processing (Hard et al., 2018; Ammad-Ud-Din et al., 2019), to name just a few.

Along with its promising prospect, FL faces practical challenges from data ***heterogeneity*** (Li et al., 2020b), in that user data from real-world is usually *non-iid* distributed, which inherently induces deflected local optimum (Karimireddy et al., 2020). Moreover, the permutation-invariant property of deep neural networks has further increased the heterogeneity among user models (Yurochkin et al., 2019; Wang et al., 2020b). Thus, performing element-wise averaging of local models, as adopted by most existing FL approaches, may not induce an ideal global model (Li et al., 2020c;b).

A variety of efforts have been made to tackle user heterogeneity, mainly from two complementary perspectives: one focuses on stabilizing local training, by regulating the deviation of local models from a global model over the *parameter space* (Li et al., 2020b; Dinh et al., 2020; Karimireddy et al., 2020). This approach may not fully leverage the underlying knowledge across user models, whose diversity suggests informative structural differences of their local data and thus deserves more investigation. Another aims to improve the efficacy of model aggregation (Yurochkin et al., 2019; Chen & Chao, 2021), among which *knowledge distillation* has emerged as an effective solution (Lin et al., 2020; Li & Wang, 2019). Provided with an unlabeled dataset as the proxy, knowledge distillation alleviates the model drift issue induced by heterogeneity, by enriching the global model with the ensemble knowledge from local models, which is shown to be more effective than simple parameter-averaging. However, the prerequisite of a *proxy data* can leave such an approach infeasible for many applications, where a carefully engineered dataset may not always be available on the server. Moreover, by only refining the global model, the inherent heterogeneity among user models is not fully addressed, which may in turn affect the quality of the knowledge ensemble, especially if they are biased due to limited local data (Khoussainov et al., 2005), which is a typical case for FL.

Observing the challenge in the presence of user heterogeneity and the limitations of prior art, in this work, we propose a

[1]Department of Computer Science and Engineering, Michigan State University, Michigan, USA. Correspondence to: Zhuangdi Zhu <zhuzhuan@msu.edu>, Jiayu Zhou <jiayuz@msu.edu>.

*Proceedings of the $38^{th}$ International Conference on Machine Learning*, PMLR 139, 2021. 

***data-free*** knowledge distillation approach for FL, dubbed as FEDGEN (*Fe*derated *D*istillation via *Gen*erative Learning). Specifically, FEDGEN learns a generative model derived solely from the prediction rules of user models, which, given a target label, can yield feature representations that are consistent with the ensemble of user predictions. This generator is later broadcasted to users, escorting their model training with augmented samples over the latent space, which embodies the distilled knowledge from other peer users. Given a latent space with a dimension much smaller than the input space, the generator learned by FEDGEN can be lightweight, introducing minimal overhead to the current FL framework.

The proposed FEDGEN enjoys multifold benefits: i) It extracts the knowledge out of users which was otherwise mitigated after model averaging, without depending on any external data. ii) Contrary to certain prior work that only refines the global model, our approach directly regulates local model updating using the extracted knowledge. We show that such knowledge imposes an inductive bias to local models, leading to better generalization performance under *non-iid* data distributions. iii) Furthermore, the proposed approach is ready to address more challenging FL scenarios, where sharing *entire* model parameters is impractical due to privacy or communication constraints, since the proposed approach only requires the prediction layer of local models for knowledge extraction.

Extensive empirical studies echoed by theoretical elaborations show that, our proposed approach yields a global model with better generalization performance using fewer communication rounds, compared with the state-of-the-art.

## 2. Notations and Preliminaries

Without ambiguity, in this work, we discuss a typical FL setting for *supervised learning*, i.e., the general problem of multi-class classification. Let $\mathcal{X} \subset \mathbb{R}^p$ be an instance space, $\mathcal{Z} \subset \mathbb{R}^d$ be a *latent* feature space with $d < p$, and $\mathcal{Y} \subset \mathbb{R}$ be an output space. $\mathcal{T}$ denotes a *domain* which consists of a data distribution $\mathcal{D}$ over $\mathcal{X}$ and a ground-truth *labeling* function $c^* : \mathcal{X} \to \mathcal{Y}$, *i.e.* $\mathcal{T} := \langle \mathcal{D}, c^* \rangle$. Note that we will use the term *domain* and *task* equivalently. A model parameterized by $\boldsymbol{\theta} := [\boldsymbol{\theta}^f; \boldsymbol{\theta}^p]$ consists of two components: a feature extractor $f : \mathcal{X} \to \mathcal{Z}$ parametrized by $\boldsymbol{\theta}^f$, and a predictor $h : \mathcal{Z} \to \triangle^{\mathcal{Y}}$ parameterized by $\boldsymbol{\theta}^p$, where $\triangle^{\mathcal{Y}}$ is the simplex over $\mathcal{Y}$. Given a non-negative, convex loss function $l : \triangle^{\mathcal{Y}} \times \mathcal{Y} \to \mathbb{R}$, the ***risk*** of a model parameterized by $\boldsymbol{\theta}$ on domain $\mathcal{T}$ is defined as $\mathcal{L}_{\mathcal{T}}(\boldsymbol{\theta}) := \mathbb{E}_{x \sim \mathcal{D}} \left[ l \left( h(f(x; \boldsymbol{\theta}^f); \boldsymbol{\theta}^p), c^*(x) \right) \right]$.

**Federated Learning** aims to learn a global model parameterized by $\boldsymbol{\theta}$ that minimizes its risk on each of the user tasks $\mathcal{T}_k$ (McMahan et al., 2017):

$$\min_{\boldsymbol{\theta}} \ \mathbb{E}_{\mathcal{T}_k \in \boldsymbol{\mathcal{T}}} \left[ \mathcal{L}_k(\boldsymbol{\theta}) \right], \tag{1}$$

where $\boldsymbol{\mathcal{T}} = \{\mathcal{T}_k\}_{k=1}^K$ is the collection of user tasks. We consider all tasks sharing the same labeling rules $c^*$ and loss function $l$, *i.e.*, $\mathcal{T}_k = \langle \mathcal{D}_k, c^* \rangle$. In practice, Equation 1 is empirically optimized by $\min_{\boldsymbol{\theta}} \frac{1}{K} \sum_{k=1}^K \hat{\mathcal{L}}_k(\boldsymbol{\theta})$, where $\hat{\mathcal{L}}_k(\boldsymbol{\theta}) := \frac{1}{|\hat{\mathcal{D}}_k|} \sum_{x_i \in \hat{\mathcal{D}}_k} \left[ l(h(f(x_i; \boldsymbol{\theta}^f); \boldsymbol{\theta}^p), c^*(x_i)) \right]$ is the *empirical* risk over an observable dataset $\hat{\mathcal{D}}_k$. An implied assumption for FL is that the ***global*** data $\hat{\mathcal{D}}$ is distributed to each of the local domains, with $\hat{\mathcal{D}} = \cup \{\hat{\mathcal{D}}_k\}_{k=1}^K$.

**Knowledge Distillation** (KD) is also referred as a *teacher-student* paradigm, with the goal of learning a lightweight student model using knowledge distilled from one or more powerful teachers (Buciluǎ et al., 2006; Ba & Caruana, 2014). Typical KD leverages a ***proxy*** dataset $\hat{\mathcal{D}}_{\text{P}}$ to minimize the discrepancy between the logits outputs from the teacher model $\boldsymbol{\theta}_T$ and the student model $\boldsymbol{\theta}_S$, respectively. A representative choice is to use Kullback-Leibler divergence to measure such discrepancy (Hinton et al., 2015):

$$\min_{\boldsymbol{\theta}_S} \ \mathbb{E}_{x \sim \hat{\mathcal{D}}_{\text{P}}} \left[ D_{\text{KL}} \left[ \sigma(g(f(x; \boldsymbol{\theta}_T^f); \boldsymbol{\theta}_T^p) \| \sigma(g(f(x; \boldsymbol{\theta}_S^f); \boldsymbol{\theta}_S^p) \right] \right],$$

where $g(\cdot)$ is the logits output of an predictor $h$, and $\sigma(\cdot)$ is the non-linear activation applied to such logits, *i.e.* $h(z; \boldsymbol{\theta}^p) = \sigma(g(z; \boldsymbol{\theta}^p))$.

The idea of KD has been extended to FL to tackle user heterogeneity (Lin et al., 2020; Chen & Chao, 2021), by treating each user model $\boldsymbol{\theta}_k$ as the *teacher*, whose information is aggregated into the *student* (global) model $\boldsymbol{\theta}$ to improve its generalization performance:

$$\min_{\boldsymbol{\theta}} \ \mathop{\mathbb{E}}_{x \sim \hat{\mathcal{D}}_{\text{P}}} \left[ D_{\text{KL}} [\sigma(\frac{1}{K} \sum_{k=1}^K g(f(x; \boldsymbol{\theta}_k^f); \boldsymbol{\theta}_k^p)) \| \sigma(g(f(x; \boldsymbol{\theta}^f); \boldsymbol{\theta}^p)] \right].$$

One primary limitation of the above approach resides in its dependence on a proxy dataset $\hat{\mathcal{D}}_{\text{P}}$, the choice of which needs delicate consideration and plays a key role in the distillation performance (Lin et al., 2020). Next, we show how we make KD feasible for FL in a *data-free* manner.

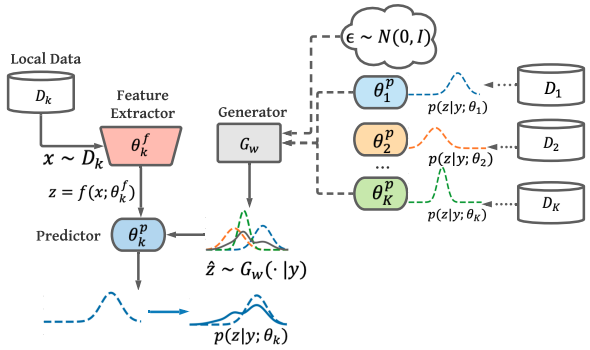



*Figure 1.* Overview of FEDGEN: a generator $G_{\boldsymbol{w}}(\cdot|y)$ is learned by the server to aggregate information from different local clients without observing their data. The generator is then sent to local users, whose knowledge is distilled to user models to adjust their interpretations of a good feature distribution.

**Algorithm 1** FEDGEN

**Require:** Tasks $\{\mathcal{T}_k\}_{k=1}^K$;
Global parameters $\boldsymbol{\theta}$, local parameters $\{\boldsymbol{\theta}_k\}_{k=1}^K$;
Generator parameter $\boldsymbol{w}$; $\hat{p}(y)$ uniformly initialized;
Learning rate $\alpha$, $\beta$, local steps $T$, batch size $B$, local label counter $c_k$.
**repeat**
  Server selects active users $\mathcal{A}$ uniformly at random, then broadcast $\boldsymbol{w}, \boldsymbol{\theta}, \hat{p}(y)$ to $\mathcal{A}$.
  **for** all user $k \in \mathcal{A}$ in parallel **do**
    $\boldsymbol{\theta}_k \leftarrow \boldsymbol{\theta}$,
    **for** $t = 1, \ldots, T$ **do**
      $\{x_i, y_i\}_{i=1}^B \sim \mathcal{T}_k, \{\hat{z}_i \sim G_{\boldsymbol{w}}(\cdot|\hat{y}_i), \hat{y}_i \sim \hat{p}(y)\}_{i=1}^B$.
      Update label counter $c_k$.
    $\boldsymbol{\theta}_k \leftarrow \boldsymbol{\theta}_k - \beta \nabla_{\boldsymbol{\theta}_k} J(\boldsymbol{\theta}_k)$. ▷ Optimize Equation 5
    **end for**
    User sends $\boldsymbol{\theta}_k$, $c_k$ back to server.
  **end for**
  Server updates $\boldsymbol{\theta} \leftarrow \frac{1}{|\mathcal{A}|} \sum_{k \in \mathcal{A}} \boldsymbol{\theta}_k$, and $\hat{p}(y)$ based on $\{c_k\}_{k \in \mathcal{A}}$.
  $\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \nabla_{\boldsymbol{w}} J(\boldsymbol{w})$. ▷ Optimize Equation 4
**until** training stop

## 3. FEDGEN: Data-Free Federated Distillation via Generative Learning

In this section, we elaborate our proposed approach with a summary shown in Algorithm 1. An overview of its learning procedure in illustrated in Figure 1.

### 3.1. Knowledge Extraction

Our core idea is to extract knowledge about the global view of data distribution, which is otherwise non-observable by conventional FL, and distill such knowledge to local models to guide their learning. We first consider learning a conditional distribution $Q^* : \mathcal{Y} \to \mathcal{X}$ to characterize such knowledge, which is consistent with the ground-truth data distributions:

$$Q^* = \underset{Q:\mathcal{Y}\to\mathcal{X}}{\arg\max}\ \mathbb{E}_{y\sim p(y)}\mathbb{E}_{x\sim Q(x|y)}[\log p(y|x)], \quad (2)$$

where $p(y)$ and $p(y|x)$ are the ground-truth *prior* and *posterior* distributions of the target labels, respectively, both of which are unknown. To make Equation 2 optimizable w.r.t $Q$, we replace $p(y)$ and $p(x|y)$ with their empirical approximations. First, we estimate $p(y)$ as:

$$\hat{p}(y) \propto \sum\nolimits_k \mathbb{E}_{x\sim\hat{\mathcal{D}}_k}[\mathrm{I}(c^*(x) = y)],$$

where $\mathrm{I}(\cdot)$ is an indicator function, and $\hat{\mathcal{D}}_k$ is the observable data for domain $\mathcal{T}_k$. In practice, $\hat{p}(y)$ can be obtained by requiring the training label counts from users during the model uploading phase. Next, we approximate $p(y|x)$ using the ensemble wisdom from user models:

$$\log \hat{p}(y|x) \propto \frac{1}{K} \sum\nolimits_{k=1}^K \log p(y|x; \boldsymbol{\theta}_k).$$

Equipped with the above approximations, directly optimizing Equation 2 over the input space $\mathcal{X}$ can still be prohibitive: it brings computation overloads when $\mathcal{X}$ is of high dimension, and may also leak information about the user data profile. A more approachable idea is hence to recover an *induced* distribution $G^* : \mathcal{Y} \to \mathcal{Z}$ over a latent space, which is more compact than the raw data space and can alleviate certain privacy-related concerns:

$$G^* = \underset{G:\mathcal{Y}\to\mathcal{Z}}{\arg\max}\ \mathbb{E}_{y\sim\hat{p}(y)}\mathbb{E}_{z\sim G(z|y)}\left[\sum_{k=1}^K \log p(y|z; \boldsymbol{\theta}_k^p)\right]. \quad (3)$$

Following the above reasoning, we aim to perform knowledge extraction by learning a conditional generator $G$, parameterized by $\boldsymbol{w}$ to optimize the following objective:

$$\min_{\boldsymbol{w}} J(\boldsymbol{w}) := \mathbb{E}_{y\sim\hat{p}(y)}\mathbb{E}_{z\sim G_{\boldsymbol{w}}(z|y)}\left[l(\sigma(\frac{1}{K}\sum_{k=1}^K g(z; \boldsymbol{\theta}_k^p)), y)\right], \quad (4)$$

where $g$ and $\sigma$ are the logit-output and the activation function as defined in Section 2. Given an arbitrary sample $y$, optimizing Equation 4 only requires access to the predictor modules $\boldsymbol{\theta}_k^p$ of user models. Specifically, to enable diversified outputs from $G(\cdot|y)$, we introduce a noise vector $\epsilon \sim \mathcal{N}(0, I)$ to the generator, which is resemblant to the reparameterization technique proposed by prior art (Kingma & Welling, 2014), so that $z \sim G_{\boldsymbol{w}}(\cdot|y) \equiv G_{\boldsymbol{w}}(y, \epsilon|\epsilon \sim \mathcal{N}(0, I))$. We discuss more implementation details in the supplementary.

Given arbitrary target labels $y$, the proposed generator can yield feature representations $z \sim G_{\boldsymbol{w}}(\cdot|y)$ that induce ideal predictions from the ensemble of user models. In other words, the generator approximates an induced image of a *consensual* distribution, which is consistent with the user data from a global view.

### 3.2. Knowledge Distillation

The learned generator $G_{\boldsymbol{w}}$ is then broadcasted to local users, so that each user model can sample from $G_w$ to obtain augmented representations $z \sim G_w(\cdot|y)$ over the feature space. As a result, the objective of a local model $\boldsymbol{\theta}_k$ is altered to maximize the probability that it yields ideal predictions for the augmented samples:

$$\min_{\boldsymbol{\theta}_k} J(\boldsymbol{\theta}_k) := \hat{\mathcal{L}}_k(\boldsymbol{\theta}_k) + \hat{\mathbb{E}}_{y\sim\hat{p}(y), z\sim G_{\boldsymbol{w}}(z|y)}\left[l(h(z; \boldsymbol{\theta}_k^p); y)\right], \quad (5)$$

where $\hat{\mathcal{L}}_k(\boldsymbol{\theta}_k) := \frac{1}{|\hat{\mathcal{D}}_k|}\sum_{x_i\in\hat{\mathcal{D}}_k}\left[l(h(f(x_i; \boldsymbol{\theta}_k^f); \boldsymbol{\theta}_k^p), c^*(x_i))\right]$ is the empirical risk given local data $\hat{\mathcal{D}}_k$. We show later that the augmented samples can introduce inductive bias to local users, reinforcing their model learning with a better generalization performance.

Up to this end, our proposed approach has realized data-free knowledge distillation, by interactively learning a lightweight generator that primarily depends on the prediction rule of local models, and leveraging the generator to convey consensual knowledge to local users. We justify in Section 6.2 that our approach can effectively handle user heterogeneity in FL, which also enjoys theoretical advantages as analyzed in Section 4.

### 3.3. Extensions for Flexible Parameter Sharing

In addition to tackling data heterogeneity, FEDGEN can also handle a challenging FL scenario where sharing the entire model is against communication or privacy prerequisites. On one hand, advanced networks with deep feature extraction layers typically contain millions of parameters (He et al., 2016; Brown et al., 2020), which bring significant burdens to communication. On the other hand, it has been shown feasible to backdoor regular FL approaches (Wang et al., 2020a). For practical FL applications such as healthcare or finance, sharing entire model parameters may be associated with considerable privacy risks, as discussed in prior work (He et al., 2020).

FEDGEN is ready to alleviate those problems, by sharing only the prediction layer $\boldsymbol{\theta}_k^p$ of local models, which is the primary information needed to optimizing Equation 4, while keeping the feature extractor $\boldsymbol{\theta}_k^f$ localized. This partial sharing paradigm is more efficient, and at the same time less vulnerable to data leakage, as compared with a strategy that shares the entire model. Empirical study in Section 6.4 shows that, FEDGEN significantly benefits local users, even without sharing feature extraction modules. We defer the algorithmic summary of this variant approach to the supplementary.

## 4. FEDGEN Analysis

In this section, we provide multiple perspectives to understand our proposed approach. We first visualize *what* knowledge is learned and distilled by FEDGEN, then analyze *why* the distilled knowledge is favorable, from the viewpoint of *distribution matching* and *domain adaptation*, respectively. We primarily focus on interpreting the rationale behind FEDGEN and leave detailed discussion and derivations to the supplementary.

### 4.1. Knowledge Distillation for Inductive Bias

We illustrate the KD process in FEDGEN on a FL prototype, which contains three users, each assigned with a disjoint dataset $\hat{\mathcal{D}}_k, k \in \{1,2,3\}$. When trained using only the local data, a user model is prone to learn biased decision boundaries (See Figure 2a).

Next, a generator $G_w(\cdot|y)$ is learned based on the prediction rule of user models. For clear visualizations, we learn $G_w(\cdot|y)$ on the raw feature space $\mathcal{Y} \to \mathcal{X} \subset \mathbb{R}^2$ instead of a latent space. As shown in Figure 3, $r(x|y)$, which denotes the distribution derived from $G_w(x|y)$, gradually coincides with the ground-truth $p(x|y)$ (Figure 2d), even when the individual local models are biased. In other words, $G_w(x|y)$ can fuse the aggregated information from user models to approximate a global data distribution.

We then let users sample from $G_w(x|y)$, which serves as an inductive bias for users with limited data. As a result, each user can observe beyond its own training data and adjust their decision boundaries to approach to the ensemble wisdom (Figure 2b).

### 4.2. Knowledge Distillation for Distribution Matching

A notable difference between FEDGEN and prior work is that the knowledge is distilled to user models instead of the global model. As a result, the distilled knowledge, which conveys inductive bias to users, can directly regulate their learning by performing *distribution matching* over the latent space $\mathcal{Z}$:

**Remark 1.** *Let $p(y)$ be the prior distribution of labels, and $r(z|y) : \mathcal{Y} \to \mathcal{Z}$ be the conditional distribution derived from generator $G_{\boldsymbol{w}}$. Then regulating a user model $\boldsymbol{\theta}_k$ using samples from $r(z|y)$ can minimize the conditional KL-divergence between two distributions, derived from the generator and the user, respectively:*

$$\begin{aligned} &\max_{\boldsymbol{\theta}_k} \ \mathbb{E}_{y\sim p(y), z\sim r(z|y)} \left[\log p(y|z;\boldsymbol{\theta}_k)\right] \\ \equiv &\min_{\boldsymbol{\theta}_k} \ D_{\mathrm{KL}}[r(z|y) \| p(z|y;\boldsymbol{\theta}_k)], \end{aligned} \tag{6}$$

where we define $p(z|y;\boldsymbol{\theta}_k)$ as the probability that the input feature to the predictor $\boldsymbol{\theta}_k$ is $z$ given that it yields a label $y$. In practice, Equation 6 is optimized by using empirical samples from the generator: $\{(z,y)|y \sim \hat{p}(y), z \sim G_w(z|y)\}$, which is consistent with the second term of the local model objective (Equation 5), in that $\forall\ y \in \mathcal{Y}$:

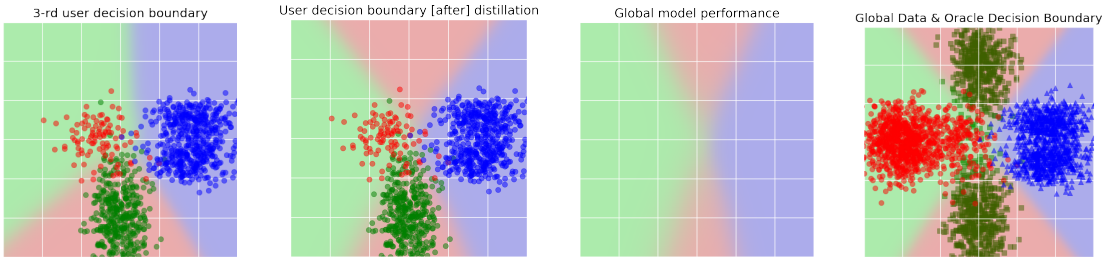
$$\max_{\boldsymbol{\theta}_k} \ \mathbb{E}_{z\sim r(z|y)} \left[\log p(y|z;\boldsymbol{\theta}_k)\right] \approx \min_{\boldsymbol{\theta}_k} \ \mathbb{E}_{z\sim G_{\boldsymbol{w}}(z|y)} \left[l(h(z;\boldsymbol{\theta}_k^p); y)\right].$$

Distinguished from prior work that applies weight regularization to local models (Li et al., 2020b; Dinh et al., 2020), FEDGEN can serve as an alternative and compatible solution to address user heterogeneity, which inherently bridges the gap among user models w.r.t their interpretations of an ideal feature distribution.

### 4.3. Knowledge Distillation for Improved Generalization

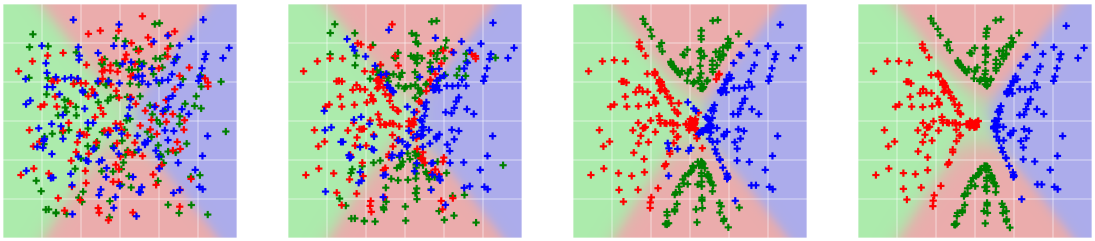
One can also draw a theoretical connection from the knowledge learned by FEDGEN to an improved generalization bound. To see this, we first present a performance bound for the aggregated model in FL, which is built upon prior arts from *domain adaptation* (Ben-David et al., 2007; 2010):

**Theorem 1.** **(Generalization Bounds for FL)** *Consider an FL system with $K$ users. Let $\mathcal{T}_k = \langle \mathcal{D}_k, c^* \rangle$ and $\mathcal{T} = \langle \mathcal{D}, c^* \rangle$ be the $k$-th local domain and the global domain, respectively. Let*




(a) User decision boundaries ***before*** KD. (b) User decision boundaries ***after*** KD. (c) Global model decision boundaries. (d) *Oracle* decision boundaries learned over all data.

*Figure 2.* After KD, accuracy has improved from $81.2\%$ to $98.4\%$ for one user ( Fig 2a - Fig 2b), while a global model obtained by parameter-averaging (*without* KD) has $93.2\%$ accuracy (Fig 2c), compared with an oracle model with $98.6\%$ accuracy (Fig 2d).



(a) Randomly initialized $r(x|y)$. (b) $r(x|y)$ learned after 50 training steps. (c) $r(x|y)$ learned after 150 training steps. (d) $r(x|y)$ learned after 250 training steps.

*Figure 3.* Samples from the generator gradually approaches to ground-truth distribution, where each user model (teacher) sees limited, disjoint local data. Background color indicates oracle decision boundaries learned over the global data.

*$\mathcal{R}: \mathcal{X} \to \mathcal{Z}$ be a feature extraction function that is simultaneously shared among users. Denote $h_k$ the hypothesis learned on domain $\mathcal{T}_k$, and $h = \frac{1}{K}\sum_{k=1}^{K} h_k$ the global ensemble of user hypotheses. Then with probability at least $1-\delta$:*

$$\mathcal{L}_{\mathcal{T}}(h) \equiv \mathcal{L}_{\mathcal{T}}\left(\frac{1}{K}\sum_k h_k\right) \leq \frac{1}{K}\sum_k \hat{\mathcal{L}}_{\mathcal{T}_k}(h_k) + \frac{1}{K}\sum_k \left(d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}}) + \lambda_k\right) + \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4K}{\delta}\right)},$$

*where $\hat{\mathcal{L}}_{\mathcal{T}_k}(h_k)$ is the empirical risk on $\mathcal{T}_k$, $\lambda_k := \min_h(\mathcal{L}_{\mathcal{T}_k}(h) + \mathcal{L}_{\mathcal{T}}(h))$ denotes an oracle performance. $d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}})$ denotes the divergence measured over a symmetric-difference hypothesis space. $\tilde{\mathcal{D}}_k$ and $\tilde{\mathcal{D}}$ is the **induced** image of $\mathcal{D}_k$ and $\mathcal{D}$ over $\mathcal{R}$, respectively, s.t. $\mathbb{E}_{z\sim\tilde{\mathcal{D}}_k}[\mathcal{B}(z)] = \mathbb{E}_{x\sim\mathcal{D}_k}[\mathcal{B}(\mathcal{R}(x))]$ given a probability event $\mathcal{B}$, and so for $\tilde{\mathcal{D}}$.*

Specifically, $\mathcal{L}_{\mathcal{T}}(h)$ is usually considered as an ideal upper-bound for the global model in FL (Peng et al., 2019; Lin et al., 2020). Two key implications can be derived from Theorem 1: i) Large user ***heterogeneity*** leads to high distribution divergence ($d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}})$), which undermines the quality of the global model; ii) More empirical samples ($m$) are favorable to the generalization performance, which softens the numerical constraints.

In other words, the generalization performance can be improved by enriching local users with augmented data that aligns with the global distribution:

**Corollary 1.** *Let $\mathcal{T}$, $\mathcal{T}_k$, $\mathcal{R}$ defined as in Theorem 1. $\mathcal{D}_{\text{A}}$ denotes an **augmented** distribution, and $\mathcal{D}'_k = \frac{1}{2}(\mathcal{D}_k + \mathcal{D}_{\text{A}})$ is a **mixture** of distributions. Accordingly, $\tilde{\mathcal{D}}_{\text{A}}$, $\tilde{\mathcal{D}}'_k$ denotes the **induced** image of $\mathcal{D}_{\text{A}}$, $\mathcal{D}'_k$ over $\mathcal{R}$, respectively. Let $\hat{\mathcal{D}}'_k = \hat{\mathcal{D}}_k \cup \hat{\mathcal{D}}_{\text{A}}$ be an empirical dataset of $\mathcal{D}'_k$, with $|\hat{\mathcal{D}}_k| = m$, $|\hat{\mathcal{D}}'_k| = m' > m$. If $d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A, \tilde{\mathcal{D}})$ is bounded, s.t $\exists\, \epsilon > 0, d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A, \tilde{\mathcal{D}}) \leq \epsilon$, then with probability $1-\delta$:*

$$\mathcal{L}_{\mathcal{T}}(h) \leq \frac{1}{K}\sum_k \mathcal{L}_{\mathcal{T}'_k}(h_k) + \frac{1}{K}\sum_k (d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k, \tilde{\mathcal{D}}) + \lambda'_k) + \sqrt{\frac{4}{m'}\left(d\log\frac{2em'}{d} + \log\frac{4K}{\delta}\right)}, \tag{7}$$

*where $\mathcal{T}'_k = \{\mathcal{D}'_k, c^*\}$ is the updated local domain, $d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k, \tilde{\mathcal{D}}) \leq d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}})$ when $\epsilon$ is small, and $\sqrt{\frac{4}{m'}(d\log\frac{2em'}{d} + \log\frac{4K}{\delta})} < \sqrt{\frac{4}{m}(d\log\frac{2em}{d} + \log\frac{4K}{\delta})}$.*

Such an augmented distribution $\mathcal{D}_{\text{A}}$ can facilitate FL from multiple aspects: not only does it relax the numerical constraints with more empirical samples ($m' > m$), but it also reduces the discrepancy between the local and global feature distributions ($d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k, \tilde{\mathcal{D}})$). This finding coincides the merits of FEDGEN: since the generator $G_w(z|y)$ is learned to recover an aggregated distribution over the feature space, one can treat samples from the generator $\{z|y \sim \hat{p}(y), z \sim G_w(z|y)\}$ as the augmented data from $\tilde{\mathcal{D}}_{\text{A}}$, which naturally has a small deviation from the global induced distribution $\tilde{\mathcal{D}}$. More rigorous analysis along this line is left to our future work. We elaborate the role of such an augmentation distribution $\mathcal{D}_A$ in the supplementary.

## 5. Related Work

**Federated Learning** (FL) is first proposed by (McMahan et al., 2017) as a decentralized machine learning paradigm. Subsequent work along this line tackles different challenges faced by FL, including heterogeneity (Karimireddy et al., 2020; Li et al., 2020b; Mansour et al., 2020), privacy (Duchi et al., 2014; Agarwal et al., 2018), communication efficiency (Guha et al., 2019; Konečnỳ et al., 2016), and convergence analysis (Kairouz et al., 2019; Qu et al., 2020; Yuan & Li, 2019). Specifically, a wealth of work has been proposed to handle user ***heterogeneity***, by regularizing model weight updates (Li et al., 2020b), allowing personalized user models (Fallah et al., 2020; Dinh et al., 2020), or introducing new model aggregation schemes (Yurochkin et al., 2019; Mansour et al., 2020). We refer readers to (Li et al., 2020a) for an organized discussion of recent progress on FL.

**Knowledge Distillation** (KD) is a technique to compress knowledge from one or more teacher models into an empty student (Hinton et al., 2015; Buciluǎ et al., 2006; Ba & Caruana, 2014; Jacobs et al., 1991). Conventional KD hinges on a proxy dataset (Hinton et al., 2015). More recent work enables KD with fewer data involved, such as dataset distillation (Wang et al., 2018), or core-data selection (Tsang et al., 2005; Sener & Savarese, 2018). Later there emerges data-free KD approaches which aim to reconstruct samples used for training the teacher (Yoo et al., 2019; Micaelli & Storkey, 2019). Particularly, (Lopes et al., 2017) extracts the meta-data from the teacher's activation layers. (Yoo et al., 2019) learns a conditional generator which yield samples that maximizes the teacher's prediction probability of a target label. Along the same spirit, (Micaelli & Storkey, 2019) learns a generator by adversarial training. Different from prior work, we learn a generative model that is tailored for FL, by ensembling the knowledge of multiple user models over the latent space, which is more lightweight for learning and communication.

**Knowledge Distillation in Federated Learning** has recently emerged as an effective approach to tackle user heterogeneity. Most existing work is data-dependent (Lin et al., 2020; Sun & Lyu, 2020; Guha et al., 2019; Chen & Chao, 2021). Particularly, (Lin et al., 2020) proposed **FEDDFUSION**, which performs KD to refine the global model, assuming that an unlabeled dataset is available with samples from the same or similar domains. Complementary KD efforts have been made to confront data heterogeneity (Li & Wang, 2019; Sattler, 2021). Specifically, (Li & Wang, 2019) transmits the proxy dataset instead of the model parameters. FEDAUX (Sattler, 2021) performs *data-dependent* distillation by leveraging an auxiliary dataset to initialize the server model and to weighted-ensemble user models, while FEDGEN performs knowledge distillation in a data-free manner. FEDMIX (Yoon, 2021) is a *data-augmented* FL framework, where users share their *batch-averaged* data among others to assist local training. On the country, FEDGEN extracts knowledge from the existing user model parameters, which faces less privacy risks. **FEDDISTILL** (Federated Distillation) is proposed by (Seo et al., 2020) which extracts from user models the statistics of the logit-vector outputs, and shares this meta-data to users for KD. We provide detailed comparisons with work along this line in Section 6.

## 6. Experiments

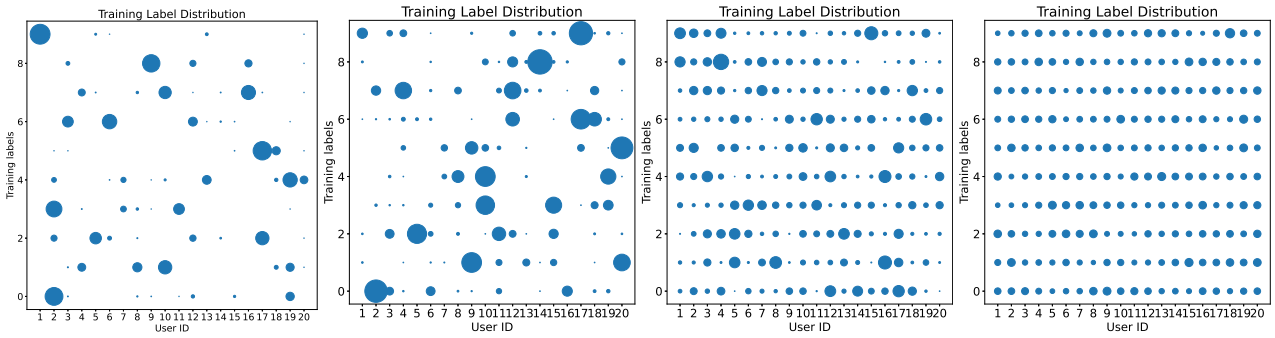
In this section, we compare the performance of our proposed approach with other key related work. We leave implementation details and extended experimental results to the supplementary.

### 6.1. Setup

**Baselines:** In addition to **FEDAVG** (McMahan et al., 2017), **FEDPROX** regularizes the local model training with a proximal term in the model objective (Li et al., 2020b). **FEDENSEMBLE** extends FEDAVG to ensemble the prediction output of all user models. **FEDDFUSION** is a data-based KD approach (Lin et al., 2020), for which we provide unlabeled training samples as the proxy dataset. **FEDDISTILL** (Jeong et al., 2018) is a data-free KD approach which shares label-wise average of logit-vectors among users. It does not share network parameters and therefore experience non-negligible performance drops compared with other baselines. For a fair comparison, we derive a baseline from FEDDISTILL, which shares both model parameters and the label-wise logit vectors. We name this stronger baseline as **FEDDISTILL**$^+$.

**Dataset:** We conduct experiments on three image datasets: **MNIST** (LeCun & Cortes, 2010), **EMNIST** (Cohen et al., 2017), and **CELEBA** (Liu et al., 2015), as suggested by the LEAF FL benchmark (Caldas et al., 2018). Among them, MNIST and EMNIST dataset is for digit and character image classifications, and CELEBA is a celebrity-face dataset which is used to learn a binary-classification task, *i.e.* to predict whether the celebrity in the picture is smiling.

**Configurations:** Unless otherwise mentioned, we run 200 global communication rounds, with 20 user models in total and an active-user ratio $r = 50\%$. We adopt a local updating step $T = 20$, and each step uses a mini batch with size $B = 32$. We use at most $50\%$ of the total training dataset and distribute it to user models, and use all testing dataset for performance evaluation. For the classifier, we follow the network architecture of (McMahan et al., 2017), and treat the last MLP layer as the predictor $\boldsymbol{\theta}_k^p$ and all previous layers as the feature extractor $\boldsymbol{\theta}_k^f$. The generator $G_{\boldsymbol{w}}$ is MLP based. It takes a noise vector $\epsilon$ and an one-hot label vector $y$ as the input, which, after a hidden layer with dimension $d_h$, outputs a feature representation with dimension $d$. To further increase the diversity of the generator output, we also leverage the idea of *diversity loss* from prior work (Mao

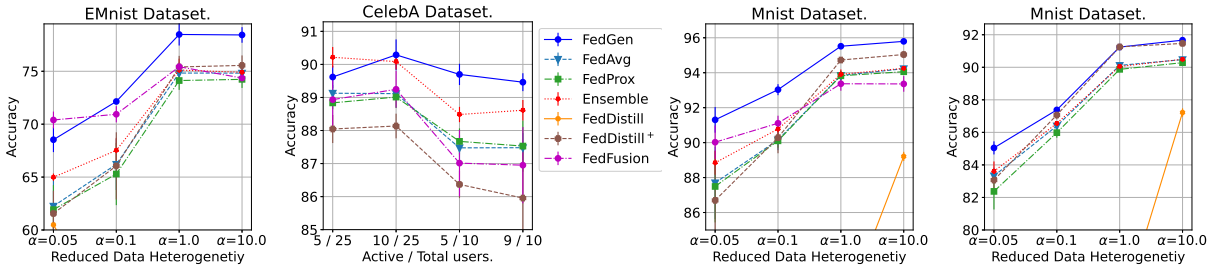



(a) $\alpha = 0.05$ (b) $\alpha = 0.1$ (c) $\alpha = 1$ (d) $\alpha = 10$

*Figure 4.* Visualization of statistical heterogeneity among users on MNIST dataset, where the $x$-axis indicates user IDs, the $y$-axis indicates class labels, and the size of scattered points indicates the number of training samples for a label available to that user.

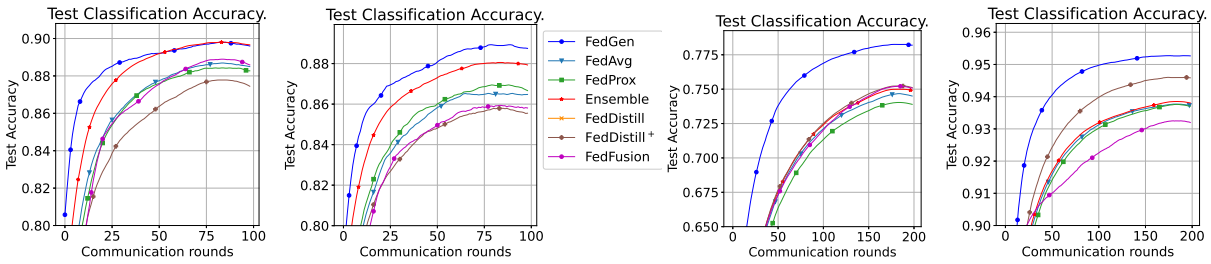
| | | Top-1 Test Accuracy. | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Dataset | Setting | FEDAVG | FEDPROX | FEDENSEMBLE | FEDDISTILL | FEDDISTILL$^+$ | FEDDFUSION | FEDGEN |
| MNIST, $T$=20 | $\alpha$ = 0.05 | 87.70±2.07 | 87.49±2.05 | 88.85±0.68 | 70.56±1.24 | 86.70±2.27 | 90.02±0.96 | **91.30±0.74** |
| | $\alpha$ = 0.1 | 90.16±0.59 | 90.10±0.39 | 90.78±0.39 | 64.11± 1.36 | 90.28±0.89 | 91.11±0.43 | **93.03±0.32** |
| | $\alpha$ = 1 | 93.84±0.25 | 93.83 ± 0.29 | 93.91±0.28 | 79.88±0.66 | 94.73±0.15 | 93.37±0.40 | **95.52±0.07** |
| CELEBA, $T$=20 | $r = 5/10$ | 87.48±0.39 | 87.67±0.39 | 88.48±0.23 | 76.68±1.23 | 86.37±0.41 | 87.01±1.00 | **89.70±0.32** |
| | $r = 5/25$ | 89.13±0.25 | 88.84±0.19 | **90.22±0.31** | 74.99±1.57 | 88.05± 0.43 | 88.93±0.79 | 89.62±0.34 |
| | $r = 10/25$ | 89.12±0.20 | 89.01±0.33 | 90.08±0.24 | 75.88±1.17 | 88.14±0.37 | 89.25±0.56 | **90.29±0.47** |
| EMNIST, $T$=20 | $\alpha$ = 0.05 | 62.25±2.82 | 61.93±2.31 | 64.99±0.35 | 60.49±1.27 | 61.56±2.15 | **70.40±0.79** | 68.53±1.17 |
| | $\alpha$ = 0.1 | 66.21±2.43 | 65.29±2.94 | 67.53±1.19 | 50.32±1.39 | 66.06±3.18 | 70.94±0.76 | **72.15±0.21** |
| | $\alpha$ = 10 | 74.83± 0.69 | 74.24±0.81 | 74.90±0.80 | 54.77±0.33 | 75.55 ±0.94 | 74.36±0.40 | **78.43±0.74** |
| EMNIST, $\alpha$=1 | $T$ = 20 | 74.83±0.99 | 74.12±0.88 | 75.12±1.07 | 46.19±0.70 | 75.41±1.05 | 75.43±0.37 | **78.48±1.04** |
| | $T$ = 40 | 77.02±1.09 | 75.93 ±0.95 | 77.68±0.98 | 46.72±0.73 | 78.12±0.90 | 77.58±0.37 | **78.92± 0.73** |

*Table 1.* Performance overview given different data settings. For MNIST and EMNIST, a *smaller* $\alpha$ indicates *higher* heterogeneity. For CELEBA, $r$ denotes the ratio between active users and total users. $T$ denotes the local training steps (communication delay).




(a) EMNIST Dataset. (b) CELEBA Dataset. (c) MNISTon CNN. (d) MNIST on MLP.

*Figure 5.* Visualized performance w.r.t data heterogeneity.




(a) CELEBA, $r = 10/25$. (b) CELEBA, $r = 5/10$. (c) EMNIST, $\alpha = 1$. (d) MNIST, $\alpha = 1$.

*Figure 6.* Selected learning curves, averaged over 3 random seeds.

et al., 2019) to train the generator model.

**User heterogeneity**: for MNIST and EMNIST dataset, we follow prior arts (Lin et al., 2020; Hsu et al., 2019) to model non-iid data distributions using a Dirichlet distribution **Dir**$(\alpha)$, in which a smaller $\alpha$ indicates higher data heterogeneity, as it makes the distribution of $p_k(y)$ more biased for a user $k$. We visualize the effects of adopting different $\alpha$ on the statistical heterogeneity for the MNIST dataset in Figure 4. For CELEBA, the raw data is naturally non-iid distributed. We further increase the data heterogeneity by aggregating pictures belonging to different celebrities into disjoint groups, with each group assigned to one user.

### 6.2. Performance Overview:

From Table 1, we can observe that FEDGEN outperforms other baselines with a considerable margin.

**Impacts of data heterogeneity:** FEDGEN is the only algorithm that is robust against different levels of user heterogeneity while consistently performs well. As shown in Figure 5, the gain of FEDGEN is more notable when the data distributions are highly heterogeneous (with a small $\alpha$). This result verifies our motivations, since the advantage of FEDGEN is induced from the knowledge distilled to local users, which mitigates the discrepancy of latent distributions across users. This knowledge is otherwise not accessible by baselines such as FEDAVG or FEDPROX.

As one of most competitive baselines, the advantage of FEDDFUSION vanishes as data heterogeneity becomes mitigated, which gradually becomes comparable to FEDAVG, as shown in Figure 5a and Figure 5c. Unlike FEDDFUSION, the performance gain of our approach is consistently significant, which outperforms FEDDFUSION in most cases. This discrepancy implies that our proposed approach, which directly distills the knowledge to user models, can be more effective than fine-tuning the global model using a proxy dataset, especially when the distilled knowledge contains inductive bias to guide local model learning.

As a data-free KD baseline, FEDDISTILL experiences non-negligible performance drops, which implies the importance of parameter sharing in FL. FEDDISTILL$^+$, on the other hand, is vulnerable to data heterogeneities. As shown in Table 1, it can outperform FEDAVG when data distributions are near-iid (*e.g.* when $\alpha \geq 1$ ), thanks to the shared logit statistics as the distilled knowledge, but performs worse than FEDAVG when $\alpha$ gets smaller, which indicates that sharing such meta-data alone may not be effective enough to confront user heterogeneity.

FEDENSEMBLE enjoys the benefit of ensemble predictions from all user models, although its gain is less significant compared with FEDGEN. We ascribe the leading performance of our approach to the better generalized performance of local models. Guided by the distilled knowledge, a user model in FEDGEN can quickly jump out of its local optimum, whose aggregation can be better than the ensemble of potentially biased models as in FEDENSEMBLE.

**Learning efficiency:** As shown in Figure 6, FEDGEN has the most rapid learning curves to reach a performance and outperforms other baselines. Although FEDDFUSION enjoys a learning efficiency higher than other baselines under certain data settings, due the advantages induced from a proxy data, our approach can directly benefit each local user with actively learned knowledge, whose effect is more explicit and consistent (More illustrations in supplementary).

**Comments on sharing generative model**: Given a compact latent space, the generative model can be lightweight for learning or downloading. In practice, we use a generator network with 2 compact MLP layers, whose parameter size is small compared with the user classification model. The above empirical results also indicates that the leading performance gain combined with a faster convergence rate can trade off the communication load brought by sharing a generative model.

### 6.3. Sensitivity Analysis

**Impacts of straggler users:** We explore different numbers of total users versus active users on the CELEBA dataset, with the active ratios $r$ ranging from $0.2$ to $0.9$. Figure 5b shows that our approach is next to FEDENSEMBLE when the number of straggler users are high ($r = 0.2$, with 5 out 25 active users per learning round), and is consistently better than all baselines w.r.t to the asymptotic performance given a moderate number of active users. Combined with Figure 6a and Figure 6b, one can observe that our approach requires much less communication rounds to reach high performance, regardless of the setting of straggler users.

**Effects of different network architectures:** we conduct analysis on the MNIST dataset, using both CNN and MLP network architectures. As shown in Figure 5d and Figure 5c, the outstanding performance of FEDGEN is consistent across two different network settings, although the overall performance trained with CNN networks is noticeably higher than those with MLP networks.

**Effects of communication frequency:** We explore different local updating steps $T$ on the EMNIST, so that a higher $T$ means longer communication delays before the global communication. Results in Table 1 indicates that our approach is robust against different levels of communication delays (See supplementary for more results).

**Effects of the generator's network architecture and sampling size:** Extended analysis has verified that FEDGEN is *robust* across different generator network architectures (Table 2). Moreover, sampling synthetic data from the generator only adds minor training workload to local users (Table 2). The gain of FEDGEN over FEDAVG is con-

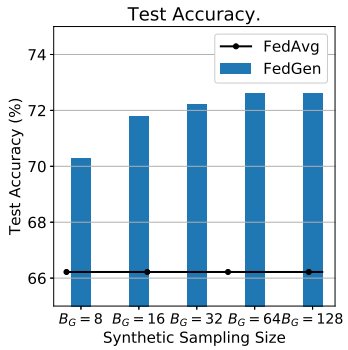


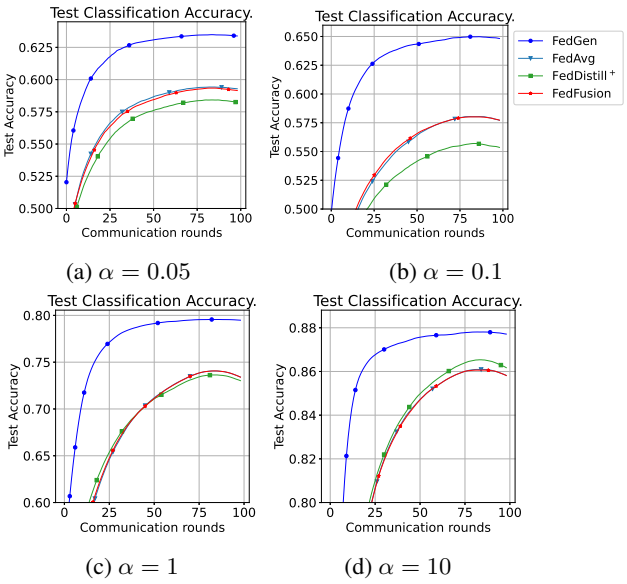

*Figure 7.* Effects of synthetic samples.

| | Effects of the Generator Network Structure. | | | | |
|---|---|---|---|---|---|
| $[d_\epsilon, d_h]$ | $[64, 256]$ | $[32, 256]$ | $[32, 128]$ | $[16, 128]$ | $[32, 64]$ |
| **Accuracy(%)** | FEDAVG=66.22±2.58 | | | | |
| FEDGEN | 71.61±0.25 | 72.09±0.46 | 72.43±0.57 | 72.01±0.76 | 70.98±0.85 |

*Table 2.* Effects of the generator's network structure, using EMNIST dataset with $\alpha = 0.1$.

| | Performance w.r.t different synthetic sample sizes. | | | | |
|---|---|---|---|---|---|
| **Generator sampling size** | $B_G = 8$ | $B_G = 16$ | $B_G = 32$ | $B_G = 64$ | $B_G = 128$ |
| **Local training time (ms)** | FEDAVG= $47.66 \pm 1.68$ | | | | |
| FEDGEN | 57.20±2.22 | 57.39±2.21 | 58.17±2.24 | 58.91±2.29 | 60.06±2.32 |

*Table 3.* Effects of the number of synthetic samples, using EMNIST dataset with $\alpha = 0.1$.




(a) $\alpha = 0.05$ (b) $\alpha = 0.1$

(c) $\alpha = 1$ (d) $\alpha = 10$

*Figure 8.* Learning curves on MNIST with limited param. sharing.

| | Top 1 Accuracy (%) | | | |
|---|---|---|---|---|
| Algorithms | FEDAVG | FEDDISTILL$^+$ | FEDDFUSION | FEDGEN |
| $\alpha = 0.05$ | 59.67±0.76 | 58.83±0.62 | 59.62±0.84 | **63.60±0.63** |
| $\alpha = 0.1$ | 58.39±0.74 | 56.25±0.98 | 58.38±0.81 | **65.42±0.29** |
| $\alpha = 1$ | 74.49± 0.57 | 74.24±0.60 | 74.51±0.55 | **79.72±0.52** |
| $\alpha = 10$ | 86.35±0.60 | 86.89±0.26 | 86.28±0.69 | **87.92±0.46** |

*Table 4.* Performance overview on MNIST, by only sharing the last prediction layer.

sistently remarkable given different synthetic sample sizes, whereas a sufficient number of synthetic samples brings even better performance (Figure 7). Especially, in Table 2, we explored different dimensions for the input noise ($d_\epsilon$) and the hidden layer ($d_h$) of the generator while keeping its output layer dimension fixed (*i.e.* the dimension of the feature space $\mathcal{Z}$). Table 3 shows the training time for one local update, averaged across users and the communication rounds. $B_G$ denotes the number of *synthetic* samples used for each mini-batch optimization. By default, we set $B_G = B$, and $B$ is the number of *real* samples drawn from the local dataset (see Algorithm 1).

### 6.4. Extensions to Flexible Parameter Sharing

Motivated to alleviate privacy and communication concerns, FEDGEN is ready to benefit distributed learning without sharing entire model parameters. To explore this potential, we conduct a case study on FEDAVG, FEDDISTILL$^+$, and FEDGEN, where user models share only the last prediction layer and keep their feature extraction layers localized. Note that FEDDFUSION is not designed to address FL with partial parameter sharing, which requires entire user models for KD. For a fair comparison, we modify FEDDFUSION to let it upload entire user models during the model aggregation phase, but disable the downloading of feature extractors, so that the server model can still be fine-tuned using the proxy data.

Results in Table 4 show that our approach consistently outperforms other baselines by a remarkable margin, the trend of which is more significant given high data heterogeneity (Figure 8). Its distinguished performance from FEDDFUSION verifies the efficacy of data-free distillation under this challenging scenario. This promising results show that FEDGEN has the potential to further reduce communication workload, not only by fast convergence but also by a flexible parameter sharing strategy.

## 7. Conclusions

In this paper, we propose an FL paradigm that enables efficient knowledge distillation to address user heterogeneity without requiring any external data. Extensive empirical experiments, guided by theoretical implications, have shown that our proposed approach can benefit federated learning with better generalization performance using less communication rounds, compared with the state-of-the-art.

### Acknowledgments

This research was jointly supported by the National Science Foundation IIS-1749940, the Office of Naval Research N00014-20-1-2382, and the National Institue on Aging RF1AG072449.

## References


Agarwal, N., Suresh, A. T., Yu, F., Kumar, S., and Mcmahan, H. B. cpsgd: Communication-efficient and differentially-private distributed sgd. *Advances in Neural Information Processing Systems*, 2018.

Ammad-Ud-Din, M., Ivannikova, E., Khan, S. A., Oyomno, W., Fu, Q., Tan, K. E., and Flanagan, A. Federated collaborative filtering for privacy-preserving personalized recommendation system. *arXiv preprint arXiv:1901.09888*, 2019.

Ba, J. and Caruana, R. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27:2654–2662, 2014.

Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*, pp. 137–144, 2007.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. W. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Wortman, J. Learning bounds for domain adaptation. In *Advances in neural information processing systems*, pp. 129–136, 2008.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 2020.

Buciluǎ, C., Caruana, R., and Niculescu-Mizil, A. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.

Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečnỳ, J., McMahan, H. B., Smith, V., and Talwalkar, A. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Chen, H.-Y. and Chao, W.-L. Fedbe: Making bayesian model ensemble applicable to federated learning. *ICLR*, 2021.

Cohen, G., Afshar, S., Tapson, J., and Van Schaik, A. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Dinh, C. T., Tran, N. H., and Nguyen, T. D. Personalized federated learning with moreau envelopes. *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020.

Duchi, J. C., Jordan, M. I., and Wainwright, M. J. Privacy aware learning. *Journal of the ACM (JACM)*, 61(6):1–57, 2014.

Fallah, A., Mokhtari, A., and Ozdaglar, A. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33, 2020.

Guha, N., Talwalkar, A., and Smith, V. One-shot federated learning. *arXiv preprint arXiv:1902.11175*, 2019.

Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

He, C., Annavaram, M., and Avestimehr, S. Group knowledge transfer: Federated learning of large cnns at the edge. *Advances in Neural Information Processing Systems*, 33, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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

Hsu, T.-M. H., Qi, H., and Brown, M. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. Adaptive mixtures of local experts. *Neural computation*, 3(1): 79–87, 1991.

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

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020.

Khoussainov, R., Heß, A., and Kushmerick, N. Ensembles of biased classifiers. In *Proceedings of the 22nd international conference on Machine learning*, pp. 425–432, 2005.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *ICLR*, 2014.

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

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

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020a.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020b.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of fedavg on non-iid data. *ICLR*, 2020c.

Lin, T., Kong, L., Stich, S. U., and Jaggi, M. Ensemble distillation for robust model fusion in federated learning. *arXiv preprint arXiv:2006.07242*, 2020.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Lopes, R. G., Fenu, S., and Starner, T. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*, 2017.

Mansour, Y., Mohri, M., Ro, J., and Suresh, A. T. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

Mao, Q., Lee, H.-Y., Tseng, H.-Y., Ma, S., and Yang, M.-H. Mode seeking generative adversarial networks for diverse image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1429–1437, 2019.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR, 2017.

Micaelli, P. and Storkey, A. J. Zero-shot knowledge transfer via adversarial belief matching. In *Advances in Neural Information Processing Systems*, pp. 9551–9561, 2019.

Peng, X., Huang, Z., Zhu, Y., and Saenko, K. Federated adversarial domain adaptation. *arXiv preprint arXiv:1911.02054*, 2019.

Qu, Z., Lin, K., Kalagnanam, J., Li, Z., Zhou, J., and Zhou, Z. Federated learning's blessing: Fedavg has linear speedup. *arXiv preprint arXiv:2007.05690*, 2020.

Sattler, F. e. a. Fedaux: Leveraging unlabeled auxiliary data in federated learning. *arXiv preprint arXiv:2102.02514*, 2021.

Sener, O. and Savarese, S. Active learning for convolutional neural networks: A core-set approach. *ICLR*, 2018.

Seo, H., Park, J., Oh, S., Bennis, M., and Kim, S.-L. Federated knowledge distillation. *arXiv preprint arXiv:2011.02367*, 2020.

Sheller, M. J., Edwards, B., Reina, G. A., Martin, J., Pati, S., Kotrotsou, A., Milchenko, M., Xu, W., Marcus, D., Colen, R. R., et al. Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Scientific reports*, 10(1):1–12, 2020.

Sun, L. and Lyu, L. Federated model distillation with noise-free differential privacy. *arXiv preprint arXiv:2009.05537*, 2020.

Tsang, I. W., Kwok, J. T., and Cheung, P.-M. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, 6(Apr):363–392, 2005.

Wang, H., Sreenivasan, K., Rajput, S., Vishwakarma, H., Agarwal, S., Sohn, J.-y., Lee, K., and Papailiopoulos, D. Attack of the tails: Yes, you really can backdoor federated learning. *arXiv preprint arXiv:2007.05084*, 2020a.

Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D., and Khazaeni, Y. Federated learning with matched averaging. *ICLR*, 2020b.

Wang, T., Zhu, J.-Y., Torralba, A., and Efros, A. A. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Yoo, J., Cho, M., Kim, T., and Kang, U. Knowledge extraction with no observable data. In *Advances in Neural Information Processing Systems*, pp. 2705–2714, 2019.

Yoon, T. e. a. Fedmix: Approximation of mixup under mean augmented federated learning. *ICLR*, 2021.

Yuan, X.-T. and Li, P. On convergence of distributed approximate newton methods: Globalization, sharper bounds and beyond. *arXiv preprint arXiv:1908.02246*, 2019.

Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, N., and Khazaeni, Y. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*, pp. 7252–7261. PMLR, 2019.

# Data-Free Knowledge Distillation for Heterogeneous Federated Learning: Supplementary Document

## 8. Theoretical Derivations

### 8.1. Notations and Preliminaries

Let $\mathcal{X} \subset \mathbb{R}^p$ be the *input* space, $\mathcal{Z} \subset \mathbb{R}^d$ be the *latent* feature space, and $\mathcal{Y} \subset \mathbb{R}$ be the output space. $\mathcal{R} : \mathcal{X} \to \mathcal{Z}$ denotes a ***representation function*** that maps inputs into features. $\mathcal{T}$ denotes a ***domain*** (or *task*), which consists of a data distribution $\mathcal{D}$ over $\mathcal{X}$ and a ground-truth *labeling* function $c^* : \mathcal{X} \to \mathcal{Y}$. Given a domain $\mathcal{T} := \langle \mathcal{D}, c^* \rangle$ and a representation function $\mathcal{R}$, we use $\tilde{\mathcal{D}}$ to denote the *induced image* of $\mathcal{D}$ under $\mathcal{R}$ (Ben-David et al., 2007), *s.t.* given a probability event $\mathcal{B}$,

$$\mathbb{E}_{z\sim\tilde{\mathcal{D}}}[\mathcal{B}(z)] = \mathbb{E}_{x\sim\mathcal{D}}[\mathcal{B}(\mathcal{R}(x))].$$

Accordingly, $\tilde{c}^*$ denotes the *induced* labeling function under $\mathcal{R}$:

$$\tilde{c}^*(z) := \mathbb{E}_{x\sim D}\left[c^*(x)|\mathcal{R}(x) = z\right].$$

Let $h : \mathcal{Z} \to \mathcal{Y}$ denote a ***hypothesis*** that maps features to predicted labels, and $\mathcal{H} \subseteq \{h : \mathcal{Z} \to \mathcal{Y}\}$ denote a hypothesis class. For our analysis, we assume the FL tasks are for binary classification, *i.e.* $\mathcal{Y} = \{0, 1\}$, and the loss function is 0-1 bounded, with $l(\hat{y}, y) = |\hat{y} - y|$. Same assumptions have been adopted by various prior art (Ben-David et al., 2007; Blitzer et al., 2008; Ben-David et al., 2010; Lin et al., 2020; Ben-David et al., 2007).

Given two distributions $\mathcal{D}$ and $\mathcal{D}'$, $d_{\mathcal{H}}(\mathcal{D}, \mathcal{D}')$ is defined as the $\mathcal{H}$-divergence between $\mathcal{D}$ and $\mathcal{D}'$, *i.e.*:

$$d_{\mathcal{H}}(\mathcal{D}, \mathcal{D}') := 2 \sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}}} |\Pr_{\mathcal{D}}(\mathcal{A}) - \Pr_{\mathcal{D}'}(\mathcal{A})|\},$$

where $\mathcal{A}_{\mathcal{H}}$ is a set of measurable subsets under $\mathcal{D}$ and $\mathcal{D}'$ for certain $h \in \mathcal{H}$. Moreover, $\mathcal{H}\triangle\mathcal{H}$ is defined as the symmetric difference hypothesis space (Blitzer et al., 2008), *i.e.*:

$$\mathcal{H}\triangle\mathcal{H} := \left\{h(z) \oplus h'(z), h, h' \in \mathcal{H}\right\}$$

where $\oplus$ denotes the XOR operator, so that $h(z) \oplus h'(z)$ indicates that $h$ and $h'$ disagrees with each other. Accordingly, $\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}$ is a set of measurable subsets for $\forall\, h(z) \oplus h'(z) \in \mathcal{H}\triangle\mathcal{H}$. Then $d_{\mathcal{H}\triangle\mathcal{H}}(\cdot, \cdot)$ is defined as the *distribution divergence* induced by the symmetric difference hypothesis space (Blitzer et al., 2008):

$$d_{\mathcal{H}\triangle\mathcal{H}}(\mathcal{D}, \mathcal{D}') := 2 \sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}} |\Pr_{\mathcal{D}}(\mathcal{A}) - \Pr_{\mathcal{D}'}(\mathcal{A})|\}.$$

Specifically, let $\mathcal{D}, \mathcal{D}'$ be two arbitrary distributions on the input space $\mathcal{X}$, and let $\tilde{\mathcal{D}}, \tilde{\mathcal{D}}'$ be their induced images over $\mathcal{R}$. Then based on the definition of $d_{\mathcal{H}\triangle\mathcal{H}}(\cdot, \cdot)$, one can have:

$$\begin{aligned} d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}, \tilde{\mathcal{D}}') &= 2 \sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}} |\mathbb{E}_{x\sim\mathcal{D}}\left[\Pr(\mathcal{A}(\mathcal{R}(x)))\right] - \mathbb{E}_{x\sim\mathcal{D}'}\left[\Pr(\mathcal{A}(\mathcal{R}(x)))\right]| \\ &= 2 \sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}} |\mathbb{E}_{z\sim\tilde{\mathcal{D}}}\left[\Pr(\mathcal{A}(x))\right] - \mathbb{E}_{z\sim\tilde{\mathcal{D}}'}\left[\Pr(\mathcal{A}(z))\right]| \\ &= 2 \sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}} |\Pr_{\tilde{\mathcal{D}}}(\mathcal{A}) - \Pr_{\tilde{\mathcal{D}}'}(\mathcal{A})|\}. \end{aligned}$$

### 8.2. Derivations of Remark 1

**Remark.** *Let $p(y)$ be the prior distribution of labels, and $r(z|y) : \mathcal{Y} \to \mathcal{Z}$ be the conditional distribution derived from generator $G_{\boldsymbol{w}}$. Then regulating a user model $\boldsymbol{\theta}_k$ using samples from $r(z|y)$ can minimize the conditional KL-divergence between two distributions, derived from the user and from the generator, respectively:*

$$\max_{\boldsymbol{\theta}_k} \mathbb{E}_{y\sim p(y), z\sim r(z|y)}\left[\log p(y|z; \boldsymbol{\theta}_k)\right] \equiv \min_{\boldsymbol{\theta}_k} D_{\mathrm{KL}}[r(z|y)\|p(z|y; \boldsymbol{\theta}_k)],$$

*Proof.* Expanding the KL-divergence, we have

$$\begin{aligned}\because\ D_{\mathrm{KL}}[r(z|y)\|p(z|y;\boldsymbol{\theta}_k)] &\equiv \mathbb{E}_{y\sim p(y)}\left[\mathbb{E}_{z\sim r(z|y)}\left[\log\frac{r(z|y)}{p(z|y;\boldsymbol{\theta})}\right]\right]\\ &= \mathbb{E}_{y\sim p(y)}\mathbb{E}_{z\sim r(z|y)}\left[\log r(z|y)\right] - \mathbb{E}_{y\sim p(y)}\mathbb{E}_{z\sim r(z|y)}\left[\log p(z|y;\boldsymbol{\theta})\right]\\ &= \underbrace{-H(r(z|y))}_{\text{constant w.r.t }\boldsymbol{\theta}_k} - \mathbb{E}_{y\sim p(y)}\mathbb{E}_{z\sim r(z|y)}[\log p(z|y;\boldsymbol{\theta})].\end{aligned}$$

where $H(r(z|y))$ is constant w.r.t $\boldsymbol{\theta}_k$. Therefore when optimizing $\boldsymbol{\theta}_k$ we have:

$$\begin{aligned}&\min_{\boldsymbol{\theta}_k}\ D_{\mathrm{KL}}[r(z|y)\|p(z|y;\boldsymbol{\theta}_k)]\\ \equiv&\min_{\boldsymbol{\theta}_k}\ -\mathbb{E}_{y\sim p(y),z\sim r(z|y)}\left[\log p(z|y;\boldsymbol{\theta}_k)\right]\\ \equiv&\max_{\boldsymbol{\theta}_k}\mathbb{E}_{y\sim p(y)}\mathbb{E}_{z\sim r(z|y)}[\log\frac{p(y|z;\boldsymbol{\theta}_k)p(z)}{p(y)}]\\ \equiv&\max_{\boldsymbol{\theta}_k}\mathbb{E}_{y\sim p(y)}\mathbb{E}_{z\sim r(z|y)}[\log p(y|z;\boldsymbol{\theta}_k)+\log p(z)-\log p(y)]\\ \equiv&\max_{\boldsymbol{\theta}_k}\mathbb{E}_{y\sim p(y)}\mathbb{E}_{z\sim r(z|y)}[\log p(y|z;\boldsymbol{\theta}_k)].\end{aligned}$$

where $H(r(z|y))$ denotes the entropy of the probability distribution $r(z|y)$ which is *not* optimizable w.r.t $\boldsymbol{\theta}_k$, and $p(z|y;\boldsymbol{\theta}_k) := \frac{p(y|z;\boldsymbol{\theta}_k)p(z)}{p(y)}$ is defined as the probability that the input representation to the predictor is $z$ if it yields a label $y$. □

### 8.3. Derivations of Theorem 1

Before deriving Theorem 1, we first present an upper-bound for the generalization performance from prior art (Ben-David et al., 2007), which analyzes the role of a feature representation function in the context of *domain adaptation*:

**Lemma 1. Generalization Bounds for Domain Adaptation (Ben-David et al., 2007; Blitzer et al., 2008):**

*Let $\mathcal{T}_S$ and $\mathcal{T}_T$ be the source and target domains, whose data distributions are $\mathcal{D}_S$ and $\mathcal{D}_T$. Let $\mathcal{R}:\mathcal{X}\to\mathcal{Z}$ be a feature representation function, and $\tilde{\mathcal{D}}_S,\tilde{\mathcal{D}}_T$ be the induced images of $\mathcal{D}_S$ and $\mathcal{D}_T$ over $\mathcal{R}$, respectively. Let $\mathcal{H}$ be a set of hypothesis with VC-dimension $d$. Then with probability at least $1-\delta$, $\forall\ h\in\mathcal{H}$:*

$$\mathcal{L}_{\mathcal{T}_T}(h)\le\hat{\mathcal{L}}_{\mathcal{T}_S}(h)+\sqrt{\frac{4}{m}\left(d\log\frac{2em}{d}+\log\frac{4}{\delta}\right)}+d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_S,\tilde{\mathcal{D}}_T)+\lambda, \tag{8}$$

where $e$ is the base of the natural logarithm, $\hat{\mathcal{L}}_{\mathcal{T}_S}$(h) is the empirical risk of the source domain given $m$ observable samples, and $\lambda=\min_{h\in\mathcal{H}}\left(\mathcal{L}_{\mathcal{T}_T}(h)+\mathcal{L}_{\mathcal{T}_S}(h)\right)$ is the optimal risk on the two domains.

One insight from Lemma 1 is that a good representation function plays a tradeoff between minimizing the empirical risk ($\hat{\mathcal{L}}_{\mathcal{T}_S}(h)$) and the induced distributional discrepancy ( $d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_S,\tilde{\mathcal{D}}_T)$). Based on Lemma 1, one can establish Theorem 1 as the following:

**Theorem.** (**Generalization Bounds for FL**) *Consider an FL system with $K$ users. Let $\mathcal{T}_k=\langle\mathcal{D}_k,c^*\rangle$ and $\mathcal{T}=\langle\mathcal{D},c^*\rangle$ be the $k$-th local domain and the global domain, respectively. Let $\mathcal{R}:\mathcal{X}\to\mathcal{Z}$ be a feature extraction function that is simultaneously shared among users. Let $h_k$ denote the hypothesis learned on domain $\mathcal{T}_k$, and $h=\frac{1}{K}\sum_{k=1}^{K}h_k$ be the global ensemble of user predictors. Then with probability at least $1-\delta$:*

$$\mathcal{L}_{\mathcal{T}}(h)\le\frac{1}{K}\sum_{k\in[K]}\hat{\mathcal{L}}_{\mathcal{T}_k}(h_k)+\frac{1}{K}\sum_{k\in[K]}\left(d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}})+\lambda_k\right)+\sqrt{\frac{4}{m}\left(d\log\frac{2em}{d}+\log\frac{4K}{\delta}\right)},$$

*where $\hat{\mathcal{L}}_{\mathcal{T}_k}(h_k)$ is the empirical risk of $h_k$, $\lambda_k:=\min_h(\mathcal{L}_{\mathcal{T}_k}(h)+\mathcal{L}_{\mathcal{T}}(h))$ denotes an oracle performance on $\mathcal{T}_k$ and $\mathcal{T}$, and $\tilde{\mathcal{D}}_k$ and $\tilde{\mathcal{D}}$ is the **induced** image of $\mathcal{D}_k$ and $\mathcal{D}$ from $\mathcal{R}$, respectively, s.t. $\mathbb{E}_{z\sim\tilde{\mathcal{D}}_k}[\mathcal{B}(z)]=\mathbb{E}_{x\sim\mathcal{D}_k}[\mathcal{B}(\mathcal{R}(x))]$ given a probability event $\mathcal{B}$, and so for $\tilde{\mathcal{D}}$.*

*Proof.* By treating each one of the local domains $k\in[K]$ as the *source* and the global domain as the *target*, one can have that, $\forall\ \delta>0$, with probability $1-\frac{\delta}{K}$:

$$\mathcal{L}_{\mathcal{T}}(h_k)\le\hat{\mathcal{L}}_{\mathcal{T}_k}(h_k)+d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}})+\lambda_k+\sqrt{\frac{4}{m}\left(d\log\frac{2em}{d}+\log\frac{4K}{\delta}\right)}.$$

Also, due to the convexity of risk function and Jesen inequality, one can have:

$$\mathcal{L}_{\mathcal{T}}(h) \equiv \mathcal{L}_{\mathcal{T}}\left(\frac{1}{K}\sum_{k\in[K]} h_k\right) \leq \frac{1}{K}\sum_{k\in[K]} \mathcal{L}_{\mathcal{T}}(h_k).$$

Therefore,

$$\begin{aligned}
&\Pr\left[\mathcal{L}_{\mathcal{T}}(h) > \frac{1}{K}\sum_{k\in[K]}\left(\hat{\mathcal{L}}_{\mathcal{T}_k}(h_k) + \sum_{k\in[K]}(d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}}) + \lambda_k) + \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4K}{\delta}\right)}\right)\right] \\
\leq &\Pr\left[\frac{1}{K}\sum_{k\in[K]}\mathcal{L}_{\mathcal{T}}(h_k) > \frac{1}{K}\sum_{k\in[K]}\left(\hat{\mathcal{L}}_{\mathcal{T}_k}(h_k) + \sum_{k\in[K]}(d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}}) + \lambda_k) + \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4K}{\delta}\right)}\right)\right] \\
\leq &\Pr\left[\bigvee_{k\in[K]}\mathcal{L}_{\mathcal{T}}(h_k) > \hat{\mathcal{L}}_{\mathcal{T}_k}(h_k) + d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}}) + \lambda_k + \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4K}{\delta}\right)}\right] \\
\leq &\sum_{k\in[K]}\frac{\delta}{K} = \delta.
\end{aligned}$$

□

Theorem 1 shows that the performance of the aggregated hypothesis is upper-bounded by: 1) the local performance of each user hypothesis ($\hat{\mathcal{L}}_{\mathcal{T}_k}(h_k)$), 2) the dissimilarity between the global and local distributions over the feature space ($d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}})$), 3) the oracle performance ($\lambda_k$), and 4) the numerical constraints regarding the number of empirical samples $m$ and the VC-dimension $d$.

### 8.4. Derivations of Corollary 1

**Corollary.** *Let $\mathcal{T}$, $\mathcal{T}_k$, $\mathcal{R}$ defined as in Theorem 1. $\mathcal{D}_{\mathrm{A}}$ denotes an **augmented** data distribution, and $\mathcal{D}'_k = \frac{1}{2}(\mathcal{D}_k + \mathcal{D}_{\mathrm{A}})$ is a **mixture** of distributions. Accordingly, $\tilde{\mathcal{D}}_{\mathrm{A}}$ and $\tilde{\mathcal{D}}'_k$ denote the **induced** image of $\mathcal{D}_{\mathrm{A}}$ and $\mathcal{D}'_k$ over $\mathcal{R}$, respectively. Let $\hat{\mathcal{D}}'_k = \hat{\mathcal{D}}_k \cup \hat{\mathcal{D}}_{\mathrm{A}}$ be an empirical dataset of $\mathcal{D}'_k$, with $|\hat{\mathcal{D}}_k|$=m, $|\hat{\mathcal{D}}'_k| = |\hat{\mathcal{D}}_k| + |\hat{\mathcal{D}}_{\mathrm{A}}| = m'$ . Assume the discrepancy between $\tilde{\mathcal{D}}_{\mathrm{A}}$ and $\tilde{\mathcal{D}}$ is bounded, s.t $\exists\,\epsilon > 0, d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A,\tilde{\mathcal{D}}) \leq \epsilon$, then with probability $1-\delta$:*

$$\mathcal{L}_{\mathcal{T}}(h) \leq \frac{1}{K}\sum_k \mathcal{L}_{\mathcal{T}'_k}(h_k) + \frac{1}{K}\sum_k (d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k,\tilde{\mathcal{D}})) + \frac{1}{K}\sum_k \lambda'_k + \sqrt{\frac{4}{m'}\left(d\log\frac{2em'}{d} + \log\frac{4K}{\delta}\right)}, \tag{9}$$

*where $\mathcal{T}'_k = \{\mathcal{D}'_k, c^*\}$ is the updated local domain, $\lambda'_k = \min_h(\mathcal{L}_{\mathcal{T}'_k}(h) + \mathcal{L}_{\mathcal{T}}(h))$ denotes the oracle performance, and $d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k,\tilde{\mathcal{D}}) \leq d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}})$ when $\epsilon$ is small.*

*Proof.* Equation 9 can be directly derived by Theorem 1. We now focus on analyzing the relation between $d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}})$ and $d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k,\tilde{\mathcal{D}})$, which is the data dissimilarity ***before*** and ***after*** data augmentation using samples from distribution $\mathcal{D}_A$, respectively.

Based on the definition of $d_{\mathcal{H}\triangle\mathcal{H}}(\cdot,\cdot)$, one can derive that:

$$\begin{aligned}
&d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k,\tilde{\mathcal{D}}) \\
=&2\sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}}\left|\mathbb{E}_{z\sim\tilde{\mathcal{D}}'_k}[\Pr(\mathcal{A}(z))] - \mathbb{E}_{z\sim\tilde{\mathcal{D}}}[\Pr(\mathcal{A}(z))]\right| \\
=&2\sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}}\left|\mathbb{E}_{z\sim\frac{1}{2}(\tilde{\mathcal{D}}_k+\tilde{\mathcal{D}}_A)}[\Pr(\mathcal{A}(z))] - \mathbb{E}_{z\sim\tilde{\mathcal{D}}}[\Pr(\mathcal{A}(z))]\right| \\
=&2\sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}}\left|\frac{1}{2}\mathbb{E}_{z\sim\tilde{\mathcal{D}}_K}[\Pr(\mathcal{A}(z))] + \frac{1}{2}\mathbb{E}_{z\sim\tilde{\mathcal{D}}_A}[\Pr(\mathcal{A}(z))] - \mathbb{E}_{z\sim\tilde{\mathcal{D}}}[\Pr(\mathcal{A}(z))]\right| \\
\leq&\sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}}\left|\mathbb{E}_{z\sim\tilde{\mathcal{D}}_K}[\Pr(\mathcal{A}(z))] - \mathbb{E}_{z\sim\tilde{\mathcal{D}}}[\Pr(\mathcal{A}(z))]\right| + \sup_{\mathcal{A}\in\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}}\left|\mathbb{E}_{z\sim\tilde{\mathcal{D}}_A}[\Pr(\mathcal{A}(z))] - \mathbb{E}_{z\sim\tilde{\mathcal{D}}}[\Pr(\mathcal{A}(z))]\right| \\
=&\frac{1}{2}d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k,\tilde{\mathcal{D}}) + \frac{1}{2}d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A,\tilde{\mathcal{D}}).
\end{aligned}$$

It is clear that $\frac{1}{2}d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A,\tilde{\mathcal{D}})$, which is bounded by $\epsilon$, affects the dissimilarity between the *induced* image of local and the global distribution, therefore plays a key role in upper-bounding the global performance ($\mathcal{L}_{\mathcal{T}}(h)$ in Equation 9). Next, we discuss different

scenarios when FL can benefit from such augmented data, and when the quality of augmented distribution $\mathcal{D}_A$ can limit the generalization performance of the aggregated model.

**$\mathcal{D}_A$ can benefit local users when $\epsilon$ is small:** To see this, one can assume that:

$$d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A, \tilde{\mathcal{D}}) = \epsilon \leq \min_k d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}}),$$

of which the intuition is that, after feature mapping, the discrepancy between the augmented distribution and the global distribution is smaller than the discrepancy between an individual user and the global. Based on this assumption, one can conclude that $\forall\ \mathcal{T}_k \in \boldsymbol{\mathcal{T}}$:

$$\begin{aligned} d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k, \tilde{\mathcal{D}}) =& \frac{1}{2} d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}}) + \frac{1}{2} d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A, \tilde{\mathcal{D}}) \\ \leq& \frac{1}{2} d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}}) + \min_j d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_j, \tilde{\mathcal{D}}) \\ \leq& d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}}), \end{aligned}$$

Therefore, a small $d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A, \tilde{\mathcal{D}})$ benefits local users w.r.t their generalization performance, by both reducing the data discrepancy and enriching the empirical samples, in that:

$$\mathcal{L}_{\mathcal{T}}(h_k) \leq \mathcal{L}_{\mathcal{T}'_k}(h_k) + \lambda'_k + \underbrace{\leq d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k, \tilde{\mathcal{D}})}_{\leq d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}})} + \underbrace{\sqrt{\frac{4}{m'}\left(d\log\frac{2em'}{d} + \log\frac{4}{\delta}\right)}}_{\leq\sqrt{\frac{4}{m}(d\log\frac{2em}{d}+\log\frac{4}{\delta})}} \qquad \text{( Derived from Lemma 1).}$$

**$\mathcal{D}_A$ has positive effects on the generalization performance when $\epsilon$ is moderate:** Instead, one might as well assume that

$$d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_A, \tilde{\mathcal{D}}) = \epsilon \leq \frac{1}{K}\sum_{k=1}^{K} d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}}),$$

which implies that, after feature mapping over $\mathcal{R}$, the dissimilarity between $\mathcal{D}_A$ and the global distribution $\mathcal{D}$ is at least as small as the *average* dissimilarity between local users and the global. Based on this assumption, one can derive that:

$$\sum_k d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}'_k, \tilde{\mathcal{D}}) \leq \sum_k d_{\mathcal{H}\triangle\mathcal{H}}(\tilde{\mathcal{D}}_k, \tilde{\mathcal{D}}),\ \sqrt{\frac{4}{m'}\left(d\log\frac{2em'}{d} + \log\frac{4}{\delta}\right)} \leq \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4}{\delta}\right)},$$

which can still contribute to a tighter upper-bound for the global performance in Equation 9, compared with not using the augmented data.

Conversely, when $\epsilon$ is over-large, which implies that $\mathcal{D}_A$ is not relevant to the original FL task, it may have negative impacts on the generalization performance. □

# 9. Extended Experiments

We first discuss some practical considerations for implementing our algorithm:

- **Weighting user models:** User models vary in their ability to predict certain labels over others due to their statistical heterogeneity. Therefore, we use the number of training labels available to users to summarize a weight matrix $\boldsymbol{\Lambda} = \{\lambda_k^c | c \in \mathcal{Y}, k \in \{1, 2, \cdots, K\}\}$, *s.t.* $\forall c, i, j,\ \frac{\lambda_i^c}{\lambda_j^c} = \frac{n_i^c}{n_j^c}$ indicates the ratio of training samples for label $c$ between two users $i$ and $j$, and $\sum_k \lambda_k^c = 1\ \forall c \in \mathcal{Y}$. We then apply this weight matrix to adjust the generator objective as the following:

$$\min_{\boldsymbol{w}} J(\boldsymbol{w}) := \mathbb{E}_{y\sim\hat{p}(y)}\mathbb{E}_{z\sim G_{\boldsymbol{w}}(z|y)}\left[\lambda_k^y l\left(\sigma\left(\frac{1}{K}\sum\nolimits_{k=1}^{K} g(z;\boldsymbol{\theta}_k^p)\right), y\right)\right].$$

  We found that this weighted objective can further mitigate the impact of negative ensemble, especially when a teacher model is too weak to predict certain labels due to lacking training samples of that category.

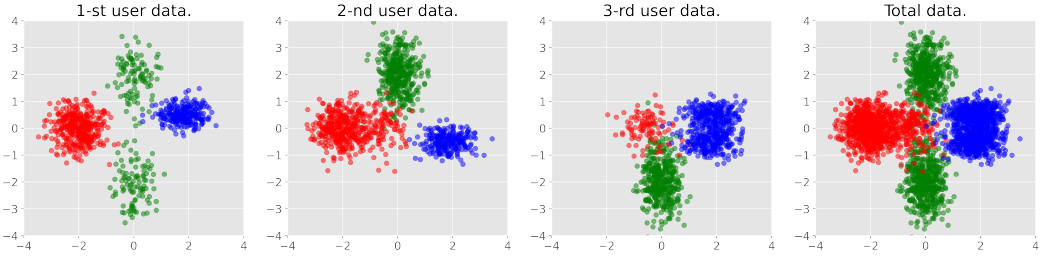
- **Stochastic generative learning:** Built upon prior arts on generative learning (Kingma & Welling, 2014), we use an auxiliary noise vector with dimension $d_n$ to infer the desirable feature representation for a given label $y$, *s.t.* $z \sim G_{\boldsymbol{w}}(\cdot|y) \equiv G_{\boldsymbol{w}}(y, \epsilon|\epsilon \sim \mathcal{N}(0, I))$. To further increase the diversity of the generator output, we also leverage the idea of *diversity loss* from prior work (Mao et al., 2019) to train the generator model.

## 9.1. Prototype Results

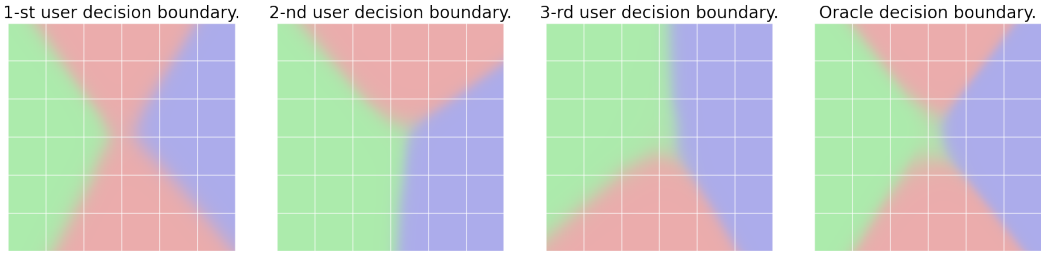
We adopt an one-round FL setting for the prototype experiment, for which the dataset distributions of local users, as well as their model decision boundaries *before* and *after* knowledge distillation, are illustrated in Figure 9. Accuracy of user models on the global dataset is also summarized in Table 5, from which one can observe that the generalization performance of user models have been notably improved by the distilled knowledge.

| | User 1 | User 2 | User 3 | Oracle |
|---|---|---|---|---|
| Before | 97.1 | 81.3 | 81.2 | 98.4 |
| After | 98.6 | 98.3 | 98.2 | |

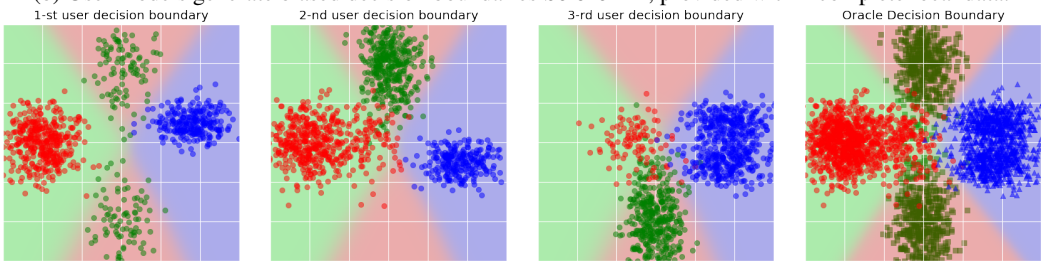
*Table 5.* Accuracy (%) before and after KD.




(a) Local user data distribution and total data distribution.




(b) User models generate biased decision boundaries **before** KD, provided with incomplete local data.




(c) Decision Boundaries of user models are improved **after** KD.

*Figure 9.* Knowledge distillation process for the prototype experiment.

## 9.2. Experimental Setup

We provide the network architecture for the generator and the classifier in Table 6 and Table 7. For the generator $G_{\boldsymbol{w}}$, we adopt a two-MLP layer network. It takes a noise vector $\epsilon$ and an one-hot label vector $y$ as the input, which, after a hidden layer with dimension $d_h$, outputs a feature representation with dimension $d$. For the classifier, we adopt a network architecture with a CNN module followed by a MLP module. Hyperparameter settings for the experiments are provided in Table 8.

| Dataset | Hyperparameter | Value |
|---|---|---|
| CELEBA | $d_n, d_h, d$ | 32, 128, 32 |
| MNIST& EMNIST | $d_n, d_h, d$ | 32, 256, 32 |

*Table 6.* Network architecture for the generator $G_{\boldsymbol{w}}$.

| Dataset | Hyperparameter | Value |
|---|---|---|
| CELEBA | CNN Module | [16, M, 32, M, 64] |
| | MLP Module | [784, 32] |
| MNIST & EMNIST | CNN Module | [6, 16] |
| | MLP Module | [784, 32] |

*Table 7.* Network architecture for the classification model.

| | Hyperparameter | Value |
|---|---|---|
| Shared Parameters | Learning rate | 0.01 |
| | Optimizer | sgd |
| | Local update steps ($T$) | 20 |
| | Batch size ($B$) | 32 |
| | Communication rounds | 200 |
| | # of total users | 20 |
| | # of active users | 10 |
| FEDDFUSION | Ensemble Optimizer | adam |
| | Generator learning rate | $10^{-4}$ |
| | Ensemble batch size | 128 |
| FEDGEN | Generator Optimizer | adam |
| | Generator learning rate | $10^{-4}$ |
| | Generator inference size | 128 |
| | User distillation batch size | 32 |
| FEDDISTILL& FEDDISTILL$^+$ | Distillation coefficient | 0.1 |
| FEDPROX | Proximal coefficient | 0.1 |

*Table 8.* We use the above configurations for experiments unless mentioned otherwise.

### 9.3. FEDGEN with Partial Parameter Sharing

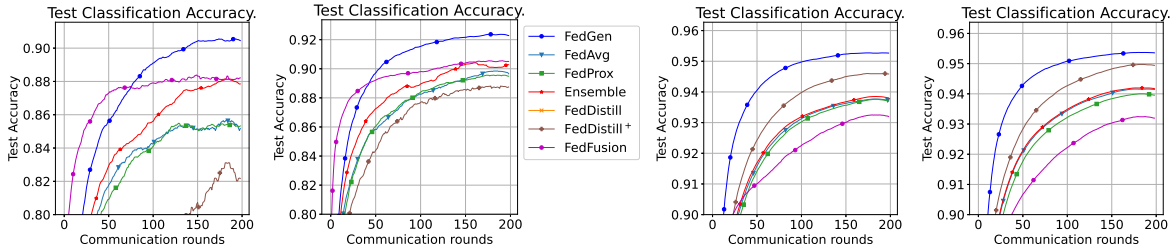
Algorithm 2 summarizes an variant approach of FEDGEN for a specific FL setting, where only the last prediction layer is shared among users while keeping the feature extraction layers localized.

**Algorithm 2** FEDGEN with Partial Parameter Sharing

**Require:** Tasks $\mathcal{T}_k, k \in \{1, \cdots, K\}$;
Global predictor $\boldsymbol{\theta}^p$, local parameters $\{\boldsymbol{\theta}_k = [\boldsymbol{\theta}_k^f; \boldsymbol{\theta}_k^p]\}_{k=1}^K$;
Generator parameter $\boldsymbol{w}$; $\hat{p}(y)$ uniformly initialized;
Learning rate $\alpha$, $\beta$, local steps $T$, batch size $B$, local label counter $c_k$.
**repeat**
  Server selects active users $\mathcal{A}$ uniformly at random, then broadcast $\boldsymbol{w}, \boldsymbol{\theta}^p, \hat{p}(y)$ to $\mathcal{A}$.
  **for** all user $k \in \mathcal{A}$ in parallel **do**
    $\boldsymbol{\theta}_k^p \leftarrow \boldsymbol{\theta}^p$,
    **for** $t = 1, \ldots, T$ **do**
      $\{x_i, y_i\}_{i=1}^B \sim \mathcal{T}_k$, $\{\hat{z}_i \sim G_{\boldsymbol{w}}(\cdot|\hat{y}_i), \hat{y}_i \sim \hat{p}(y)\}_{i=1}^B$.
      Update label counter $c_k$.
    $\boldsymbol{\theta}_k \leftarrow \boldsymbol{\theta}_k - \beta \nabla_{\boldsymbol{\theta}_k} J(\boldsymbol{\theta}_k)$.
    **end for**
    User sends $\boldsymbol{\theta}_k^p$, $c_k$ back to server.
  **end for**
  Server updates $\boldsymbol{\theta}^p \leftarrow \frac{1}{|\mathcal{A}|} \sum_{k \in \mathcal{A}} \boldsymbol{\theta}_k^p$, and $\hat{p}(y)$ based on $\{c_k\}_{k \in \mathcal{A}}$.
  $\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \nabla_{\boldsymbol{w}} J(\boldsymbol{w})$.
**until** training stop

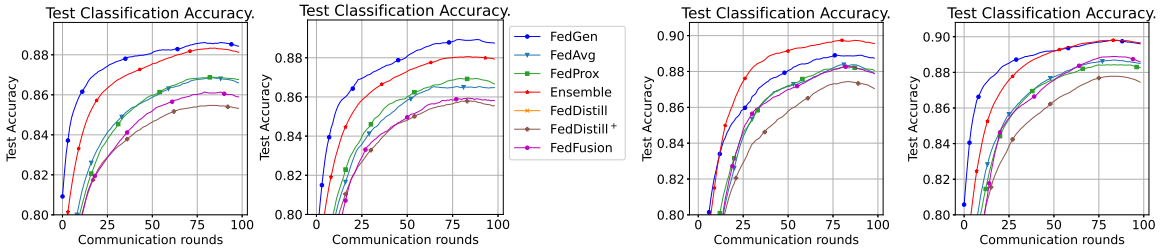
### 9.4. Extended Experimental Results

We elaborate the learning curves trained on the MNIST, CELEBA, and EMNIST dataset in Figure 10, Figure 11, and Figure 12, respectively, with their performance summarized in Table 9.



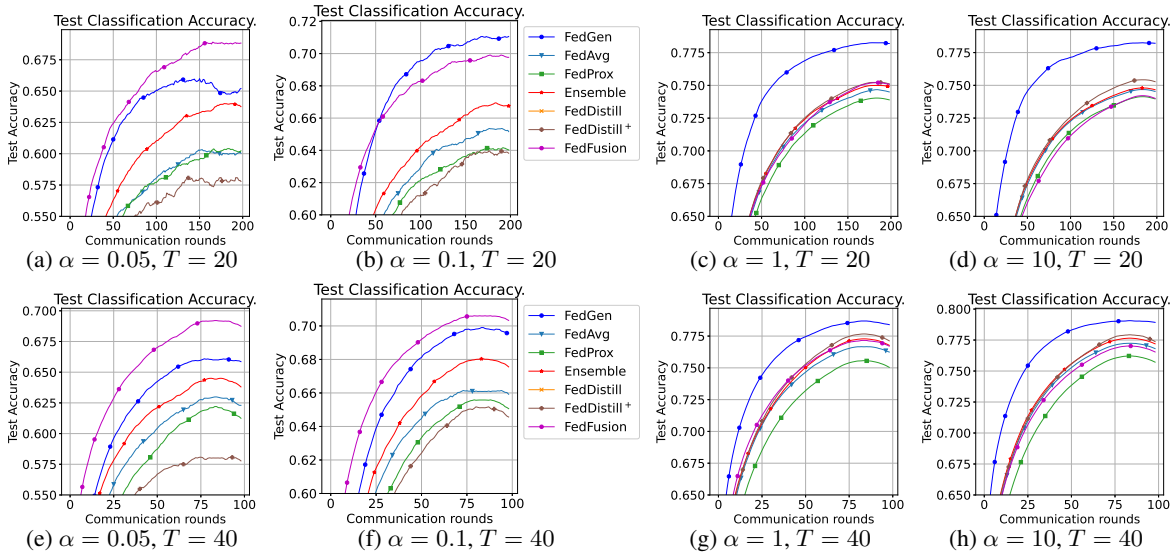

(a) $\alpha = 0.05$ (b) $\alpha = 0.1$ (c) $\alpha = 1$ (d) $\alpha = 10$

*Figure 10.* Performance curves on MNIST dataset, where a smaller $\alpha$ denotes larger data heterogeneity.




(a) CELEBA, $r = 9/10$. (b) CELEBA, $r = 5/10$. (c) CELEBA, $r = 5/25$. (d) CELEBA, $r = 10/25$.

*Figure 11.* Performance curves on CELEBA dataset.




(a) $\alpha = 0.05, T = 20$ (b) $\alpha = 0.1, T = 20$ (c) $\alpha = 1, T = 20$ (d) $\alpha = 10, T = 20$

(e) $\alpha = 0.05, T = 40$ (f) $\alpha = 0.1, T = 40$ (g) $\alpha = 1, T = 40$ (h) $\alpha = 10, T = 40$

*Figure 12.* Performance curves on EMNIST dataset, under different data heterogeneity and communication frequencies.

| Top-1 Test Accuracy. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Dataset | Setting | FEDAVG | FEDPROX | FEDENSEMBLE | FEDDISTILL | FEDDISTILL$^+$ | FEDDFUSION | FEDGEN |
| MNIST | $\alpha$ = 0.05 | 87.70±2.07 | 87.49±2.05 | 88.85±0.68 | 70.56±1.24 | 86.70±2.27 | 90.02±0.96 | **91.30±0.74** |
| | $\alpha$ = 0.1 | 90.16±0.59 | 90.10±0.39 | 90.78±0.39 | 64.11± 1.36 | 90.28±0.89 | 91.11±0.43 | **93.03±0.32** |
| | $\alpha$ = 1 | 93.84±0.25 | 93.83 ± 0.29 | 93.91±0.28 | 79.88±0.66 | 94.73±0.15 | 93.37±0.40 | **95.52±0.07** |
| | $\alpha$ = 10 | 94.23±0.13 | 94.06±0.10 | 94.25±0.11 | 89.21±0.26 | 95.04±0.21 | 93.36±0.45 | **95.79±0.10** |
| CELEBA | $r = 5/10$ | 87.48±0.39 | 87.67±0.39 | 88.48±0.23 | 76.68±1.23 | 86.37±0.41 | 87.01±1.00 | **89.70±0.32** |
| | $r = 5/25$ | 89.13±0.25 | 88.84±0.19 | **90.22±0.31** | 74.99±1.57 | 88.05± 0.43 | 88.93±0.79 | 89.62±0.34 |
| | $r = 10/25$ | 89.12±0.20 | 89.01±0.33 | 90.08±0.24 | 75.88±1.17 | 88.14±0.37 | 89.25±0.56 | **90.29±0.47** |
| EMNIST, $T$=20 | $\alpha$ = 0.05 | 62.25±2.82 | 61.93±2.31 | 64.99±0.35 | 60.49±1.27 | 61.56±2.15 | **70.40±0.79** | 68.53±1.17 |
| | $\alpha$ = 0.1 | 66.21±2.43 | 65.29±2.94 | 67.53±1.19 | 50.32±1.39 | 66.06±3.18 | 70.94±0.76 | **72.15±0.21** |
| | $\alpha$ = 1 | 74.83±0.99 | 74.12±0.88 | 75.12±1.07 | 46.19±0.70 | 75.41±1.05 | 75.43±0.37 | **78.48±1.04** |
| | $\alpha$ = 10 | 74.83± 0.69 | 74.24±0.81 | 74.90±0.80 | 54.77±0.33 | 75.55 ±0.94 | 74.36±0.40 | **78.43±0.74** |
| EMNIST, $T$=40 | $\alpha$ = 0.05 | 64.51±1.13 | 63.60±0.69 | 65.74±0.45 | 60.73±1.62 | 60.73±1.06 | **70.46±1.16** | 67.64 ±0.75 |
| | $\alpha$ = 0.1 | 67.71±1.31 | 66.79±0.77 | 68.96±0.66 | 49.54±1.18 | 67.01±0.38 | **71.55±0.43** | 70.90 ±0.49 |
| | $\alpha$ = 1 | 77.02±1.09 | 75.93 ±0.95 | 77.68±0.98 | 46.72±0.73 | 78.12±0.90 | 77.58±0.37 | **78.92± 0.73** |
| | $\alpha$ = 10 | 77.52±0.66 | 76.54±0.71 | 77.92±0.62 | 54.85±0.44 | 78.37±0.76 | 77.31±0.45 | **79.29±0.53** |

*Table 9.* Performance overview under different data heterogeneity settings. For MNIST and EMNIST, user data follows the Dirichlet distribution with hyperparameter $\alpha$, with a **smaller** $\alpha$ indicating higher heterogeneity. For CELEBA, $r$ denotes the ratio between active users and total users. $T$ denotes the local training steps (communication delay). All above experiments use batch size $B$=32.