Road Map of Appendix Our appendix is organized into five sections. The theoretical analysis and proof is in Appendix A. Appendix B shows the results for Membership Inference Attack (MIA) on DESA trained models using DIGITS datasets. Appendix C 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 D introduce the selected datasets and how we synthesize anchor data in detail. Appendix E describes the model architectures (ConvNet and AlexNet) we use in our experiments. Finally, Appendix F 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 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 first two groups are used to construct the cross-entropy loss and those from the third one is for the knowledge distillation loss. Without loss of generality, at ith client, we set the weight for P_i, P^{Syn} and P^{Syn}_{KD} as α, α^{Syn} and α^{Syn}_{KD}, respectively. For notation simplicity, we assume $\alpha + \alpha^{Syn} + \alpha^{Syn}_{KD} = 1$. Then the training source data at ith client is $P^S_i = \alpha P_i + \alpha^{Syn} P^{Syn} + \alpha^{Syn}_{KD} P^{Syn}_{KD}$.

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

$$\epsilon_{P^T}(M_i) \leq \epsilon_{P_i}(M_i) + \frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) + \lambda(P_i) \tag{10}$$

where $\frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) = \sup_{M, M' \in \mathcal{M}_i} |\mathbb{P}_{\mathbf{x} \sim P_i}[M(\mathbf{x}) \neq M'(\mathbf{x})] - \mathbb{P}_{\mathbf{x} \sim P^T}[M(\mathbf{x}) \neq M'(\mathbf{x})]|$ and $\lambda(P_i) = \min_{M \in \mathcal{M}_i} \epsilon_{P_i}(M) + \epsilon_{P^T}(M)$ is a constant. Then with (10) and Lemma 1, we have that

$$\begin{aligned}
\epsilon_{P^T}(M_i) \leq & \epsilon_{P^S_i}(M_i) + \frac{\alpha}{2} d_{\mathcal{H}\Delta\mathcal{H}}(P_i, P^T) + \alpha_i \lambda(P_i) + \alpha^{Syn} (\sup_{\rho, \rho'} |\epsilon_{\psi \circ P^T}(\rho, \rho') - \epsilon_{\psi \circ P^{Syn}}(\rho, \rho')| \\
& + \epsilon_{P^T}(f^{Syn})) + \alpha^{Syn}_{KD} (\sup_{\rho, \rho'} |\epsilon_{\psi \circ P^T}(\rho, \rho') - \epsilon_{\psi \circ P^{Syn}}(\rho, \rho')| + \epsilon_{P^T_{KD}}(f^{Syn})) \\
\leq & \epsilon_{P^S_i}(M_i) + \frac{\alpha}{2} d_{\mathcal{H}\Delta\mathcal{H}}(P_i, P^T) + \alpha \lambda(P_i) + \alpha^{Syn} \epsilon_{P^T}(f^{Syn}) + \alpha^{Syn}_{KD} \epsilon_{P^T_{KD}}(f^{Syn}) \\
& + (\alpha^{Syn} + \alpha^{Syn}_{KD}) \sup_{\rho, \rho'} |\epsilon_{\psi \circ P^T}(\rho, \rho') - \epsilon_{\psi \circ P^{Syn}}(\rho, \rho')|
\end{aligned} \tag{11}$$

With the condition that $\psi \circ P^{Syn} \to \psi \circ P^T$, the above bound can be simplified as

$$\epsilon_{P^T}(M_i) \leq \epsilon_{P^S_i}(M_i) + \frac{\alpha}{2} d_{\mathcal{H}\Delta\mathcal{H}}(P_i, P^T) + \alpha \lambda(P_i) + \alpha^{Syn} \epsilon_{P^T}(f^{Syn}) + \alpha^{Syn}_{KD} \epsilon_{P^T}(f^{Syn}_{KD}). \tag{12}$$

□

A.2 PROOF FOR PROPOSITION 2

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

$$\begin{aligned}
\sup_{M \in \mathcal{M}_i} |\epsilon_{P^{Syn}}(M) - \epsilon_{P^T}(M)| + \epsilon_{P^T}(f^{Syn}) \leq \\
\inf_{M \in \mathcal{M}_i} |\epsilon_{P_i}(M) - \epsilon_{P^T}(M)| + \frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) + \lambda(P_i).
\end{aligned} \tag{13}$$

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

$$\begin{aligned}
& \epsilon_{P^{Syn}}(M) - \epsilon_{P^T}(M) + \epsilon_{P^T}(f^{Syn}) \leq \epsilon_{P_i}(M) - \epsilon_{P^T}(M) + \frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) + \lambda(P_i) \\
\Rightarrow \quad & \epsilon_{P^{Syn}}(M) + \epsilon_{P^T}(f^{Syn}) \leq \epsilon_{P_i}(M) + \frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) + \lambda(P_i)
\end{aligned} \tag{14}$$

Note that the right side of (14) is the original bound in Theorem 2 in Ben-David et al. (2010). Similarly, we can achieve

$$\epsilon_{P_{KD}^{Syn}}(M) + \epsilon_{P^T}(f_{KD}^{Syn}) \leq \epsilon_{P_i}(M) + \frac{1}{2} d_{\mathcal{M}_i \Delta \mathcal{M}_i}(P_i, P^T) + \lambda(P_i) \tag{15}$$

Combining (14-15) together, we can conclude that our global generalization bound is tighter than the original bound. □

A.3 Some useful lemmas and claims

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

$$\epsilon_{P^T}(M) \leq \epsilon_P(M) + \sup_{\rho,\rho'} |\epsilon_{\psi \circ P^T}(\rho, \rho') - \epsilon_{\psi \circ P}(\rho, \rho')| + \epsilon_{P^T}(f), \tag{16}$$

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

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

$$\begin{aligned}
\epsilon_{P^T}(M) - \epsilon_{P^{Syn}}(M) &\overset{(a)}{=} \epsilon_{P^T}(M, f^T) - \epsilon_{P^{Syn}}(M, f^{Syn}) \\
&\overset{(b)}{\leq} |\epsilon_{P^T}(M, f^{Syn}) + \epsilon_{P^T}(f^{Syn}, f^T) - \epsilon_{P^{Syn}}(M, f^{Syn})| \\
&\leq |\epsilon_{P^T}(M, f^{Syn}) - \epsilon_{P^{Syn}}(M, f^{Syn})| + \epsilon_{P^T}(f^{Syn}) \\
&= |\epsilon_{P^T}(\rho \circ \psi, f^{Syn}) - \epsilon_{P^{Syn}}(\rho \circ \psi, f^{Syn})| + \epsilon_{P^T}(f^{Syn}) \\
&= |\epsilon_{\psi \circ P^T}(\rho, f^{Syn} \circ \psi^{-1}) - \epsilon_{\psi \circ P^{Syn}}(\rho, f^{Syn} \circ \psi^{-1})| + \epsilon_{P^T}(f^{Syn}) \\
&\leq \sup_{\rho,\rho'} |\epsilon_{\psi \circ P^T}(\rho, \rho') - \epsilon_{\psi \circ P^{Syn}}(\rho, \rho')| + \epsilon_{P^T}(f^{Syn})
\end{aligned} \tag{17}$$

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

$$\epsilon_{P^T}(M) \leq \epsilon_{P^{Syn}}(M) + \sup_{\rho,\rho'} |\epsilon_{\psi \circ P^T}(\rho, \rho') - \epsilon_{\psi \circ P^{Syn}}(\rho, \rho')| + \epsilon_{P^T}(f^{Syn}). \tag{18}$$

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_{KD}^{Syn}. □

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 ith 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:*

$$\min_{M \in \mathcal{M}} \alpha \mathbb{E}_{(\mathbf{x},y) \sim P_i} L_{CE}(y, M(\mathbf{x})) + \alpha^{Syn} \mathbb{E}_{(\mathbf{x},y) \sim P^{Syn}} L_{CE}(y, M(\mathbf{x})) + \alpha_{KD}^{Syn} \mathbb{E}_{(\mathbf{x},y) \sim P_{KD}^{Syn}} L_{KL}(y \| M(\mathbf{x})) \tag{19}$$

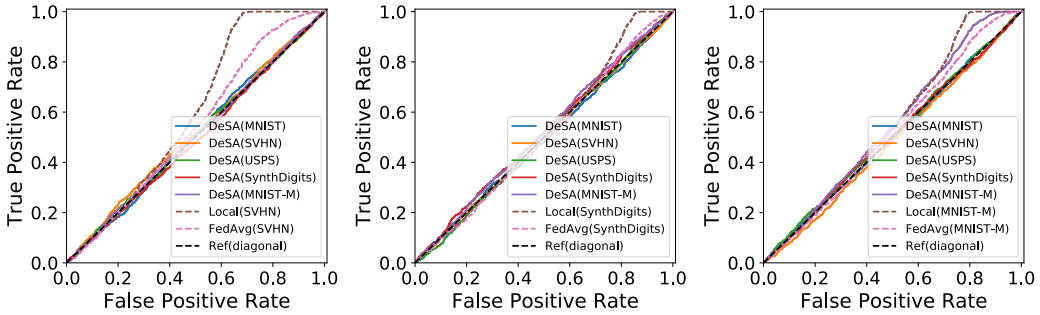
Proof. Note that

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

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 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).



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

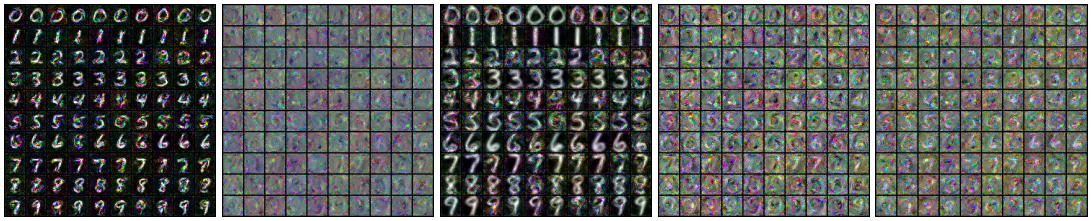
Figure 5: MIA on the models trained by SVHN, SynthDigits, and MNIST-M clients. Observe that the synthetic data sharing of DESA does not reveal other clients' local data identity information.

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 5. 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 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.

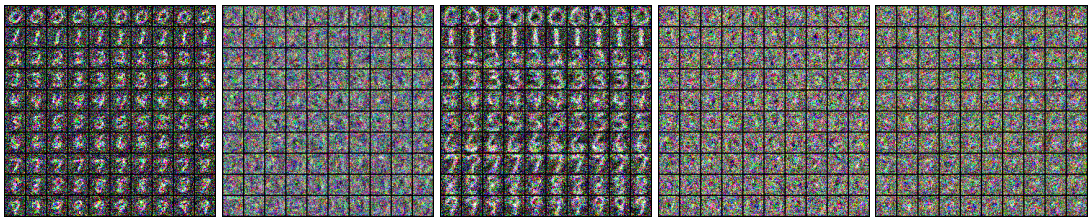
C DIFFERENTIAL PRIVACY FOR DATA SYNTHESIS

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 6 and Figure 7, 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 6: 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.



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

Figure 7: 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.

We replace the synthetic data by DP synthetic data and perform DIGITS experiments, and the result is shown in Table 3. 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.

Table 3: 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 has comparable results as DeSA with non-DP synthetic data.

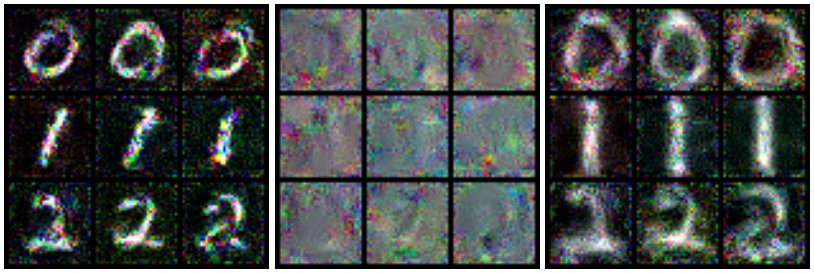
			DIGITS					
			MN(C)	SV(A)	US(C)	Syn(A)	MM(C)	Avg
FedHe		intra	**98.89**	83.33	98.76	93.89	94.31	93.84
		inter	49.67	62.50	37.67	70.77	65.89	57.30
FedDF	C	intra	92.29	20.73	86.18	15.19	51.57	53.19
		inter	34.85	12.39	18.28	17.21	37.28	24.00
	F	intra	92.59	19.56	93.33	77.86	69.83	70.63
		inter	39.03	18.21	30.66	54.02	54.26	39.24
FCCL	C	intra	11.25	19.10	13.66	10.08	11.25	13.07
		inter	13.52	11.56	12.92	13.82	13.52	13.07
	F	intra	95.01	72.92	94.73	86.11	90.10	87.78
		inter	34.29	58.04	29.62	57.28	60.46	47.94
DeSA(D_{VHL}^{Syn})		intra	98.69	80.09	98.66	90.68	94.40	92.50
		inter	43.33	57.51	28.26	62.02	67.69	51.76
DeSA		intra	98.78	**86.42**	**98.87**	**95.21**	**94.54**	**94.77**
		inter	**65.59**	**80.14**	**64.14**	**78.97**	**69.98**	**71.76**
DeSA(DP)		intra	98.80	86.15	98.55	94.42	94.21	**94.42**
		inter	62.17	74.45	55.09	76.64	69.01	67.47

D DATASETS AND SYNTHETIC IMAGES

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.
2) `OFFICE`={Amazon (Saenko et al., 2010), Caltech (Griffin et al., 2007), DSLR (Saenko et al., 2010), and WebCam (Saenko 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`.
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.

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, e 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. The numbers of synthetic data for clients are 500 for {`DIGITS`, `CIFAR10C`} and 100 for {`OFFICE`}, respectively.

We show the local synthetic images and global anchor images of `DIGITS`, `OFFICE`, and `CIFAR10C` in Figure 8, Figure 9, and Figure 10, respectively.

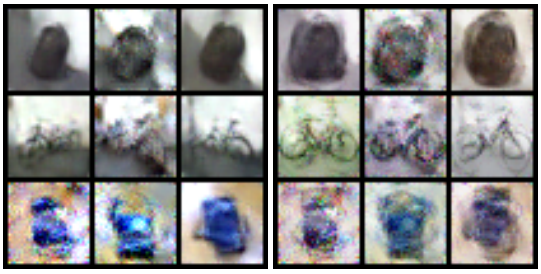


(a) Mnist (b) SVHN (c) USPS

(d) SynthDigits (e) Mnist-M (f) Average

Figure 8: Visualization of the sampled 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.

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



(d) Webcam (e) Average

Figure 9: Visualization of the sampled global and local synthetic images from the OFFICE dataset. (a) visualized the Amazon client; (b) visualized the Caltech client; (c) visualized the DSLR client; (d) visualized the Webcam client; (e) visualized the averaged synthetic data.

E 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}.

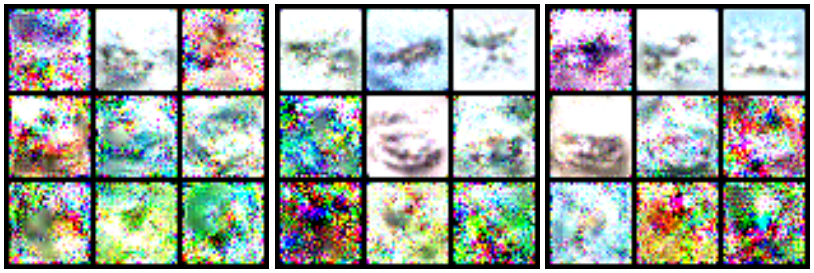
Table 4: AlexNet architecture. For the convolutional layer (Conv2D), we list parameters with a sequence of input and output dimensions, kernel size, stride, and padding. For the max pooling layer (MaxPool2D), we list kernel and stride. For a fully connected layer (FC), we list input and output dimensions.

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)

F MORE RELATED WORK

F.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.



(a) Client0 (b) Client1 (c) Client2

(d) Client3 (e) Client4 (f) Average

Figure 10: Visualization of the sampled global and local synthetic images from the first 5 clients in `CIFAR10C` dataset. (a) visualized the first client; (b) visualized the second client; (c) visualized the third client; (d) visualized the forth client; (e) visualized the fifth client; (f) visualized the server synthetic data.

Table 5: 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)

F.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.

F.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., 2021b) 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

F.2 MODEL HETEROGENEOUS FEDERATED LEARNING

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

F.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.

F.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.

F.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.

F.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

F.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.