Overcoming Data and Model heterogeneities in Decentralized Federated Learning via Synthetic Anchors

Chun-Yin Huang [1 2] **Kartik Srinivas** [1 3] **Xin Zhang** [4] **Xiaoxiao Li** [1 2]

Abstract

Conventional Federated Learning (FL) involves collaborative training of a global model while maintaining user data privacy. One of its branches, decentralized FL, is a serverless network that allows clients to own and optimize different local models separately, which results in saving management and communication resources. Despite the promising advancements in decentralized FL, it may reduce model generalizability due to lacking a global model. In this scenario, managing data and model heterogeneity among clients becomes a crucial problem, which poses a unique challenge that must be overcome: *How can every client's local model learn generalizable representation in a decentralized manner?* To address this challenge, we propose a novel **De**centralized FL technique by introducing **S**ynthetic **A**nchors, dubbed as DeSA. Based on the theory of domain adaptation and Knowledge Distillation (KD), we theoretically and empirically show that synthesizing global anchors based on raw data distribution facilitates mutual knowledge transfer. We further design two effective regularization terms for local training: *1) REG loss* that regularizes the distribution of the client's latent embedding with the anchors and *2) KD loss* that enables clients to learn from others. Through extensive experiments on diverse client data distributions, we showcase the effectiveness of DeSA in enhancing both inter- and intra-domain accuracy of each client. The implementation of DeSA can be found at: https://github.com/ubc-tea/DESA

[1]Department of Electrical and Computer Engineering, The University of British Columbia, Canada [2]Vector Institute, Canada [3]Indian Institute of Technology Hyderabad, India (Work was done during internship at UBC) [4]Meta, U.S.A.. Correspondence to: Xiaoxiao Li <xiaoxiao.li@ece.ubc.ca>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

1. Introduction

Federated learning (FL) has emerged as an important paradigm to perform machine learning from multi-source data in a distributed manner. Conventional FL techniques leverage a large number of clients to process a global model learning, which is coordinated by a central server. However, there arises concerns on increased vulnerability of system failures and trustworthiness concerns for the central server design in the conventional FL. An emerging paradigm, called **decentralized FL**, is featured by its serverless setting to address the issues. Recent work has shown decentralized FL framework can provide more flexibility and solubility (Beltrán et al., 2023; Yuan et al., 2023b), where they relax the use of central server for model aggregation. However, this could deflect the generalization capability of each client model. Although most of the works in decentralized FL focus on model personalization (Huang et al., 2022), we consider it crucial for decentralized FL to be generalizable since local training data may not align with local testing data in practice.

Client heterogeneity is a common phenomenon in FL that can deteriorate model generalizability. On one hand, **data-heterogeneity** relaxes the assumption that the data across all the client are independent and identically distributed (i.i.d.). To solve the problem, a plethora of methods have been proposed. However, most of them assumes that the model architectures are invariant across clients (Li et al., 2020b;a; 2021b; Karimireddy et al., 2020; Tang et al., 2022). On the other hand, many practical FL applications (*e.g.*, Internet-of-Things and mobile device system) face **model-heterogeneity**, where clients have devices with different computation capabilities and memory constraints. In conventional FL, strategies have been proposed to leverage knowledge transferring to address the model heterogeneity issue, *e.g.*, server collects labeled data with the similar distribution as the client data or clients transmit models (Lin et al., 2020; Zhu et al., 2021). However, these operations usually require a server to coordinate the knowledge distillation and assume global data is available (Li & Wang, 2019; Lin et al., 2020; Tan et al., 2022). Thus, they are not applicable to decentralized FL and may not fit into real-world scenarios. Recently, there are works proposing to perform test-time

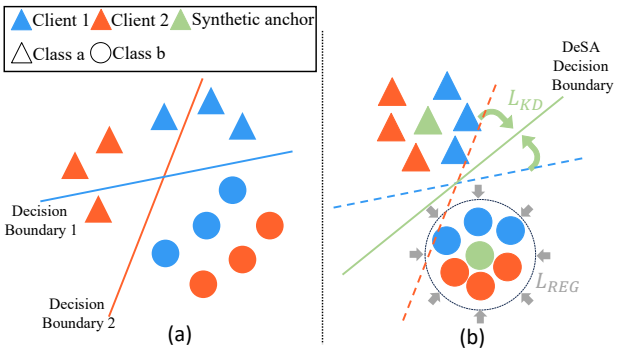


Figure 1: The decision boundary before (a) and after (b) applying our proposed $\mathcal{L}_{\text{REG}}$ (Eq. 4) and $\mathcal{L}_{\text{KD}}$ (Eq. 6) using our synthetic anchor data. $\mathcal{L}_{\text{REG}}$ aims to group the raw feature towards synthetic anchor feature, and $\mathcal{L}_{\text{KD}}$ twists the local decision boundary towards the generalized decision boundary.

adaptation for out-of-federation clients (Jiang et al., 2023), while this paper focuses on the solution during FL training time.

We can see that both heterogeneous FL and decentralized FL leave the gray space of the following practical research question: *How can **every** client model perform well on other client domains, in a completely decentralized and heterogeneous FL setup?* Such a problem is referred as *decentralized federated mutual learning*, which is further detailed in Section 2.2. To the best of our knowledge, we are the first to address both data and model heterogeneity issues under serverless decentralized FL setting (see the comparison with related work in Table 1).

In this work, we tackle the research question by performing local heterogeneity harmonized training and knowledge distillation. In particular, we synthesize a lightweight synthetic data generation process via distribution matching (Zhao & Bilen, 2023), and the synthetic data are exchangeable across clients to augment local datasets. We theoretically and empirically show that the synthetic data can serve as **anchor points** to improves FL for two purposes: 1) reducing the domain-gap between the distributions in the latent feature space; and 2) enabling global knowledge distillation for mutual learning. The effect of the two losses are visualized in Figure 1. In summary, we tackle a realistic and challenging setting in decentralized FL, where both data and model heterogeneities exist, without acquiring publicly available global real data. Our contributions are listed as follows:

- To circumvent the heterogeneity on data and model, we propose an innovative algorithm named Decentralized Federated Learning with Synthetic Anchors (DeSA) that utilizes only a small number of synthetic data.
- We theoretically and empirically show that the strategic design of synthetic anchor data and our novel FL loss function effectively boost local model generalization in diverse data scenarios.

Table 1: Comparison of the settings with other related heterogeneous FL and decentralized FL methods.

Methods	Data Heterogeneity	Model Heterogeneity	Serverless	No Public Data
VHL (Tang et al., 2022)[a]	✓	✗	✗	✓
FedGen (Zhu et al., 2021)	✓	✗	✗	✓
FedHe (Chan & Ngai, 2021)	✗	✓	✓	✓
FedDF (Lin et al., 2020)	✓	✓	✗	✗
FCCL (Huang et al., 2022)	✓	✓	✗	✗
FedProto (Tan et al., 2022)	✓	✓	✗	✓
FedFTG (Zhang et al., 2022b)	✓	✓	✗	✓
DENSE (Zhang et al., 2022a)	✓	✓	✗	✓
DeSA (ours)	✓	✓	✓	✓

[a] VHL has a single global model, trained using mutual information from all clients. Therefore we reference it under Mutual Learning.

- We conduct extensive experiments prove DeSA's effectiveness, surpassing existing decentralized FL algorithms. It excels in inter- and intra-client performance across diverse tasks.

2. Preliminaries

2.1. Conventional Federated Learning

Conventional FL aims to learn a **single generalized global model** that performs optimally on all the clients' data domains. Mathematically, the learning problem can be formulated as

$$M^* = \arg\min_{M \in \mathcal{M}} \sum_{i=1}^{N} \mathbb{E}_{\mathbf{x},y \sim P_i}[\mathcal{L}(M(\mathbf{x}), y)] \qquad (1)$$

where M^* is the optimized global model from the shared model space $\mathcal{M}$, P_i is the local data distribution at the ith client, $\mathcal{L}$ is the loss function and $(\mathbf{x}, y)$ is the feature-label pair. Inspired by the pioneering work of FedAvg (McMahan et al., 2017), a plethora of methods have tried to fill in the performance gap of FedAvg on data-heterogeneous scenario, which can be categorized in two main orthogonal directions: *Direction 1* aims to minimize the difference between the local and global model parameters to improve convergence (Li et al., 2020a; Karimireddy et al., 2020; Wang et al., 2020). *Direction 2* enforces consistency in local embedded features using anchors and regularization loss (Tang et al., 2022; Zhou et al., 2022; Ye et al., 2022). This work follows the second research direction and aim to leverage anchor points to handle data heterogeneity. We also tackle the more challenging problem of domain shift, unlike other methods that only assume a label-shift amongst the client-data distributions.

2.2. Decentralized FL and Mutual Learning

Standard decentralized FL aims to solve the similar generalization objective as conventional FL (*i.e.*, Eq. 1), only, without a central server to do so (Gao et al., 2022), and the objective applies to each local models M_i. Here, we focus on learning from each other under heterogeneous models

and data distributions. This brings in an essential line of works, known as *mutual learning*, where clients learn from each other to obtain the essential generalizability for their models. For example, during quarantine for pandemic, hospitals want to collaboratively train a model for classifying the virus. It is desired the models are generalizable to virus variants from different area, so that after quarantine the local models are still effective for incoming tourists.

Although mutual learning with heterogeneous data and models has been studied recently, most of them assume the existence of public real data (Lin et al., 2020; Huang et al., 2022; Gong et al., 2022) or a central server to coordinate the generation of synthetic data from the local client data (Zhang et al., 2022a; Zhu et al., 2021; Zhang et al., 2022b). Another line of works rely on a server to aggregate locally generated logits or prototypes, and use it as local training guidance (Jeong et al., 2018; Chan & Ngai, 2021; Tan et al., 2022). In addition, more recent works have suggested that each clients train two models, a larger model for local training and a smaller model for mutual information exchange (Wu et al., 2022; Shen et al., 2023). However, none of the above methods simultaneously address both non-iid data and heterogeneous models under serverless and data-free setting. In this work, we explore mutual learning to optimize both local (intra-client) and global (inter-client) dataset accuracy (see the detailed setup in Sec. 3.1). We list the comparison with other methods in Table 1 and more detailed related works in Appendix G.

3. Method

3.1. Notation and Problem Setup

Suppose there are N clients with ith client denoted as C_i. Let's represent the **private datasets** on C_i as $D_i = \{\mathbf{x}, y\}$, where $\mathbf{x}$ is the feature and $y \in \{1, \cdots, K\}$ is the label from K classes. Let $\mathcal{L}$ represent a real-valued loss function for classification (*e.g.*,cross-entropy loss). Denote the communication neighboring nodes of the client C_i in the system as $\mathcal{N}(C_i)$ and the local models as $\{M_i = \rho_i \circ \psi_i\}_{i=1}^{i=N}$, where ψ_i represents the feature encoder and ρ_i represents the classification head for the ith client's model M_i. DeSA returns trained client models $\{M_i\}_{i=1}^{i=N}$.

Our work aims to connect two key areas, heterogeneous FL and decentralized FL, termed as *decentralized federated mutual learning*, where we train **multiple client models** in a decentralized way such that they can **generalize** well across all clients' data domains. Mathematically, our objective is formulated as, for every client i,

$$M_i^* = \arg\min_{M_i \in \mathcal{M}_i} \underbrace{\mathbb{E}_{\mathbf{x},y\sim P_i}[\mathcal{L}(M_i(\mathbf{x}), y)]}_{\text{Intra-client}} + \underbrace{\sum_{j\in\mathcal{N}(C_i)} \mathbb{E}_{\mathbf{x},y\sim P_j}[\mathcal{L}(M_i(\mathbf{x}), y)]}_{\text{Inter-client}}, \quad (2)$$

where M_i^* is the best possible model for client i with respect to the model space $\mathcal{M}_i$.

Overview of DeSA. The overall objective of DeSA is to improve local models' generalizability in FL training under both model and data heterogeneity in a serverless setting as shown in in Figure 2(a). The pipeline of DeSA is depicted in Figure 2(b). Our algorithm contains three important aspects: 1) we generate synthetic anchor data by matching raw data distribution and share them amongst the client's neighbors; 2) we train each client model locally with a synthetic anchor-based feature regularizer; and 3) we allow the models to learn from each other via knowledge distillation based on the synthetic anchors. The effectiveness of steps 2 and 3 can be observed in Figure 1. The next three subsections delve deeper into these three designs. The full algorithm is depicted in Algorithm 1.

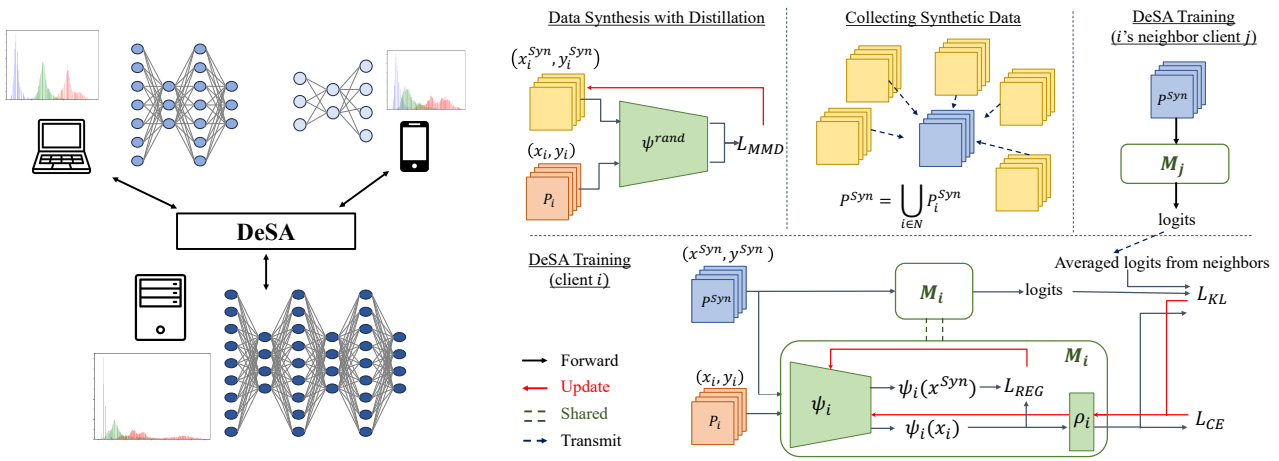
3.2. Synthetic Anchor Datasets Generation

The recent success of dataset distillation-based data synthesis technique that generates data with similar representation power as the original raw data (Zhao et al., 2020; Zhao & Bilen, 2023). Thus, we propose to leverage this method to efficiently and effectively generate a synthetic anchor dataset without requiring any additional model pretraining. Inspired by our theoretical analysis in Sec. 4, we utilize distribution matching (Zhao & Bilen, 2023) to distill local synthetic anchor data using the empirical maximum mean discrepancy loss (MMD) (Gretton et al., 2012) as follows,

$$D_i^{Syn} = \arg\min_{D} ||\frac{1}{|D_i|}\sum_{(\mathbf{x},y)\in D_i} \psi^{\text{rand}}(\mathbf{x}|y) - \frac{1}{|D|}\sum_{(\mathbf{x},y)\in D} \psi^{\text{rand}}(\mathbf{x}|y)||^2, \quad (3)$$

where ψ^{rand} is a randomly sampled feature extractor for each iteration, D_i is the raw data for client i, and D_i^{Syn} is its target synthetic data. Following Eq. 3, we can manipulate the synthetic anchor dataset generation in a class-balanced manner, which enables the label prior to being unbiased towards a set of classes.

Similar to other FL work sharing distilled synthetic data for efficiency (Song et al., 2023), After local data synthesis, we request each client to share it among peers to ensure they possess same global information, and the global synthetic



(a) Heterogeneous Setup

(b) DESA pipeline

Figure 2: Heterogeneous setup and DESA pipeline. (a) We assume a realistic FL scenario, where clients have different data distributions and computational powers, which results in different model architectures. (b) DESA pipeline consists of three phases, local data synthesis (top left) , global synthetic data aggregation (top right)(Section 3.2), and decentralized training (bottom) using anchor regularization(Section 3.3) and knowledge distillation (Section 3.4).

anchor data is denoted as $D^{Syn} = \cup_i D_i^{Syn}$ [1]. As shown in Algorithm 1, our method is designed to work only to receive neighbor node information, and DeSA is designed for peer-to-peer decentralized network. Since the loss requires all nodes' information, we can leverage the FastMix algorithm to aggregate all nodes' information as in (Ye et al., 2020; Luo & Ye, 2022). This method can aggregate all nodes' information via adjacent nodes' communication at a linear speed. It is very common in fully decentralized optimization. In fact, our method can also work if each node can only receive neighbor nodes' information during training, and we empirically show the feasibility in our CIFAR10C experiments by sampling neighboring clients.

It is worth noting that, different from (Song et al., 2023), we further propose novel loss terms and training strategies to help mitigate the distribution discrepancy between the clients, which are detailed in the following sections (see Sec. 3.3 and Sec. 3.4), enabling improved model performance, as intuitively incorporating D^{Syn} into training can only achieve sub-optimal results (see Figure 3 when both λ_{KD} and λ_{REG} equals to 0.).

3.3. REG Loss for Feature Regularization

The synthetic anchor regularization loss enforces the model to learn a **client-domain invariant** representation of the data. (Tang et al., 2022) and other domain incremental works (Rostami, 2021) show that, adding a distribution discrepancy based loss in the **latent space** enables learning of a domain-invariant encoder ψ. However, most of the domain adaptation works require explicit access to the real data from other domains, or generates random noise as anchors. We propose using the latent space distribution of the synthetic anchor data D^{Syn} as a synthetic anchor to which the client-model specific encoders ψ_i can project their **local private data** onto. The loss function can be therefore defined as,

$$\mathcal{L}_{REG}(\psi_i) = \mathbb{E}[d(\psi_i(D^{Syn})||\psi_i(D_i))], \tag{4}$$

where K is the number of classes, d is distance computed using the supervised contrastive loss,

$$d(\psi_i; D^{Syn}, D_i) = \sum_{j \in B} -\frac{1}{|B_{\setminus j}^{y_j}|} \sum_{\mathbf{x}_p \in B_{\setminus j}^{y_j}} \log \frac{\exp(\psi_i(\mathbf{x}_j) \cdot \psi_i(\mathbf{x}_p)/\tau_{temp})}{\sum_{\mathbf{x}_a \in B_{\setminus j}} \exp(\psi_i(\mathbf{x}_j) \cdot \psi_i(\mathbf{x}_a)/\tau_{temp})} \tag{5}$$

where $B_{\setminus j}$ represents a batch containing both local raw data D_i and global synthetic data D^{Syn} but without data j, $B_{\setminus j}^{y_j}$ is a subset of $B_{\setminus j}$ only with samples belonging to class y_j, and τ_{temp} is a scalar temperature parameter. Note that we will detach the synthetic anchor data to ensure we are pulling local features to global features.

3.4. Knowledge Distillation for Information Exchange

This step allows a single client model to learn from all the other models using a common synthetic anchor dataset D^{Syn}. Under our setting of model heterogeneity among clients, we cannot aggregate the model parameters by simply averaging as in FedAvg (McMahan et al., 2017). Instead, we

[1] By default, we perform simple interpolation (averaging) among clients as it is shown that using this mixup strategy can improve model fairness (Chuang & Mroueh, 2021).

propose to utilize knowledge distillation (KD) (Hinton et al., 2015) for decentralized model aggregation. Specifically, the fact that D^{Syn} is representative of the joint distributions of the clients allows it to be an ideal dataset for knowledge transfer. Thus, to enable the client model to mimic the predictions of the other models, we also incorporate KD loss using D^{Syn}, formulated as

$$\mathcal{L}_{KD}(M_i) = \mathcal{L}_{KL}(M_i(\mathbf{x}^{Syn}), \bar{Z}_i),$$
$$\bar{Z}_i = \frac{1}{|N(C_i)|} \sum_{j \in N(C_i)} M_j(\mathbf{x}^{Syn}), \tag{6}$$

where $(\mathbf{x}^{Syn}, y^{Syn}) \sim D^{Syn}$, $N(i)$ is the neighbor clients of client C_i, and $\mathcal{L}_{KL}$ is the KL-Divergence between the output logits of $\mathbf{x}^{Syn}$ on M_i and the averaged output logits of $\mathbf{x}^{Syn}$ on $M_j, \forall j \in N(C_i)$. Finally, we formulate our objective function as

$$\mathcal{L} = \mathcal{L}_{CE}(D_i \cup D^{Syn}; M_i) + \lambda_{REG}\mathcal{L}_{REG}(D_i, D^{Syn}; M_i)$$
$$+ \lambda_{KD}\mathcal{L}_{\text{KD}}(D^{Syn}; M_i, \bar{Z}_i), \tag{7}$$

where $\mathcal{L}_{CE}(D; M)$ is the K-classes cross entropy loss on data D and model M. λ_{REG} and λ_{KD} are the hyperparameters for regularization and KD losses, $\mathcal{L}_{REG}$ and $\mathcal{L}_{KD}$ are as defined in Eq. 4 and Eq. 6, and $\bar{Z}_i$ is the shared logits from neighboring clients $N(C_i)$. We also incorporate our class-conditional-generated global synthetic data D^{Syn} in the CE loss to enforce models to perform well on general domains and to benefit from the augmented dataset. Overall, we formulate our objective function as

Algorithm 1 Serverless DeSA (Procedures for Client i)

procedure INIT(C_i)
  for all $j \in \mathcal{N}(C_i)$ **do**
    $D^{Syn} = D^{Syn} \cup$ GET-IMG(C_j)
procedure LOCALTRAIN(C_i, t)
  if client C_i is sampled **then**
    share $Z_i = M_i(D^{Syn})$ to $\mathcal{N}(C_i)$
    get $\bar{Z}_i = 1/|\mathcal{N}(C_i)| \sum_j^{j \in \mathcal{N}(C_i)} Z_j$
    $\mathcal{L}_{CE}$ = CLASSIFICATION($D_i \cup D^{Syn}; M_i$)
    $\mathcal{L}_{REG}$ = FEATURE-REG($D_i, D^{Syn}; M_i$)
    $\mathcal{L}_{KD}$ = KD($D^{Syn}; M_i, \bar{Z}$)
    $\mathcal{L} = \mathcal{L}_{CE} + \lambda_{REG}\mathcal{L}_{REG} + \lambda_{KD}\mathcal{L}_{KD}$
    $M_i = M_i - \eta \nabla_{M_i} \mathcal{L}$

4. Theoretical Analysis

In this section, we focus on providing a theoretical justification for our algorithm. The technical challenge is to analyze the effect of minimizing the overall loss function (Eq. 7) on the generalizability on the global data distribution P^T of a client model.

Notation: Here, we state the intuitive definitions borrowed from (Ben-David et al., 2010). For the precise definitions please refer to the notation table in Appendix A. The domain pair (P, f^P) represents the source distribution and its optimal labeling function. The ϵ - error of a hypothesis M on the domain pair (P, f^P) is defined as the probability of a mismatch between the optimal labeling function f^P and the hypothesis M. Additionally, the $\mathcal{H}\Delta\mathcal{H}$ divergence ($d_{\mathcal{H}\Delta\mathcal{H}}$) describes a distance measure between two distributions.

Analysis: Our analysis focuses on the generalization on a global data distribution P^T that is the average of the client distributions (Marfoq et al., 2021), with labeling function f^T the same as f^S . We assume that the minimization of loss in Eq (2) matches the optimal labeling function f^T of the global distribution P^T, and formalizes the intuition of generalizing over closely related client distributions.

We proceed by defining the distribution of our global synthetic data as P^{Syn} with the corresponding labeling function f^{Syn}. As P^{Syn} is also leveraged for the knowledge distillation, inspired by (Feng et al., 2021) we describe $(P^{Syn}_{KD}, f^{Syn}_{KD})$ as follows.

Definition 4.1. The extended knowledge distillation (KD) domain pair $(P^{Syn}_{KD}, f^{Syn}_{KD})$ of a client C_i, originating from the KD dataset D^{Syn}_{KD} is defined as

$$D^{Syn}_{KD} = \{\mathbf{x}^{Syn}, \frac{1}{|N(C_i)|} \sum_{j \in N(C_i)} M_j(\mathbf{x}^{Syn})\} \sim (P^{Syn}_{KD}, f^{Syn}_{KD})$$

where $M_i(\mathbf{x}^{Syn})$ is the predicted logit on global synthetic data $\mathbf{x}^{Syn} \sim P^{Syn}(x)$.

Definition 4.2. We define the the overall source distribution of the client C_i as P^S_i, which is a convex combination of the local and synthetic distributions

$$P^S_i = \alpha P_i + \alpha^{Syn} P^{Syn} + \alpha^{Syn}_{KD} P^{Syn}_{KD} \tag{8}$$

The positive component weights $\{\alpha, \alpha^{Syn}, \alpha^{Syn}_{KD}\}$ describe the dependence of the client C_i, on the local, synthetic and knowledge distillation data, and $\alpha + \alpha^{Syn} + \alpha^{Syn}_{KD} = 1$.

Theorem 1. *Denote the client C_i's model as $M_i = \rho_i \circ \psi_i \in \mathcal{P}_i \circ \Psi_i = \mathcal{M}_i$ and its overall source distribution as P^S_i with component weights ($\boldsymbol{\alpha}$). Then the generalization error on the global data distribution P^T can be bounded as follows*

$$\epsilon_{P^T}(M_i) \leq \epsilon_{P^S_i}(M_i) + \alpha \mathbf{C}(P_i, P_T)$$
$$+ \alpha^{Syn}\epsilon_{P^T}(f^{Syn}) + \alpha^{Syn}_{KD}\epsilon_{P^T}(f^{Syn}_{KD})$$
$$+ \frac{(1-\alpha)}{2} d_{\mathcal{P}_i \Delta \mathcal{P}_i}(\psi \circ P^{Syn}, \psi \circ P^T) \tag{9}$$

where $\mathbf{C}(P_i, P_T)$ are small distance terms depending on the distributions P_i and P_T.

Below we give a short summary of the interpretation of our theorem in 4.3. For a more detailed interpretation, please refer to Appendix A.

Remark 4.3. The first term is minimized via local cross-entropy loss. The second term is a constant given data distributions. The third and fourth terms measure the discrepancy between the labeling function of the target domain f_T and the labeling function f^{Syn} or f^{Syn}_{KD} of the corresponding synthetic distribution P^{Syn} or P^{Syn}_{KD}. With proper dataset distillation (Zhao & Bilen, 2023), we have the model trained on D^{Syn} similar to that trained by D^T, *i.e.*, $\mathbb{E}_{\mathbf{x}\sim P^T}[l(M^T(\mathbf{x}), y)] \simeq \mathbb{E}_{\mathbf{x}\sim P^{Syn}}[l(M^{Syn}(\mathbf{x}), y)]$, implying $f^{Syn} \rightarrow f^T$ and then we have a small $\epsilon_{P^T}(f^{Syn})$. A small $\epsilon_{P^T}(f^{Syn}_{KD})$ can be achieved when every client achieves low $\mathcal{L}_{CE}$, indicating the model ability to learn f^{Syn}_{KD} approximating to f^T. The last term $d_{\mathcal{P}_i \Delta \mathcal{P}_i}$ motivates the need for reducing our domain-invariant regularizer in Eq. 4, elicited to be bounded.

Furthermore, for the domain pair (D, f^D), we denote $\mathcal{J}(D) = |\epsilon_D(M) - \epsilon_{P^T}(M)| + \epsilon_{P^T}(f^D)$. The following proposition implies our generalization bound in Theorem 1 is tighter than the generalization bound of training with local data in Eq. 11 (Ben-David et al., 2010) under some mild conditions.

Proposition 2. *Under the conditions in Theorem 1, if it further holds that*

$$\sup_{M\in\mathcal{M}_i} \min\{\mathcal{J}(P^{Syn}), \mathcal{J}(P^{Syn}_{KD})\} \leq \inf_{M\in\mathcal{M}_i} (\epsilon_{P_i}(M) - \epsilon_{P^T}(M)) + \mathbf{C}(P_i, P_T) \quad (10)$$

then we can get a tighter generalization bound on the ith client's model M_i than learning with local data only.

The key idea of Proposition 2 is to have the generalization bounds induced by P^{Syn} and P^{Syn}_{KD} is smaller than the generalization bound by the local training data distribution P_i. When the local data heterogeneity is severe, $\inf_{M\in\mathcal{M}_i}(\epsilon_{P_i}(M) - \epsilon_{P^T}(M))$ and $\mathbf{C}(P_i, P^T)$ would be large. As the synthetic data and the extended KD data are approaching the the global data distribution, the left side term in (10) would be small. Thus, the above proposition points out that, to reach better generalization, the model learning should rely more on the synthetic data and the extended KD data, when the local data are highly heterogeneous and the synthetic and the extended KD datasets are similar to the global ones.

5. Experiment

5.1. Training Setup

Datasets and Models We extensively evaluate DeSA under data heterogeneity in our experiments. Specifically, we consider three classification tasks on three sets of domain-shifted datasets:
1) `DIGITS`={MNIST (LeCun et al., 1998), SVHN (Netzer et al., 2011), USPS (Hull, 1994), SynthDigits (Ganin & Lempitsky, 2015), MNIST-M (Ganin & Lempitsky, 2015)} contains digits from different styles, and each dataset represents one client.
2) `OFFICE`={Amazon (Saenko et al., 2010), Caltech (Griffin et al., 2007), DSLR (Saenko et al., 2010), and WebCam (Saenko et al., 2010)} contains images from different cameras and environments, and, similarly, each dataset represents one client.
3) `CIFAR10C` consists 57 subsets with domain- and label-shifted datasets sampled using Dirichlet distribution with $\beta = 2$ from Cifar10-C (Hendrycks & Dietterich, 2019).
More information about datasets and image synthesis can be found in Appendix E. In our model heterogeneity experiments (Sec. 5.2), we randomly assign model architectures from {ConvNet, AlexNet} for each client, while in model homogeneous experiments, we use ConvNet for all clients (see Appendix F for model details).

Comparison Methods We compare DeSA with two sets of baseline federated learning methods: one considers heterogeneous models (Sec. 5.2) and the other considers homogeneous models (Sec. 5.3). For ***heterogeneous model experiments***, we compare with FedHe (Chan & Ngai, 2021), FedDF (Lin et al., 2020), FCCL (Huang et al., 2022), and FedProto (Tan et al., 2022), and assume the clients owns personalized models[2]. For ***homogeneous model experiments***, we compare with FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020b), MOON (Li et al., 2021b), Scaffold (Karimireddy et al., 2020), FedGen (Zhu et al., 2021), and VHL (Tang et al., 2022), and assume these baseline methods can leverage a server for global model aggregation.

FL Training Setup If not otherwise specified, we use SGD optimizer with a learning rate of 10^{-2}, and our default setting for local model update epochs is 1, total update rounds is 100, and the batch size for local training is 32. Since we only have a few clients for `DIGITS` and `OFFICE` experiments, we will select all the clients for each iteration, while we randomly sample 10% and 20% clients for each round when performing `CIFAR10C` experiments. By default, λ_{REG} and λ_{KD} are set to 1.

[2]For the purposes of this comparison, we have excluded FedFTG (Zhang et al., 2022b) and DENSE (Zhang et al., 2022a), which address heterogeneities in different learning scenarios. FedFTG focuses on fine-tuning a global model, and DENSE belongs to one-shot FL, and both of them requires aggregate local information and train a generator on the server side. Note that none of the data-sharing-based baseline methods employ privacy-preserving techniques.

Table 2: Heterogeneous model experiments. We compare with model heterogeneous FL methods and report the averaged global accuracy over all client models.The best accuracy is marked in **bold**. Observe that DeSA can achieve best averaged global accuracy on DIGITS and OFFICE. For CIFAR10C, DeSA can outperform most of the baseline methods except for FCCL, which utilizes CIFAR100 as the public dataset. This is because CIFAR100 and CIFAR10C have a genuine semantic overlap.

		DIGITS						OFFICE					CIFAR10C	
		MN(C)[a]	SV(A)	US(C)	Syn(A)	MM(C)	Avg	AM(A)	CA(C)	DS(A)	WE(A)	Avg	0.1	0.2
FedHe		59.51	66.67	49.89	75.39	71.57	64.81	33.33	47.17	36.86	52.96	42.59	47.73	51.26
FedDF	Cifar100	65.98	65.21	61.30	69.65	**74.48**	67.32	38.87	49.51	33.12	46.89	42.09	27.69	35.70
	FMNIST	43.05	69.14	44.95	74.67	71.27	60.61	39.13	46.53	40.23	43.77	42.36	28.26	36.50
FCCL	Cifar100	-	-	-	-	-	-	38.22	49.10	44.68	52.26	46.07	51.70	50.78
	FMNIST	46.43	61.02	42.64	63.05	66.39	55.91	27.39	46.78	38.56	48.47	40.30	**52.40**	51.83
FedProto		62.59	71.74	58.52	**81.19**	74.44	69.70	38.08	25.06	26.49	47.22	34.21	16.82	31.39
DeSA(D_{VHL}^{Syn})[b]		54.40	62.03	42.34	67.75	73.03	59.91	8.82	48.98	16.90	49.13	30.96	47.49	52.04
DeSA		**70.12**	**76.17**	**71.17**	81.10	73.83	**74.47**	**51.35**	**52.80**	**52.17**	**52.31**	**54.46**	48.19	**52.80**

[a] The letter inside the parenthesis is the model architecture used by the client. A and C represent AlexNet and ConvNet, respectively.
[b] For VHL baseline, we use the synthetic data sampling strategy in VHL only. The purpose is to show DeSA can generate better synthetic anchor data for feature regularization and knowledge distillation.

5.2. Heterogeneous Model Experiments

The objective of the experiments is to show that DeSA can effectively leverage and learn generalized information from other clients under data and model heterogeneities. Thus, we report the averaged global accuracy by testing i-th local model on every client j's ($\forall j, j \in [N]$) test sets. Note FedDF and FCCL require accessing to public available data. To make a fair comparison, we use FMNIST (Xiao et al., 2017) and Cifar100 (Krizhevsky et al., 2009) as public datasets for knowledge distillation. DeSA(D_{VHL}^{Syn}) uses our training pipeline, but the synthetic data is sampled from an untrained StyleGAN (Karras et al., 2019) as in VHL (Tang et al., 2022).

Based on our experiment results for heterogeneous models in Table. 2, it is evident that our method DeSA significantly enhances the overall accuracy on DIGITS and OFFICE. In the CIFAR10C experiments, we increase the total training rounds to 200 since the performance of FL is notably hindered by a low client sampling ratio, especially in the case of serverless methods. As shown in the table, DeSA can improve the global accuracy and out-performs most of the baseline methods, except for FCCL when the client sampling ratio is 0.1. We believe this is due to the extreme decentralized learning setting, *i.e.,* under model and data heterogeneity, serverless, and low client sampling ratio scenarios. FCCL utilizes pre-trained models, which provides a good starting point. However, we observe that the method tends to overfit easier by the fact that the performance drops when the client sampling ratio increases. Moreover, it is worth mentioning that the accuracies of FedDF and FCCL vary significantly when switching from Cifar100 to FMNIST (the FedDF-DIGITS and FCCL-OFFICE results). We found that the training with both methods are easy to obtain nan loss, which we mark as '-' in the table. This suggests that these methods strongly rely on the public data, which restricts the utility of the methods. In contrast, DeSA does not depend on public data, and furthermore, it is completely serverless.

5.3. Homogeneous Model Experiments

Table 3: Homogeneous model experiments. We compare with model homogeneous FL methods and report the averaged local accuracy. The best accuracy is marked in **bold**.

	DIGITS	OFFICE	CIFAR10C	
			0.1	0.2
FedAvg	94.20	76.45	65.26	66.40
FedProx	94.19	76.45	65.33	66.36
MOON	94.37	73.64	64.74	66.70
Scaffold	94.95	77.52	**65.66**	67.15
VHL	94.11	75.69	64.67	66.55
FedGen	82.62	63.60	45.77	48.10
DeSA	**95.53**	**82.92**	64.47	**68.13**

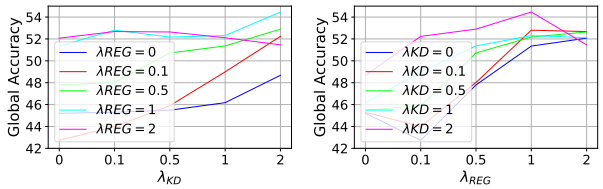
Among the baseline methods for homogeneous model experiments, they learn a generalizable global model via model aggregation on a server, which is not required in DeSA. As these baseline methods are only evaluated on their single global model, for a fair comparison, we report the averaged local accuracy in the experiments. Specifically, the average local accuracy for the baseline methods is the average performance over testing the global model over all the clients; while we report, we report the average local accuracy on testing i-th local model on client i's test set over all the clients.

One can observe from Table 3 that DeSA can effectively leverage local information and outperforms other methods DIGITS and OFFICE. For CIFAR10C, although DeSA has highest averaged local accuracy when client ratio is 0.2, it has lower performance when client ratio is 0.1. This is because smaller client sampling ratios have a larger impact on decentralized learning as we do not have a global model, and thus some clients may suffer from low model performance

due to insufficient training and the scarce global information from the sampled neighbor clients. Overall, despite the serverless setting, DESA is compatible with the baseline methods that have central servers.

5.4. Ablation studies for DESA

The effectiveness of DESA relies on the novel designs of synthetic anchor data and the losses. To evaluate how these designs influences the performance of DESA, we vary the number of synthetic anchor data (IPC) and the loss coefficients ($\lambda's$) in the following paragraphs. If not otherwise specified, we use the default hyperparameters and model heterogeneous setting. We report the averaged global accuracy in this section.



(a) λ_{KD} (b) λ_{REG}

Figure 3: Ablation studies for λ's using OFFICE. We report the averaged global accuracy when changing the λ values.

Table 4: Ablation study on the size of synthetic dataset using DIGITS. We use Images-Per-Class (IPC) to show its size.

IPC	5	10	20	50	100	200
Global Acc	70.12	72.74	72.46	74.32	70.29	70.45

Evaluation of λ Selections λ_{KD} and λ_{REG} play an important role to help the clients to learn generalized information as well as improving the performance on local data. We select OFFICE as the candidate dataset for this set of experiments because it has larger domain-shift. We vary both of λ_{KD} and λ_{REG} between 0 to 2 and report the global accuracy in Figure 3(a). One can observe that when λ_{KD} or λ_{REG} increases, the overall global accuracy increases. However, in Figure 3(b), when we increase λ_{REG} to 2, the performance drops. This happens because the magnitude of REG loss term dominates the total training loss. Overall, we conclude that both λ_{KD} helps the local model learn information from other clients' models, and λ_{REG} improves the global performance by enforcing the local model to learn generalized features.

Evaluation of Size of Synthetic Dataset The size of synthetic data is a critical hyperparameter for DESA as it represents the shared local information. Since DESA synthesizes class-balanced data, we use Images-Per-Class (IPC) to represent the size of the synthetic data. We select DIGITS as the candidate dataset for this set of experiments because it contains larger number of data for each client, which allows us to increase IPC up to 200. One can observe in Table 4 that blindly increasing the IPC does not guarantee to obtain optimal global accuracy. It will cause the loss function to be dominated by the last 2 terms of Eq. 9, *i.e.,* by synthetic data. However, synthesizing larger number of synthetic data may degrade its quality, and the sampled batch for $\mathcal{L}_{REG}$ may fail to capture the distribution.

5.5. Further Discussion

Communication Overhead Although DESA requires training and transferring local synthetic data to every clients, the process happens before the FL training, and can be preprocessed offline. During DESA training, clients only need to share logits *w.r.t.* global synthetic data, resulting in a lightweight *in-training* communication overhead. We discuss the communication of DESA compared to standard FL in Appendix D, and show that DESA has lower overall communication overhead.

Privacy DESA requires sharing image-level information among clients in FL, which may raise privacy concerns. Therefore, we empirically show that our distilled synthetic data can protect against some privacy attacks in Appendix B.1. We further discuss DESA's potential for higher privacy guarantee using Differential Privacy (Abadi et al., 2016) in Appendix B.2. Furthermore, we claim that decentralized FL with both data and model heterogeneities is an extremely challenging setting, where existing solutions either require sharing real public data (Lin et al., 2020; Huang et al., 2022) or synthetic data generated from GAN-based generator (Zhang et al., 2022a;b)

.

Theory vs. Practice We note that obtaining the tight bound from our theoretical findings in Theorem 1 requires proper dataset distillation. In Section 3.2, we propose an efficient approximate solution for dataset distillation. Although perfect dataset distillation may not be achievable (as shown in the visualization in Appendix E), we have found through experimentation that using our synthetic data in combination with the proposed $\mathcal{L}_{REG}$ (Eq. 4) and $\mathcal{L}_{KD}$ (Eq. 6) can already lead to improved generalization.

6. Conclusion

A novel and effective method, DESA, is presented that utilizes synthetic data to deal with both data and model heterogeneities in serverless decentralized FL. In particular, DESA introduces a pipeline that involves synthetic data generalization, and we propose a new scheme that incorporates the synthetic data as anchor points in decentralized FL model training. To address heterogeneity issues, we utilize the synthetic anchor data and propose two regularization losses: anchor loss and knowledge distillation loss. We

provide theoretical analysis on the generalization bound to justify the effectiveness of DeSA using the synthetic anchor data. Empirically, the resulted client models not only achieve compelling local performance but also can generalize well onto other clients' data distributions, boosting cross-domain performance. Through extensive experiments on various classification tasks, we show that DeSA robustly improves the efficacy of collaborative learning when compared with state-of-the-art methods, under both model and data heterogeneous settings.

Acknowledgement

This work is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), Public Safety Canada, CIFAR Catalyst Grant, and Compute Canada Research Platform.

Impact Statement

This paper represents an effort to progress the field of machine learning and distributed learning. Our work has various potential societal impacts, although we do not believe any specific one must be highlighted here.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

Albuquerque, I., Monteiro, J., Darvishi, M., Falk, T. H., and Mitliagkas, I. Generalizing to unseen domains via distribution matching. *arXiv preprint arXiv:1911.00804*, 2019.

Assran, M., Loizou, N., Ballas, N., and Rabbat, M. Stochastic gradient push for distributed deep learning. In *International Conference on Machine Learning*, pp. 344–353. PMLR, 2019.

Beltrán, E. T. M., Pérez, M. Q., Sánchez, P. M. S., Bernal, S. L., Bovet, G., Pérez, M. G., Pérez, G. M., and Celdrán, A. H. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. *IEEE Communications Surveys & Tutorials*, 2023.

Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19, 2006.

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

Carlini, N., Chien, S., Nasr, M., Song, S., Terzis, A., and Tramer, F. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914. IEEE, 2022a.

Carlini, N., Feldman, V., and Nasr, M. No free lunch in" privacy for free: How does dataset condensation help privacy". *arXiv preprint arXiv:2209.14987*, 2022b.

Cazenavette, G., Wang, T., Torralba, A., Efros, A. A., and Zhu, J.-Y. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4750–4759, 2022.

Chan, Y. H. and Ngai, E. C. Fedhe: Heterogeneous models and communication-efficient federated learning. In *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*, pp. 207–214. IEEE, 2021.

Chang, K., Balachandar, N., Lam, C., Yi, D., Brown, J., Beers, A., Rosen, B., Rubin, D. L., and Kalpathy-Cramer, J. Distributed deep learning networks among institutions for medical imaging. *Journal of the American Medical Informatics Association*, 25(8):945–954, 2018.

Chuang, C.-Y. and Mroueh, Y. Fair mixup: Fairness via interpolation. *arXiv preprint arXiv:2103.06503*, 2021.

Crammer, K., Kearns, M., and Wortman, J. Learning from multiple sources. *Journal of Machine Learning Research*, 9(8), 2008.

Cui, J., Wang, R., Si, S., and Hsieh, C.-J. Scaling up dataset distillation to imagenet-1k with constant memory. In *International Conference on Machine Learning*, pp. 6565–6590. PMLR, 2023.

Donahue, K. and Kleinberg, J. Model-sharing games: Analyzing federated learning under voluntary participation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 5303–5311, 2021.

Dong, T., Zhao, B., and Lyu, L. Privacy for free: How does dataset condensation help privacy? *arXiv preprint arXiv:2206.00240*, 2022.

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*, 33:3557–3568, 2020.

Feng, H., You, Z., Chen, M., Zhang, T., Zhu, M., Wu, F., Wu, C., and Chen, W. Kd3a: Unsupervised multi-source decentralized domain adaptation via knowledge distillation. In *ICML*, pp. 3274–3283, 2021.

Ganin, Y. and Lempitsky, V. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pp. 1180–1189. PMLR, 2015.

Gao, D., Yao, X., and Yang, Q. A survey on heterogeneous federated learning. *arXiv preprint arXiv:2210.04505*, 2022.

Ghosh, A., Chung, J., Yin, D., and Ramchandran, K. An efficient framework for clustered federated learning. *IEEE Transactions on Information Theory*, 68(12):8076–8091, 2022.

Gong, X., Sharma, A., Karanam, S., Wu, Z., Chen, T., Doermann, D., and Innanje, A. Preserving privacy in federated learning with ensemble cross-domain knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 11891–11899, 2022.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Griffin, G., Holub, A., and Perona, P. Caltech-256 object category dataset. 2007.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

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

Huang, W., Ye, M., and Du, B. Learn from others and be yourself in heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10143–10153, 2022.

Huang, Y., Chu, L., Zhou, Z., Wang, L., Liu, J., Pei, J., and Zhang, Y. Personalized cross-silo federated learning on non-iid data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 7865–7873, 2021a.

Huang, Y., Gupta, S., Song, Z., Li, K., and Arora, S. Evaluating gradient inversion attacks and defenses in federated learning. *Advances in Neural Information Processing Systems*, 34:7232–7241, 2021b.

Hull, J. J. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M., and Kim, S. Federated distillation and augmentation under non-iid private data. *NIPS Wksp. MLPCD*, 2018.

Jiang, M., Yang, H., Cheng, C., and Dou, Q. Iop-fl: Inside-outside personalization for federated medical image segmentation. *IEEE Transactions on Medical Imaging*, 2023.

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.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lee, S., Chun, S., Jung, S., Yun, S., and Yoon, S. Dataset condensation with contrastive signals. In *International Conference on Machine Learning*, pp. 12352–12364. PMLR, 2022.

Li, C., Li, G., and Varshney, P. K. Decentralized federated learning via mutual knowledge transfer. *IEEE Internet of Things Journal*, 9(2):1136–1147, 2021a.

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

Li, G., Togo, R., Ogawa, T., and Haseyama, M. Dataset distillation for medical dataset sharing. *arXiv preprint arXiv:2209.14603*, 2022.

Li, Q., He, B., and Song, D. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10713–10722, 2021b.

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, 2020a.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of fedavg on non-iid data. *International Conference on Learning Representations*, 2020b.

Lin, T., Kong, L., Stich, S. U., and Jaggi, M. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33: 2351–2363, 2020.

Luo, L. and Ye, H. Decentralized stochastic variance reduced extragradient method. *arXiv preprint arXiv:2202.00509*, 2022.

Marfoq, O., Neglia, G., Bellet, A., Kameni, L., and Vidal, R. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34:15434–15447, 2021.

Matsuda, K., Sasaki, Y., Xiao, C., and Onizuka, M. Fedme: Federated learning via model exchange. In *Proceedings of the 2022 SIAM international conference on data mining (SDM)*, pp. 459–467. SIAM, 2022.

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.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.

Pappas, C., Chatzopoulos, D., Lalis, S., and Vavalis, M. Ipls: A framework for decentralized federated learning. In *2021 IFIP Networking Conference (IFIP Networking)*, pp. 1–6. IEEE, 2021.

Rostami, M. Lifelong domain adaptation via consolidated internal distribution. *Advances in neural information processing systems*, 34:11172–11183, 2021.

Roy, A. G., Siddiqui, S., Pölsterl, S., Navab, N., and Wachinger, C. Braintorrent: A peer-to-peer environment for decentralized federated learning. *arXiv preprint arXiv:1905.06731*, 2019.

Saenko, K., Kulis, B., Fritz, M., and Darrell, T. Adapting visual category models to new domains. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pp. 213–226. Springer, 2010.

Sheller, M. J., Reina, G. A., Edwards, B., Martin, J., and Bakas, S. Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part I 4*, pp. 92–104. Springer, 2019.

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):12598, 2020.

Shen, T., Zhang, J., Jia, X., Zhang, F., Lv, Z., Kuang, K., Wu, C., and Wu, F. Federated mutual learning: a collaborative machine learning method for heterogeneous data, models, and objectives. *Frontiers of Information Technology & Electronic Engineering*, 24(10):1390–1402, 2023.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Song, R., Liu, D., Chen, D. Z., Festag, A., Trinitis, C., Schulz, M., and Knoll, A. Federated learning via decentralized dataset distillation in resource-constrained edge environments. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10. IEEE, 2023.

Tan, Y., Long, G., Liu, L., Zhou, T., Lu, Q., Jiang, J., and Zhang, C. Fedproto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8432–8440, 2022.

Tang, Z., Zhang, Y., Shi, S., He, X., Han, B., and Chu, X. Virtual homogeneity learning: Defending against data heterogeneity in federated learning. *arXiv preprint arXiv:2206.02465*, 2022.

Wang, H.-P., Chen, D., Kerkouche, R., and Fritz, M. Fedgloss-dp: Federated, global learning using synthetic sets with record level differential privacy. *arXiv preprint arXiv:2302.01068*, 2023.

Wang, J., Liu, Q., Liang, H., Joshi, G., and Poor, H. V. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

Wang, K., Zhao, B., Peng, X., Zhu, Z., Yang, S., Wang, S., Huang, G., Bilen, H., Wang, X., and You, Y. Cafe: Learning to condense dataset by aligning features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12196–12205, 2022.

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

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xiong, Y., Wang, R., Cheng, M., Yu, F., and Hsieh, C.-J. Feddm: Iterative distribution matching for communication-efficient federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16323–16332, 2023.

Yang, R., Tian, J., and Zhang, Y. Regularized mutual learning for personalized federated learning. In *Asian Conference on Machine Learning*, pp. 1521–1536. PMLR, 2021.

Ye, H., Zhou, Z., Luo, L., and Zhang, T. Decentralized accelerated proximal gradient descent. *Advances in Neural Information Processing Systems*, 33:18308–18317, 2020.

Ye, R., Ni, Z., Xu, C., Wang, J., Chen, S., and Eldar, Y. C. Fedfm: Anchor-based feature matching for data heterogeneity in federated learning. *arXiv preprint arXiv:2210.07615*, 2022.

Yuan, L., Ma, Y., Su, L., and Wang, Z. Peer-to-peer federated continual learning for naturalistic driving action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5249–5258, 2023a.

Yuan, L., Sun, L., Yu, P. S., and Wang, Z. Decentralized federated learning: A survey and perspective. *arXiv preprint arXiv:2306.01603*, 2023b.

Zhang, J., Chen, C., Li, B., Lyu, L., Wu, S., Ding, S., Shen, C., and Wu, C. Dense: Data-free one-shot federated learning. *Advances in Neural Information Processing Systems*, 35:21414–21428, 2022a.

Zhang, L., Shen, L., Ding, L., Tao, D., and Duan, L.-Y. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10174–10183, 2022b.

Zhang, Y., Xiang, T., Hospedales, T. M., and Lu, H. Deep mutual learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4320–4328, 2018.

Zhao, B. and Bilen, H. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning*, pp. 12674–12685. PMLR, 2021.

Zhao, B. and Bilen, H. Synthesizing informative training samples with gan. *arXiv preprint arXiv:2204.07513*, 2022.

Zhao, B. and Bilen, H. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 6514–6523, 2023.

Zhao, B., Mopuri, K. R., and Bilen, H. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2020.

Zhou, T., Zhang, J., and Tsang, D. Fedfa: Federated learning with feature anchors to align feature and classifier for heterogeneous data. *arXiv preprint arXiv:2211.09299*, 2022.

Zhu, Z., Hong, J., and Zhou, J. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*, pp. 12878–12889. PMLR, 2021.

Road Map of Appendix Our appendix is organized into five sections. The theoretical analysis and proof is in Appendix A. Appendix B.1 shows the results for Membership Inference Attack (MIA) on DeSA trained models using `DIGITS` datasets. Appendix B.2 discusses how we inject DP mechanism in our data synthesis process, and shows that using DP synthetic anchor data for DeSA can still yeilds comparable performance. Appendix E introduce the selected datasets and how we synthesize anchor data in detail. Appendix F describes the model architectures (ConvNet and AlexNet) we use in our experiments. Finally, Appendix G provides a detailed literature review about the related works. Our code and model checkpoints are available along with the supplementary materials.

A. Theoretical Analysis and Proofs

A.1. Notation

Table 5: Notations used

$\boldsymbol{D_i} = (x_i, y_i)_{i=1}^{i=m}$	$\triangleq$	Local Dataset of client C_i
$P_i(x, y)$	$\triangleq$	The local joint distribution of client C_i
$L_{CE}(a, b)$	$\triangleq$	Cross entropy Loss between distributions a and b
$L_{KL}(a, b)$	$\triangleq$	KL divergence loss between distributions(a, b)
$\mathcal{M}_i$	$\triangleq$	Model Space of Client C_i
$\mathcal{P}_i$	$\triangleq$	Space of Classifier heads for Client C_i
Ψ_i	$\triangleq$	Space of encoder heads for Client C_i
$M_i = \rho_i \circ \psi_i$	$\triangleq$	Model for client C_i with encoder and decoder heads sampled from $\mathcal{P}_i$ and Ψ_i
$\boldsymbol{\alpha} = [\alpha, \alpha^{Syn}, \alpha_{KD}^{Syn}]$	$\triangleq$	Component weights for the losses, as defined in Eq. 8
		Notations from (Ben-David et al., 2010)
(P, f^P)	$\triangleq$	$P(x)$ is the source data distribution and f^P is the optimal labelling function
$\epsilon_P(M)$	$\triangleq$	$\Pr_{x\sim P(x)}(M(x) \neq f_P(x))$
$d_{\mathcal{H}\Delta\mathcal{H}}(P_s, P_t)$	$\triangleq$	$2 \sup_{h,h'\in\mathcal{H}} \lvert \Pr_{x\sim P_s}(h(x) \neq h(x')) \rvert - \Pr_{x\sim P_t}(h(x) \neq h(x')) \rvert$
$\lambda(P)$	$\triangleq$	Least error of a jointly trained model $=\min_{M\in\mathcal{M}} \epsilon_P(M) + \epsilon_{P^T}(M)$
$\mathbf{C}(P_i, P^T)$	$\triangleq$	A distance term appearing in (Ben-David et al., 2010) $= \frac{1}{2}(d_{\mathcal{M}\Delta\mathcal{M}}(P_i, P^T) + \lambda(P_i)$

A.2. Proof for Theorem 1

Proof. The training data at ith client are from as three distributions: 1) the local source data; 2) the global virtual data; 3) the extended KD data. The data from the first two groups are used for the cross entropy loss and the distribution divergence, while the third is used for Knowledge distillation.

Without loss of generality, at ith client, we set the weight for P_i, P^{Syn} and P_{KD}^{Syn} as α, α^{Syn} and α_{KD}^{Syn}, respectively. For notation simplicity, we assume $\alpha + \alpha^{Syn} + \alpha_{KD}^{Syn} = 1$. Then the training source data at ith client is $P_i^S = \alpha P_i + \alpha^{Syn} P^{Syn} + \alpha_{KD}^{Syn} P_{KD}^{Syn}$.

From Theorem 2 in (Ben-David et al., 2010), it holds that

$$\epsilon_{P^T}(M_i) \leq \epsilon_{P_i}(M_i) + \mathbf{C}(P_i, P_T) \tag{11}$$

where $\mathbf{C}(P_i, P_T) = \frac{1}{2} d_{\mathcal{M}_i\Delta\mathcal{M}_i}(P_i, P^T) + \lambda(P_i)$ and $\lambda(P_i) = \min_{M\in\mathcal{M}_i} \epsilon_{P_i}(M) + \epsilon_{P^T}(M)$ is a constant. These terms are small unless the data heterogeneity is severe (Ben-David et al., 2010). Then with (11) and Lemma 1, we have three inequalities, which we will add after multiplying each one of them with their corresponding component weight $\boldsymbol{\alpha}$.

Furthermore, note that the support of P^{Syn} and P^{Syn}_{KD} are the same. (4.1). Therefore, their distances from the support of the global distribution will also be the same, i.e.

$$d_{\mathcal{P}_i \Delta \mathcal{P}_i}(\psi \circ P^{Syn}, \psi \circ P^T) = d_{\mathcal{P}_i \Delta \mathcal{P}_i}(\psi \circ P^{Syn}_{KD}, \psi \circ P^T)$$

Continuing after adding all three inequalities and using Claim 1 to introduce $\epsilon_{P^S_i}(M)$

$$\begin{aligned}
\epsilon_{P^T}(M_i) \leq & \epsilon_{P^S_i}(M_i) + \mathbf{C}(P_i, P^T) + \frac{\alpha^{Syn}}{2} d_{\mathcal{P}\Delta\mathcal{P}}(\psi \circ P^{Syn}, \psi \circ P^T) \\
& + \epsilon_{P^T}(f^{Syn})) + \frac{\alpha^{Syn}_{KD}}{2} d_{\mathcal{P}\Delta\mathcal{P}}(\psi \circ P^{Syn}, \psi \circ P^T) + \epsilon_{P^T_{KD}}(f^{Syn})) \\
\leq & \epsilon_{P^S_i}(M_i) + \alpha \mathbf{C}(P_i, P^T) + \alpha^{Syn} \epsilon_{P^T}(f^{Syn}) + \alpha^{Syn}_{KD} \epsilon_{P^T_{KD}}(f^{Syn}) \\
& + \frac{(1-\alpha)}{2} d_{\mathcal{P}\Delta\mathcal{P}}(\psi \circ P^{Syn}, \psi \circ P^T)
\end{aligned} \tag{12}$$

With the last condition coming from $\alpha + \alpha^{Syn} + \alpha^{Syn}_{KD} = 1$

$$\epsilon_{P^T}(M_i) \leq \epsilon_{P^S_i}(M_i) + \alpha \mathbf{C}(P_i, P^T) + \alpha^{Syn} \epsilon_{P^T}(f^{Syn}) + \alpha^{Syn}_{KD} \epsilon_{P^T}(f^{Syn}_{KD}) + \frac{(1-\alpha)}{2} d_{\mathcal{P}\Delta\mathcal{P}}(\psi \circ P^{Syn}, \psi \circ P^T) \tag{13}$$

□

A.3. Interpretation for Theorem 1

From Eq. (9), it can be seen that the generalization bound for M_i consists of five terms.

- The first term $\epsilon_{P^S_i}(M_i)$ is the error bound with respect to the training source data distribution. With Claim 1 in appendix, minimizing this term is equivalent to optimizing the loss $\alpha \mathbb{E}_{(\mathbf{x},y)\sim P_i} \mathcal{L}_{\text{CE}} + \alpha^{Syn} \mathbb{E}_{(\mathbf{x},y)\sim P^{Syn}} \mathcal{L}_{\text{CE}} + \alpha^{Syn}_{\text{KD}} \mathbb{E}_{(\mathbf{x},y)\sim P^{Syn}} \mathcal{L}_{\text{KD}}$. Since this is the form of our loss function in Eq. 7, we expect this term to be minimized
- The second term is inherited from the original generalization bound in (Ben-David et al., 2010) with the local training data. For our case, it can be controlled by the component weight α. If we rely less on the local data (i.e. smaller α), then these terms will be vanishing. Moreover even if we rely more on local data, it is essentially a distance measure between the local client distribution P_i and the global data distribution P^T. Since the global data distribution is an average of the closely related local client distributions, we expect this term to be small (Tang et al., 2022), (Ben-David et al., 2010), (Albuquerque et al., 2019).
- The third term measures the discrepancy between real labeling and the synthetic data labeling mechanisms. This discrepancy will be low because of our synthetic data generation process. Note that the data distillation's objective is to achieve $\mathbb{E}_{\mathbf{x}\sim P^T}[l(M^T(\mathbf{x}), y)] \simeq \mathbb{E}_{\mathbf{x}\sim P^{Syn}}[l(M^{Syn}(\mathbf{x}), y)]$ (Eq. 1 in (Zhao & Bilen, 2023)). If we change the M to a well-trained deep NN, then it's easy to see the synthetic data labelling f^{syn} will be similar to the real labelling f_T. Here we leverage the distribution matching that uses MMD loss to minimize the embedding differences between the synthetic data and real data in the same class (Zhao & Bilen, 2023) as a proxy way to achieve that.
- The fourth term originates from the knowledge distillation loss in equation 6. Here, we use the consensus knowledge from neighbour models to improve the local model. The labelling function of the extended KD data f^{syn}_{KD}, changes as training continues and the neighbour models learn to generalize well. Towards the end of training, predictions from the consensus knowledge should match the predictions of the true labeling function, therefore, f^{syn}_{KD} will be close to f_T.
- The fifth term is a distribution divergence between the encoded distributions of P^{Syn} and P^T. This is minimized by the domain invariant regularizer in Eq. 4, which acts as an anchor to pull all the encoded distributions together.

Remark: In order to get a tight generalization guarantee, we only need one of the fourth or fifth terms to be small. Since, if either any one of them is small, we can adjust the component weights α (practically λ_{REG} and λ_{KD}) to get a better generalization guarantee.

A.4. Proof for Proposition 2

Proof. Without loss of generality, let's start with

$$\sup_{M\in\mathcal{M}_i} |\epsilon_{P^{Syn}}(M) - \epsilon_{P^T}(M)| + \epsilon_{P^T}(f^{Syn}) \leq \inf_{M\in\mathcal{M}_i} (\epsilon_{P_i}(M) - \epsilon_{P^T}(M)) + \underbrace{\frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) + \lambda(P_i)}_{\mathbf{C}(P_i, P^T)}. \tag{14}$$

Then it holds that for any $M \in \mathcal{M}_i$,

$$\begin{aligned} \epsilon_{P^{Syn}}(M) - \epsilon_{P^T}(M) + \epsilon_{P^T}(f^{Syn}) &\leq \epsilon_{P_i}(M) - \epsilon_{P^T}(M) + \mathbf{C}(P_i, P^T) \\ \Rightarrow \quad \epsilon_{P^{Syn}}(M) + \epsilon_{P^T}(f^{Syn}) &\leq \epsilon_{P_i}(M) + \mathbf{C}(P_i, P^T) \end{aligned} \tag{15}$$

Note that the right side of (15) is the original bound in Theorem 2 in (Ben-David et al., 2010). Similarly, we can achieve

$$\epsilon_{P^{Syn}_{KD}}(M) + \epsilon_{P^T}(f^{Syn}_{KD}) \leq \epsilon_{P_i}(M) + \frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) + \lambda(P_i) \tag{16}$$

Combining (15-16) together with the component weights $\boldsymbol{\alpha}$ and setting $\alpha \to 0$,

$$\alpha^{Syn}_{KD} \epsilon_{P^{Syn}_{KD}}(M) + \alpha^{Syn} \epsilon_{P^{Syn}}(M) + \epsilon_{P^T}(f^{Syn}_{KD}) \leq \epsilon_{P_i}(M) + \frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) + \lambda(P_i) \tag{17}$$

Therefore, we conclude that our global generalization bound in Theorem 1 (which is the LHS of Eq. 17) is tighter than the original bound in (Ben-David et al., 2010) (RHS), when the condition of Proposition 2 holds.

□

A.5. Some useful lemmas and claims

Lemma 1. *Denote the model as $M = \rho \circ \psi \in \mathcal{M}$. The global generalization bound holds as*

$$\epsilon_{P^T}(M) \leq \epsilon_P(M) + \frac{1}{2} d_{\mathcal{P}\Delta\mathcal{P}}(\psi \circ P, \psi \circ P^T) + \epsilon_{P^T}(f), \tag{18}$$

where (P, f) could be either (P^{Syn}, f^{Syn}) or $(P^{Syn}_{KD}, f^{Syn}_{KD})$ pair.

Proof. For any model $M = \rho \circ \psi \in \mathcal{M}$, we have the following bound for the global virtual data distribution:

$$\begin{aligned} \epsilon_{P^T}(M) - \epsilon_{P^{Syn}}(M) &\overset{(a)}{=} \epsilon_{P^T}(M, f^T) - \epsilon_{P^{Syn}}(M, f^{Syn}) \\ &\overset{(b)}{\leq} |\epsilon_{P^T}(M, f^{Syn}) + \epsilon_{P^T}(f^{Syn}, f^T) - \epsilon_{P^{Syn}}(M, f^{Syn})| \\ &\leq |\epsilon_{P^T}(M, f^{Syn}) - \epsilon_{P^{Syn}}(M, f^{Syn})| + \epsilon_{P^T}(f^{Syn}) \\ &= |\epsilon_{P^T}(\rho \circ \psi, f^{Syn}) - \epsilon_{P^{Syn}}(\rho \circ \psi, f^{Syn})| + \epsilon_{P^T}(f^{Syn}) \\ &= |\epsilon_{\psi \circ P^T}(\rho, f^{Syn} \circ \psi^{-1}) - \epsilon_{\psi \circ P^{Syn}}(\rho, f^{Syn} \circ \psi^{-1})| + \epsilon_{P^T}(f^{Syn}) \\ &\leq \sup_{\rho, \rho'} |\epsilon_{\psi \circ P^T}(\rho, \rho') - \epsilon_{\psi \circ P^{Syn}}(\rho, \rho')| + \epsilon_{P^T}(f^{Syn}) \\ &\leq \frac{1}{2} d_{\mathcal{P}\Delta\mathcal{P}}(\psi \circ P^{Syn}, \psi \circ P^T) + \epsilon_{P^T}(f^{Syn}) \end{aligned} \tag{19}$$

where (a) is by definitions and (b) relies on the triangle inequality for classification error (Ben-David et al., 2006; Crammer et al., 2008). Thus, we have that

$$\epsilon_{P^T}(M) \leq \epsilon_{P^{Syn}}(M) + \frac{1}{2} d_{\mathcal{P}\Delta\mathcal{P}}(\psi \circ P^{Syn}, \psi \circ P^T) + \epsilon_{P^T}(f^{Syn}). \tag{20}$$

Similarly, as the the extended KD dataset shares the same feature distribution with the global virtual dataset, thus the above bound also holds for f^{Syn}_{KD}. □

Lemma 2 (Appendix A (Feng et al., 2021))**.** *For the extended source domain* $(\mathbf{x}^{Syn}, \hat{y}^{Syn}) \sim \hat{P}^{Syn}$*, training the related model with the knowledge distillation loss* $L_{KD} = D_{KD}(\hat{y}^{Syn} \| h(\mathbf{x}))$ *equals to optimizing the task risk* $\epsilon_{\hat{P}^{Syn}} = \mathbb{P}_{(\mathbf{x}^{Syn}, \hat{y}^{Syn}) \sim \hat{P}^{Syn}}[h(\mathbf{x}) \neq \arg\max \hat{y}^{Syn}]$.

Claim 1. *With the training source data at* i*th client as* P_i^S *with the component weight* $\boldsymbol{\alpha} = [\alpha, \alpha^{Syn}, \alpha_{KD}^{Syn}]^\top$ *on the local data, virtual data and extended KD data,* $\epsilon_{P_i^S}(h)$ *is minimized by optimizing the loss:*

$$\min_{M \in \mathcal{M}} \alpha \mathbb{E}_{(\mathbf{x},y)\sim P_i} L_{CE}(y, M(\mathbf{x})) + \alpha^{Syn} \mathbb{E}_{(\mathbf{x},y)\sim P^{Syn}} L_{CE}(y, M(\mathbf{x})) + \alpha_{KD}^{Syn} \mathbb{E}_{(\mathbf{x},y)\sim P_{KD}^{Syn}} L_{KL}(y \| M(\mathbf{x})) \tag{21}$$

Proof. Note that

$$\begin{aligned}
&\min_{M \in \mathcal{M}} \mathbb{E}_{(\mathbf{x},y)\sim P_i^{(S)}} L_{KL}(y \| M(\mathbf{x})) \\
&\propto \min_{M \in \mathcal{M}} \alpha \mathbb{E}_{(\mathbf{x},y)\sim P_i} L_{KL}(y \| M(\mathbf{x})) + \alpha^{Syn} \mathbb{E}_{(\mathbf{x},y)\sim P^{Syn}} L_{KL}(y \| M(\mathbf{x})) + \alpha_{KD}^{Syn} \mathbb{E}_{(\mathbf{x},y)\sim P_{KD}^{Syn}} L_{KL}(y \| M(\mathbf{x})) \\
&\overset{(a)}{\propto} \min_{M \in \mathcal{M}} \alpha \mathbb{E}_{(\mathbf{x},y)\sim P_i} L_{CE}(y, M(\mathbf{x})) + \alpha^{Syn} \mathbb{E}_{(\mathbf{x},y)\sim P^{Syn}} L_{CE}(y, M(\mathbf{x})) + \alpha_{KD}^{Syn} \mathbb{E}_{(\mathbf{x},y)\sim P_{KD}^{Syn}} L_{KL}(y \| M(\mathbf{x}))
\end{aligned}$$

where (a) is because $L_{KL}(y \| h(\mathbf{x})) = L_{CE}(y, h(\mathbf{x})) - H(y)$, where $H(y) = -y \log(y)$ is a constant depending on data distribution. With Lemma 2 and Pinsker's inequality, it is easy to show that $\epsilon_{P_i^S}(h)$ is minimized by optimizing the above loss. □

B. Privacy Discussion for DESA

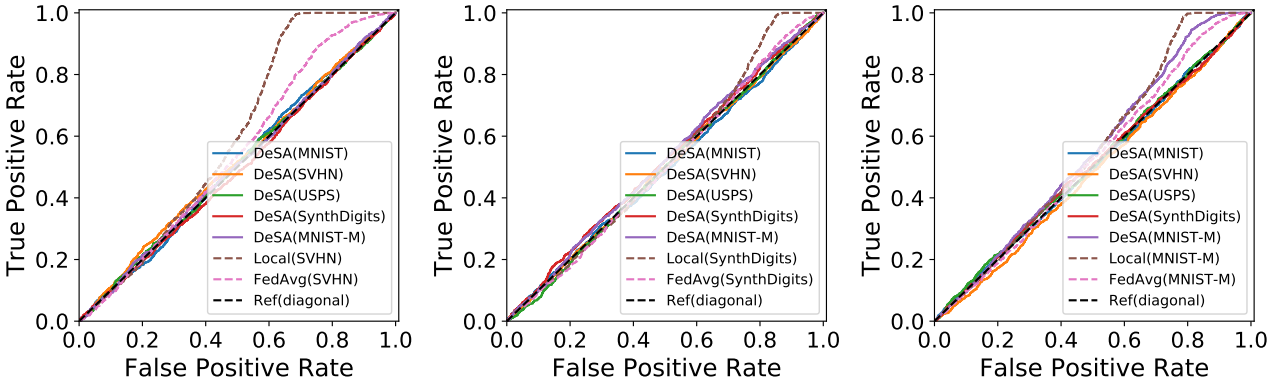
Sharing image-level information among clients may raise privacy concerns. However, we claim that decentralized FL with both data and model heterogeneities is an extremely challenging setting, where existing solutions either require sharing real public data (Lin et al., 2020; Huang et al., 2022) or synthetic data generated from real data with GAN-based generator (Zhang et al., 2022a;b). Instead, we propose to use distribution matching to distill data, a simple and less data-greedy strategy, for data synthesis. Research has shown that using distilled data can defend against privacy attacks (Dong et al., 2022) such as membership inference attacks (MIA) (Shokri et al., 2017) and gradient inversion attacks (Huang et al., 2021b). We show the DESA's defense against MIA (Carlini et al., 2022a) in Appendix B.1. In addition, recent papers have successfully applied differential privacy (DP) (Abadi et al., 2016) mechanism into data distillation (Xiong et al., 2023; Wang et al., 2023) to ensure privacy. We also discuss the feasibility of adding DP into data distillation process following (Xiong et al., 2023) and show that DESA is still effective using the DP synthetic anchor data in Appendix B.2. We would like to put more emphasis on our proposed methodology and theoretical analysis in the main text, as sharing synthetic data commonly exists in the related work mentioned above and we fairly align with their settings in our comparisons. Thus, we consider the potential privacy risk of FL with DESA is beyond the main scope of our study and leave it in our Appendix.

B.1. Membership Inference Attack

We show what under the basic setting of DESA (*i.e.*, not applying Differential Privacy when generating local synthetic data), we can better protect the membership information of local real data than local training or FedAvg (McMahan et al., 2017) on local real data only when facing Membership Inference Attack (MIA) on trained local models. Although we share the logits during communication, it's important to note that these logits are from synthetic anchor data and not real data that needs protection. Therefore, we cannot use MIA methods that rely on logits. Instead, we perform a strong MIA attack recently proposed and evaluate it following the approach in (Carlini et al., 2022a).

The goal of the experiment is to investigate whether our local model is vulnerable to MIA, namely leaking information about local real datasets' membership. To compare and demonstrate the effectiveness of the chosen attack, we also present results from local training and FedAvg training. We conduct MIA experiments using `DIGITS`. The MIA for local training and FedAvg is related to real local training data. Since we use synthetic anchor data generated from other clients with data distillation, we also provide MIA results for inferring real data of other clients. For example, if attacking SVHN's local model, local training and FeAvg report the MIA results on SVHN only, while we also report MIA results on MNIST, USPS, SynthDigits, MNIST-M for DESA.

Using the metric in (Carlini et al., 2022a), the results are shown in Figure 4. The Ref(diagonal) line indicates MIA **cannot** tell the differences between training and testing data. If the line bends towards True Positive Rate, it means the membership



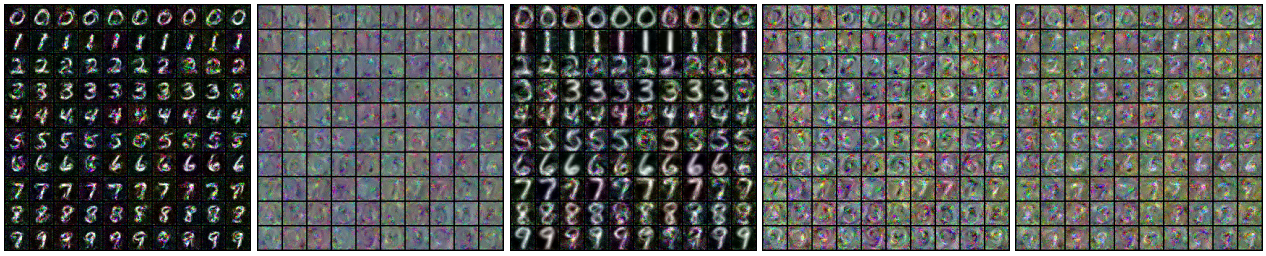
(a) SVHN (b) SynthDigits (c) MNIST-M

Figure 4: MIA on the models trained by SVHN, SynthDigits, and MNIST-M clients. Observe that the synthetic data sharing of DeSA does not reveal other clients' local data identity information.

form the training set can be inferred. It is shown that all the MIA curves of targeted and other cients lie along the Ref line for DeSA's model, which indicates that the membership of each training sets is well protected given the applied attack. While the curves for the MIA attacks on FedAvg and local training with SVHN dataset are all offset the Ref (diagonal) line towards True Positive, indicating they are more vulnerable to MIA and leaking training data information.

B.2. Differential Privacy for Data Synthesis

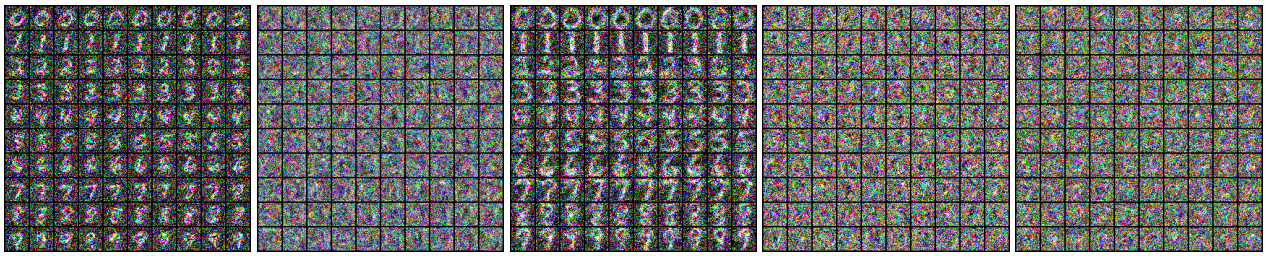
To enhance the data privacy-preservation on shared synthetic anchor data, we apply the Differential Privacy stochastic gradient descent (DP-SGD) (Abadi et al., 2016) for the synthetic image generation. DP-SGD protects local data information via noise injection on clipped gradients. In our experiments, we apply Gaussian Mechanism for the inejcted noise. Specifically, we first sample a class-balanced subset from the raw data to train the objective 3. We set up the batch size as 256. For each iteration, we clip the gradient so that its l_2-norm is 2. The injected noises are from $\mathcal{N}(0, 1.2)$. This step ensures (ϵ, δ)-DP with (ϵ, δ) values in $\{$(7.622, 0.00015), (10.3605, 0.00021), (8.6677, 0.00017), (7.3174, 0.00014), (7.6221, 0.00015)$\}$ guarantees for $\{$MNIST, SVHN, USPS, SynthDigits, MNIST-M$\}$, respectively. We visualize the non-DP and DP synthetic images from each clients in `DIGITS` in Figure 5 and Figure 6, respectively. One can observe that the synthetic data with DP mechanism is noisy and hard to inspect the individual information of the raw data.



(a) Mnist (b) SVHN (c) USPS (d) SynthDigits (e) Mnist-M

Figure 5: Visualization of the global and local synthetic images from the `DIGITS` dataset. (a) visualized the MNIST client; (b) visualized the SVHN client; (c) visualized the USPS client; (d) visualized the SynthDigits client; (e) visualized the MNIST-M client; (f) visualized the server synthetic data.

We replace the synthetic data by DP synthetic data and perform `DIGITS` experiments, and the result is shown in Table 6. It can be observed that although DeSA's performance slightly drops due to the DP mechanism, the averaged inter and intra-accuracy are in the second place, which indicates that DeSA is robust as long as we can synthesize images that roughly captures the global data distribution.



(a) Mnist (b) SVHN (c) USPS (d) SynthDigits (e) Mnist-M

Figure 6: Visualization of the global and local synthetic images from the DIGITS dataset with **DP mechanism**. (a) visualized the MNIST client; (b) visualized the SVHN client; (c) visualized the USPS client; (d) visualized the SynthDigits client; (e) visualized the MNIST-M client; (f) visualized the server synthetic data.

Table 6: We add the the results for DeSA trained with DP synthetic anchor data into our Table 2. The best result is marked as **bold**, and the second best is marked as blue. The table shows that DeSA with DP synthetic anchor data can still obtain comparable results as DeSA with non-DP synthetic data.

		DIGITS					
		MN(C)	SV(A)	US(C)	Syn(A)	MM(C)	Avg
FedHe		59.51	66.67	49.89	75.39	71.57	64.81
FedDF	Cifar100	65.98	65.21	61.30	69.65	**74.48**	67.32
	FMNIST	43.05	69.14	44.95	74.67	71.27	60.61
FCCL	CIFAR100	-	-	-	-	-	-
	FMNIST	46.43	61.02	42.64	63.05	66.39	55.91
FedProto		62.59	71.74	58.52	**81.19**	74.44	69.70
DeSA(D_{VHL}^{Syn})		54.40	62.03	42.34	67.75	73.03	59.91
DeSA		**70.12**	**76.17**	**71.17**	81.10	73.83	**74.47**
DeSA(DP)		69.06	71.54	63.92	78.93	73.16	71.12

C. Local Epoch

Here we present the effect of local epochs on DeSA. To ensure fair comparison, we fix the total training iterations for the three experiments, *i.e.*,we set FL communication rounds to 50 when local epochs is 2 to match up with local epoch equals to 1 when FL communication rounds is 100. Figure 7 shows that DeSA is robust to various local epoch selections. The experiment is run on DIGITS dataset, and we report the global accuracy.

D. Communication Overhead

As noted in Section 3.1, DeSA only requires sharing logits w.r.t. Global synthetic data during training. Thus it has a relatively low communication overhead compared to baseline methods which require sharing model parameters. For fair comparison, we analyze the communication cost based on the number of parameters Pre-FL and During-FL in Table 8. Note that we show the number of parameters for one communication round for During-FL, and the total communication cost depends on the number of global iterations. One can observe that sharing logits can largely reduce the communication overhead. For example, if we use ConvNet as our model, set IPC=50, and train for 100 global iteration, the total number of parameters for communication for DeSA will be 30.7 K $\times$ 50 (Pre-FL) + 10 (number of classes) $\times$ 50 (images/class)

Table 7: Ablation study on number of local epochs. The experiment is run on DIGITS dataset.

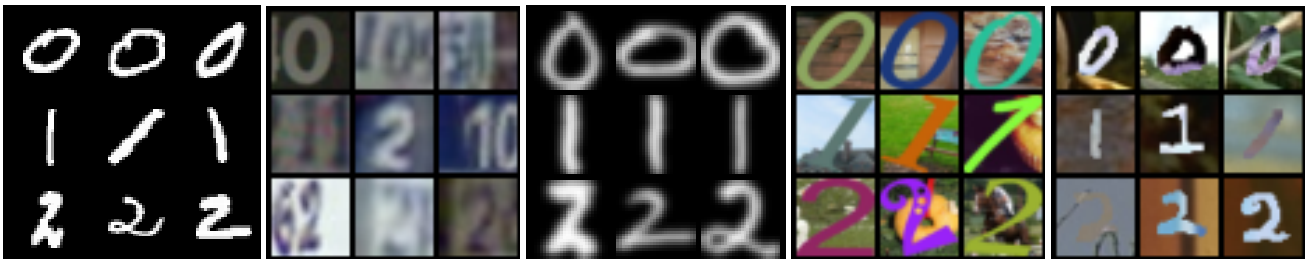
Local Epoch	1	2	5
Global Acc	74.47	74.15	74.34

$\times$ 10 (logits/image) $\times$ 100 (global iteration) = 2.04M. In comparison, baseline methods need to share 0 (Pre-FL) + 320K (parameters/iteration) $\times$ 100 (global iteration) = 32M, which is much larger than DeSA. Under model heterogeneity experimental setting, clients using AlexNet would suffer even higher total communication cost, which is 0 (Pre-FL) + 1.87M (parameters/iteration) $\times$ 100 (global iteration) = 187M.

Table 8: Comparison of communication overhead. Note that for DESA, we only share virtual global anchor logits during training. The total communication cost counts the total parameter transferred for 100 global iterations. IPC is the synthesized images per class, and C is the number of classes.

	ConvNet	AlexNet	Global Anchor Logits
Pre-FL	0	0	30.7 K $\times$ IPC $\times$ C
During-FL	320 K	1.87 M	100 $\times$ IPC $\times$ C
Total	32M	187M	40.7K $\times$ IPC $\times$ C

E. Datasets and Synthetic Images



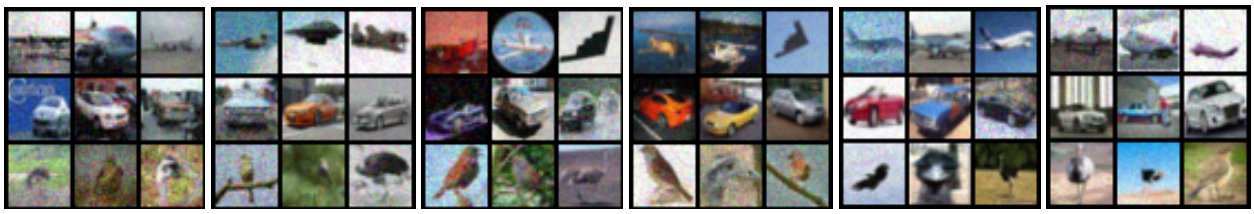
(a) (b) (c) (d) (e)

Figure 7: Visualization of the original digits dataset. (a) visualized the MNIST client; (b) visualized the SVHN client; (c) visualized the USPS client; (d) visualized the SynthDigits client; (e) visualized the MNIST-M client.

(a) Amazon (b) Caltech (c) DSLR (d) Webcam

Figure 8: Visualization of the original digits dataset. (a) visualized the Amazon client; (b) visualized the Caltech client; (c) visualized the DSLR client; (d) visualized the Webcam client

Detailed Information of Selected Datasets 1) `DIGITS`={MNIST (LeCun et al., 1998), SVHN (Netzer et al., 2011), USPS (Hull, 1994), SynthDigits (Ganin & Lempitsky, 2015), MNIST-M (Ganin & Lempitsky, 2015)} consists of 5 digit datasets with handwritten, real street and synthetic digit images of $0, 1, \cdots, 9$. Thus, we assume 5 clients for this set of experiments. We use `DIGITS` as baseline to show DESA can handle FL under large domain shift. Example images can be found in Figure 7.
2) `OFFICE`={Amazon (Saenko et al., 2010), Caltech (Griffin et al., 2007), DSLR (Saenko et al., 2010), and WebCam (Saenko



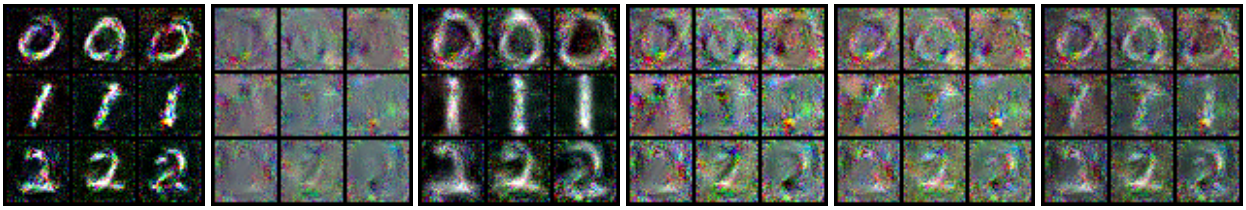
(a) (b) (c) (d) (e) (f)

Figure 9: Visualization of the original CIFAR10C. Sampled images from the first six clients.

et al., 2010)} consists of four data sources from Office-31 (Saenko et al., 2010) (Amazon, DSLR, and WebCam) and Caltech-256 (Griffin et al., 2007) (Caltech), resulting in four clients. Each client possesses images that were taken using various camera devices in different real-world environments, each featuring diverse backgrounds. We show DESA can handle FL under large domain shifted real-world images using OFFICE. Example images can be found in Figure 8.
3) CIFAR10C consists subsets extracted from Cifar10-C (Hendrycks & Dietterich, 2019), a collection of augmented Cifar10 (Krizhevsky et al., 2009) that applies 19 different corruptions. We employ a Dirichlet distribution with $\beta = 2$ for the purpose of generating three partitions within each distorted non-IID dataset. As a result, we have 57 clients with domain- and label-shifted datasets. Example images can be found in Figure 9.

Synthetic Data Generation We fix ConvNet as the backbone for data synthesis to avoid additional domain shift caused by different model architectures. We set learning rate to 1 and use SGD optimizer with momentum = 0.5. The batch size for DIGITS and CIFAR10 is set to 256, while we use 32 for OFFICE as it's clients has fewer data points. For the same reason, we use 500 synthetic data points for DIGITS and CIFAR10C, and we set 100 synthetic data points for OFFICE. The training iteration for DIGITS and OFFICE is 1000, and we set 2000 for CIFAR10C since it contains more complex images.

We show the local synthetic images and global anchor images of DIGITS, OFFICE, and CIFAR10C in Figure 10, Figure 11, and Figure 12, respectively.

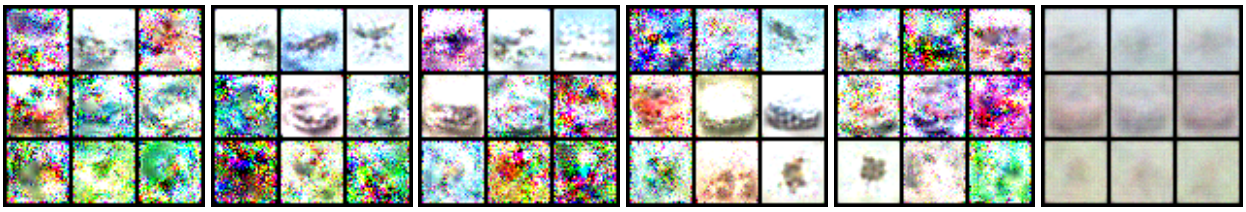


(a) Mnist (b) SVHN (c) USPS (d) SynthDigits (e) Mnist-M (f) Average

Figure 10: Visualization of the sampled global and local synthetic images from the DIGITS dataset. (a) visualized the MNIST client's synthetic data; (b) visualized the SVHN client's synthetic data; (c) visualized the USPS client's synthetic data; (d) visualized the SynthDigits client's synthetic data; (e) visualized the MNIST-M client's synthetic data; (f) visualized the server synthetic data.

(a) Amazon (b) Caltech (c) DSLR (d) Webcam (e) Average

Figure 11: Visualization of the sampled global and local synthetic images from the `OFFICE` dataset. (a) visualized the Amazon client's synthetic data; (b) visualized the Caltech client's synthetic data; (c) visualized the DSLR client's synthetic data; (d) visualized the Webcam client's synthetic data; (e) visualized the averaged synthetic data.



(a) Client0 (b) Client1 (c) Client2 (d) Client3 (e) Client4 (f) Average

Figure 12: Visualization of the sampled global and local synthetic images from the first 5 clients in `CIFAR10C` dataset. (a) visualized the first 's synthetic data; (b) visualized the second client's synthetic data; (c) visualized the third client's synthetic data; (d) visualized the forth client's synthetic data; (e) visualized the fifth client's synthetic data; (f) visualized the server synthetic data.

F. Model Architectures

We use ConvNet to perform data distillation for the best synthesis quality. For model hetero genenity scenarios, we randomly select classification model architectures from {AlexNet, ConvNet}. The detailed setup for bot models are depicted in Table 9 and Table 10

Table 9: AlexNet architecture. For the convolutional layer (Conv2D), we list parameters with a sequence of input and output dimensions, kernel size, stride, and padding. For the max pooling layer (MaxPool2D), we list kernel and stride. For a fully connected layer (FC), we list input and output dimensions.

Layer	Details
1	Conv2D(3, 128, 5, 1, 4), ReLU, MaxPoo2D(2, 2)
2	Conv2D(128, 192, 5, 1, 2), ReLU, MaxPoo2D(2, 2)
3	Conv2D(192, 256, 3, 1, 1), ReLU
4	Conv2D(256, 192, 3, 1, 1), ReLU
5	Conv2D(192, 192, 3, 1, 1), ReLU, MaxPoo2D(2, 2)
22	FC(3072, num_class)

G. More Related Work

G.1. Model Homogeneous Federated Learning

We list down different Model Homogeneous FL approaches in decentralized FL and collaborative methods that are relevant to our setting.

Table 10: ConvNet architecture. For the convolutional layer (Conv2D), we list parameters with a sequence of input and output dimensions, kernel size, stride, and padding. For the max pooling layer (MaxPool2D), we list kernel and stride. For a fully connected layer (FC), we list the input and output dimensions. For the GroupNormalization layer (GN), we list the channel dimension.

Layer	Details
1	Conv2D(3, 128, 3, 1, 1), GN(128), ReLU, AvgPool2d(2,2,0)
2	Conv2D(128, 118, 3, 1, 1), GN(128), ReLU, AvgPool2d(2,2,0)
3	Conv2D(128, 128, 3, 1, 1), GN(128), ReLU, AvgPool2d(2,2,0)
4	FC(1152, num_class)

G.1.1. Decentralized Federated Learning

In order to tackle training a global model without a server, Decentralized FL methods communicate a set of models through diverse decentralized client-network topologies (such as a ring - (Chang et al., 2018), Mesh - (Roy et al., 2019), or a sequential line (Assran et al., 2019)) using different communication protocols such as Single-peer(gossip) or Multiple-Peer(Broadcast). (Yuan et al., 2023a; Sheller et al., 2019; 2020) pass a single model from client to client similar to an Incremental Learning setup. In this continual setting, only a **single model** is trained. (Pappas et al., 2021; Roy et al., 2019; Assran et al., 2019) pass models and aggregate their weights similar to conventional FL. Since these models use averaged aggregation techniques similar to FedAvg, most of these methods assume client **model homogeneity**. DeSA's client network topology is similar to that of a Mesh using the broadcast-gossip protocol, where every client samples certain neighbours in each communication round for sharing logits.

None of the works above aim to train various client model types without a server, which is our goal.

G.1.2. Collaborative Methods

(Fallah et al., 2020) uses an MAML(model agnostic meta learning) framework to explicitly train model homogeneous client models to personalize well. The objective function of MAML evaluates the personalized performance assuming a one-step gradient descent update on the subsequent task. (Huang et al., 2021a) modifies the personalized objective by adding an attention inducing term to the objective function which promotes collaboration between pairs of clients that have similar data.

(Ghosh et al., 2022) captures settings where different groups of users have their own objectives (learning tasks) but by aggregating their private data with others in the same cluster (same learning task), they can leverage the strength in numbers in order to perform more efficient personalized federated learning (Donahue & Kleinberg, 2021) uses game theory to analyze whether a client should jointly train with other clients in a conventional FL setup [2.1] assuming it's primary objective is to minimize the MSE loss on its own private dataset. They also find techniques where it is more beneficial for the clients to create coalitions and train one global model.

All the above works either slightly change the intra-client objective to enable some collaboration between model-homogeneous clients or explicitly create client clusters to collaboratively learn from each other. They do not tackle the general objective function that we do- 2

G.2. Model Heterogeneous Federated Learning

Model heterogeneous FL approaches relevant to DeSA broadly come under the following two types.

G.2.1. Knowledge distillation methods

(Gong et al., 2022) proposes FedKD that is a one-shot centralized Knowledge distillation approach on unlabelled public data after the local training stage in-order to mitigate the accuracy drop due to the label shift amongst clients. DENSE (Zhang et al., 2022a) propose one-shot federated learning to generate decision boundary-aware synthetic data and train the global model on the server side. FedFTG (Zhang et al., 2022b) finetunes the global model by knowledge distillation with hard sample mining. (Yang et al., 2021) introduces a method called Personalized Federated Mutual Learning (PFML), which

leverages the non-IID properties to create customized models for individual parties. PFML incorporates mutual learning into the local update process within each party, enhancing both the global model and personalized local models. Furthermore, mutual distillation is employed to expedite convergence. The method assumes homogeneity of models for global server aggregation. However, all the above methods are centralized.

G.2.2. Mutual Learning Methods

Papers in this area predominantly use ideas from deep-mutual learning (Zhang et al., 2018) (Matsuda et al., 2022) uses deep mutual learning to train heterogeneous local models for the sole purpose of personalization. The method creates clusters of clients whose local models have similar outputs. Clients within a cluster exchange their local models in-order to tackle label shift amongst the data points. However, the method is centralized and each client maintains two copies of models, one which is personalized and one that is exchanged. (Li et al., 2021a) has a similar setting to (Chan & Ngai, 2021), but instead solves the problem in a peer to peer decentralized manner using soft logit predictions on the local data of a client itself. It makes its own baselines that assume model homogeneity amongst clients, also their technique assumes that there is no covariate shift because it only uses local data for the soft predictions. However, their technique can be modified for model heterogeneity. They report personalization(Intra) accuracies only.

G.3. Dataset Distillation

Data distillation methods aim to create concise data summaries D_{syn} that can effectively substitute the original dataset D in tasks such as model training, inference, and architecture search. Moreover, recent studies have justified that data distillation also preserves privacy (Dong et al., 2022; Carlini et al., 2022b) which is critical in federated learning. In practice, dataset distillation is used in healthcare for medical data sharing for privacy protection (Li et al., 2022). We briefly mention two types of Distillation works below.

G.3.1. Gradient and Trajectory Matching techniques

Gradient Matching (Zhao et al., 2020) is proposed to make the deep neural network produce similar gradients for both the terse synthetic images and the original large-scale dataset. The objective function involves matching the gradients of the loss w.r.t weights(parameters) evaluated on both D and D_{syn} at successive parameter values during the optimization on the original dataset D. Usually the cosine distance is used to measure the difference in gradient direction. Other works in this area modify the objective function slightly, by either adding class contrastive signals for better stability (Lee et al., 2022) or by adding same image-augmentations(such as crop, rotate to both D and D_{syn})(Zhao & Bilen, 2021). A similar technique is that of (Cazenavette et al., 2022) which tries to match the intermediate parameters in the optimization trajectory of both D and D_{Syn}. It is very *computationally expensive* because of a gradient unrolling in the optimization. TESLA (Cui et al., 2023) attempts at using linear-algebraic manipulations to give better computational guarantees for Trajectory matching

G.3.2. Distribution Matching techniques

Distribution matching (Zhao & Bilen, 2023) solves the distillation task via a single-level optimization, leading to a *vastly improved scalability*. More specifically, instead of matching the quality of models on D vs. D_{syn}, distribution-matching techniques directly match the distribution of D vs. D_{syn} in a latent encoded space. See 3 for the objective function. CAFE (Wang et al., 2022) further refines the distribution-matching idea by solving a bilevel optimization problem for jointly optimizing a single encoder and the data summary, rather than using a pre-determined set of encoders Adversarial techniques using Distribution matching such as IT-GAN (Zhao & Bilen, 2022) and GAN (Goodfellow et al., 2014) aren't suitable for a serverless setting. Since we aim to mitigate drifts in client-distribution across using our synthetic data, Distribution Matching is a more natural option for our work.