Federated Learning in Non-IID Settings Aided by Differentially Private Synthetic Data
Supplementary Material

Huancheng Chen
Universitiy of Texas at Austin
Austin, Texas
huanchengch@utexas.edu

Haris Vikalo
Universitiy of Texas at Austin
Austin, Texas
hvikalo@ece.utexas.edu

1. Network Architecture

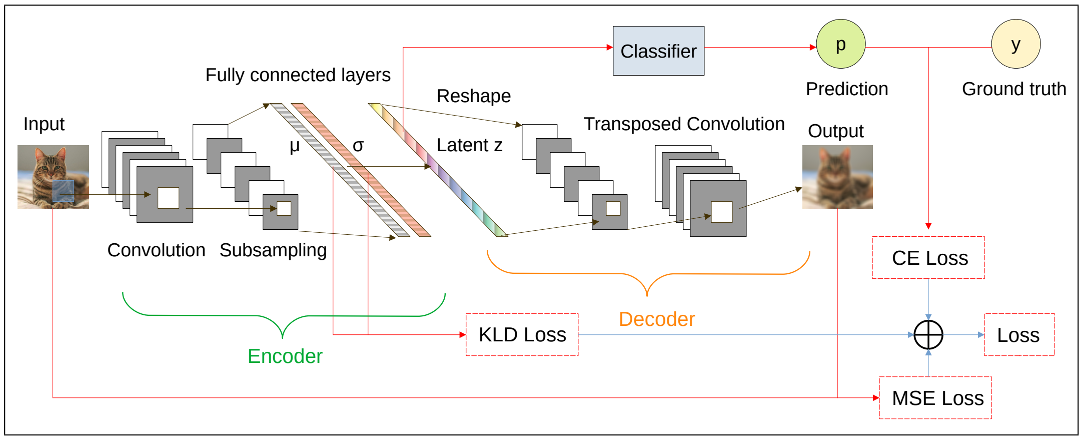


Figure 1. The architecture of the VAE. The KLD, MSE and CE stand for Kullback-Leibler divergence, mean-square error, and cross-entropy, respectively. The sum of the MSE loss and the KLD loss is weighted by the hyper-parameter λ and added to the CE loss.

For FMNIST, we use a CNN network with three 3×3 convolutional layers connected by two 2×2 max pooling layers as the base encoder. For CIFAR10 and CIFAR100, we use a CNN network with four 3×3 convolutional layers connected by two 2×2 max pooling layers as the base encoder. Both architectures utilize two fully connected layers. Moreover, we construct a decoder of a symmetric structure, composing a variational auto-encoder when implementing FedVAE and FedDPMS. We set the latent dimension l to 32 and 128 for FMNIST and CIFAR10/100, respectively. The details are illustrated in Tables 1 and 2:

2. Selection of Hyper-parameters

The hyper-parameter controlling the proximal term in FedProx is set to $\mu_{prox} = 0.001$. For Moon, we set the temperature parameter to 0.5, tune μ_{moon} across $\{1, 5, 10\}$, and report the best result. The mixup ratio parameter in FedMix was set to $\lambda_{mix} = 0.05$. In FedDPMS, we set the hyper-parameter $\lambda = 0.05$. Moreover, FedDPMS's predetermined quota for the synthetic data of each abundant class and the standard deviation of the additive Gaussian noise in experiments on FMNIST, CIFAR10 and CIFAR100 were set to $\alpha = \{50, 200, 50\}$, $\sigma = \{3, 4, 4\}$ respectively. To choose the values of n, we recall that the data is generated according to the Dirichlet distribution with a small concentration parameter β; this implies presence of relatively few classes in each local dataset. Consequently, for FMNIST/CIFAR10 we set $n = 3$, while for CIFAR100 we set $n = 10$. In practice, the value of n can be chosen with the clients' help; e.g., client i reports to the server n_i, the number of

Table 1. The layers of the base networks for all approaches other than FedVAE and FedDPMS.

Layers	FMNIST	Layers	CIFAR10/100
Input	$28 \times 28 \times 1$	Input	$32 \times 32 \times 3$
Convolution	$3 \times 3(16)$	Convolution	$3 \times 3(32)$
Maxpool	2×2	Convolution	$3 \times 3(32)$
Convolution	$3 \times 3(32)$	Maxpool	2×2
Maxpool	2×2	Convolution	$3 \times 3(64)$
FC	$7 \times 7 \times 32$	Convolution	$3 \times 3(64)$
FC	32	Maxpool	2×2
Ouput	10	FC	$8 \times 8 \times 64$
		FC	128
		Output	10/100

Table 2. The layers for base networks of FedVAE and FedDPMS.

Layers	FMNIST	Layers	CIFAR10/100
Input	$28 \times 28 \times 1$	Input	$32 \times 32 \times 3$
Convolution	$3 \times 3(16)$	Convolution	$3 \times 3(32)$
Maxpool	2×2	Convolution	$3 \times 3(32)$
Convolution	$3 \times 3(32)$	Maxpool	2×2
Maxpool	2×2	Convolution	$3 \times 3(64)$
FC	$7 \times 7 \times 32$	Convolution	$3 \times 3(64)$
FC	32	Maxpool	2×2
FC	$7 \times 7 \times 32$	FC	$8 \times 8 \times 64$
Transposed-Con	$3 \times 3(16)$	FC	128
Transposed-Con	$3 \times 3(1)$	FC	$8 \times 8 \times 64$
Output	$28 \times 28 \times 1$	Transposed-Con	$3 \times 3(64)$
		Transposed-Con	$3 \times 3(64)$
		Transposed-Con	$3 \times 3(32)$
		Transposed-Con	$3 \times 3(3)$
		Output	$32 \times 32 \times 3$

the client's data classes decidedly less abundant than the rest of the classes. The server then sets $n = \min\{n_1, n_2, \ldots, n_K\}$. The communication cost and delay of establishing n in this way is negligible. Without a loss of generality, in the experiments we set the participation rate of clients k/K to 1.

3. Matching Algorithm

The matching algorithm aims to match pairs of clients having complementary local data; the result of matching depends on class distribution of each client's local dataset. One of the options is to find the size of the intersection between "abundant classes" $\mathcal{C}_i$ and "data scarce" classes $\mathcal{H}_j$, described as Algorithm 1.

It is worth mentioning that the information sharing may be uni-directional: if client i benefits from client j's latent data representation, the reverse need not be true. In fact, it may be the case that one client supplies latent representation information to several clients if the latter lack in data classes that the former has in its local dataset. Note that while the presented framework assumes categorical data labels, the methodology can be extended to regression tasks by relying on data binning.

4. Synthetic Images

Figure.2 presents examples of artificial images generated by the variational auto-encoder on FMNIST, CIFAR10 and CIFAR100; since the FMNIST's representation space is relatively simple, one may recognize the type of synthetic images,

Algorithm 1 Matching

```
Require:
    the set of clients who shared information, $\mathcal{A}$; the set of abundant classes at the server, $\mathcal{C}$; the selected clients, $\mathcal{S}_t$ client's "data scarce" classes $\mathcal{H} = \{\mathcal{H}_i | i \in \mathcal{S}_t\}$.
Ensure:
    Receiving source $\mathbf{R}$
 1: Server executes:
 2: if $\mathcal{C} = \emptyset$ then
 3:     $\mathbf{R} \leftarrow \emptyset$
 4: else
 5:     for $i \in \mathcal{S}_t$ do
 6:         $R_i \leftarrow \mathbf{argmax}_{j \in \mathcal{A}} |\mathcal{H}_i \cap \mathcal{C}_j|$
 7:     end for
 8:     $\mathbf{R} \leftarrow \{R_i | i \in \mathcal{S}_t\}$
 9: end if
10: return $\mathbf{R}$
```

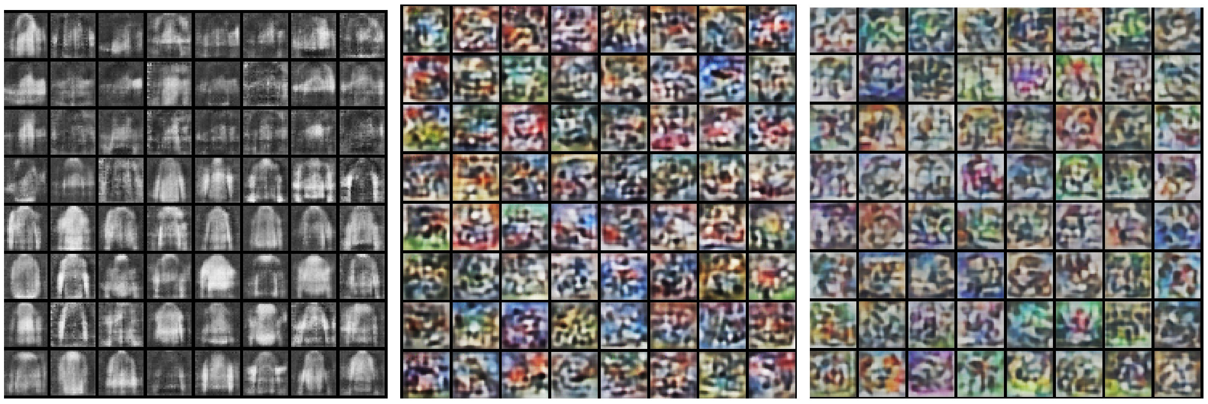

(a) FMNIST (b) CIFAR10 (c) CIFAR100

Figure 2. The synthetic images are generated based on the noisy version of latent means.

while the synthetic images generated to augment CIFAR10/100 datasets are incomprehensible. Nevertheless, even though incomprehensible to humans, synthesized images carry information that helps FedDPMS improve local training.

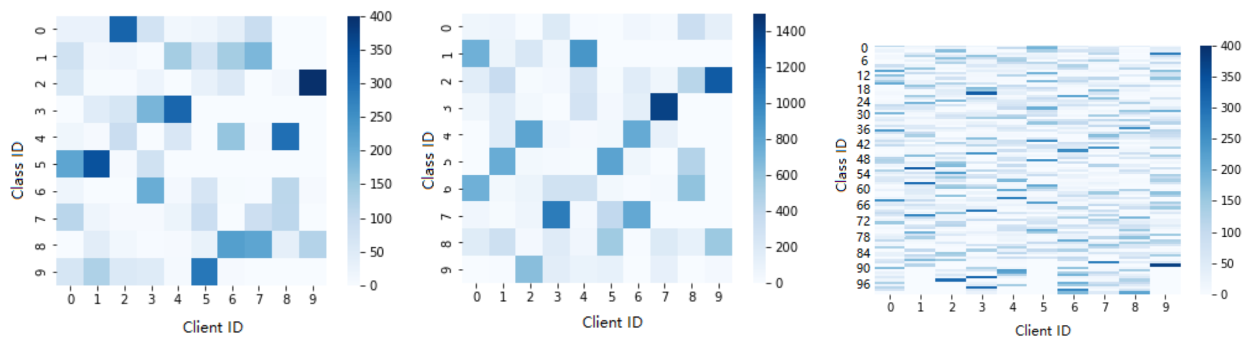
5. Visualization of Class Distribution in Data Partitions

Here we visualize class distributions of local clients induced by data partitioning of FMNIST, CIFAR10 and CIFRAR100. For example, the first column in Fig. 3, representing the class distribution of client '0' on FMNIST partition, shows that client '0' has samples in classes '1,2,5,7,9' while lacking classes '0,3,4,6,8'. As we can observe in Fig. 3, data partitions generated by the Dirichlet distribution may be extremely heterogeneous, becoming more so for smaller β.

6. Generative Model

In contrast to FedDG [5] and FedMix [6] which utilize interpolation to synthesize augmented data, FedDPMS relies on Variational Auto-Encoders (VAEs) [2] to learn latent data representation and enable data synthesis from DP-protected transformations of those representations. As its conventional counterparts, a VAE consists of two main components: an encoder and a decoder. The encoder is an approximate inference network $q_\phi(z \mid x)$ that maps input x to the latent vector z which approximates prior probability $p(z)$. The decoder is a generative network $p_\theta(x \mid z)$ which synthesizes x from the latent vector z. The goal of the VAE training is to maximize the likelihood of data points x, i.e., solve $\max_\theta \log p_\theta(x)$. This likelihood is challenging to compute/maximize; observing that

$$\log p_\theta(x) \geq E_{q_\phi(z|x)} \log p_\theta(x \mid z) - D_{KL}\left(q_\phi(z \mid x) \mid p(z)\right) \quad (1)$$



(a) FMNIST (b) CIFAR10 (c) CIFAR100

Figure 3. The class distribution of each client in non-iid data partitions generated by Dirichlet distribution with $\beta = 0.5$. Dark color implies more data while light color implies less data.

one can instead choose to maximize $E_{q_\phi(z|x)} \log p_\theta(x \mid z)$, the so-called evidence lower bound (ELBO). For convenience, we often assume that z has a Gaussian prior distribution, somewhat limiting the expressiveness of the VAE. In applications involving images, for example, this typically renders VAE-generated samples to be blurry. More broadly, when it comes to generating high-resolution images VAEs may be lagging behind GANs [1], but are faster, more reliable and theoretically better understood. Besides, for the problem studied in this paper, the utility of generated images is not measured by their resolution but by the robustness of the classification model trained on synthetically diversified distributed datasets.

7. Effects of the synthetic quota α and the additive Gaussian noise

We rely on empirical studies to investigate the impact of (predetermined) synthetic data quota α and the variance of the additive Gaussian noise σ^2 upon the performance of FedDPMS; the results of the experiments conducted on CIFAR10 are shown in Table 3. As implied by the table, when σ^2 is too large, the data synthesized using noisy means is unrecognizible by the local encoder / classifier. Note that the variance of the added noise is $S_f^2\sigma^2$, where $S_f^2 = \frac{1}{m}$. The synthetic data quota α controls the proportion of the synthesized data in the augmented local datasets. On the one hand, if the quota is too small, augmentation does not help improve diversity of local data and ultimately fails to help local training; on the other hand, large quota limits the impact of real data and in fact adversely affects accuracy of the learned models. We experimentally find that the synthesized data should amount to approximately 10% of a local dataset. The same value of α is used by all clients.

Table 3. Test accuracy for different generation quotas α and standard deviations σ. All results are based on $\beta = 0.5$. We set $\alpha = 200$ when running experiments for varied σ, while $\sigma = 4$ when running experiments for varied α. '-' in the table means VAE is not generating qualified data.

σ	ACCURACY (α = 200)	α	ACCURACY (σ = 4)
3	0.6734	50	0.6618
4	**0.6797**	100	0.6721
5	0.6719	150	0.6795
6	-	200	**0.6824**
		250	0.6712
		300	0.6655

8. Communication, Computation and Memory

Here we compare communication, computation and memory consumption of FedDPMS with state-of-the-art competing methods. To facilitate the comparison, we set FedAvg as the baseline and summarize the information in Table. 4. For clarity, we introduce the following notations: Θ is the number of parameters of the final model (encoder and classifier); l denotes the dimension of the latent space in FedDPMS; G is the size of a raw data point (e.g., if data is an image with width a and height b, then $G = ab$); n denotes the number of abundant classes; α is the number of synthetic samples in an abundant class;

$t_{avg}, t_{avg}+t_{prox}, t_{avg}+t_{moon}$ denotes the time to update the base model in FedAvg, FedProx and Moon methods in a round, respectively; K denotes the number of clients in the system; k is the number of clients selected in a round; $\nu = \frac{k}{K}$ denotes the sampling rate; T denotes the overall number of training epochs and T_p denotes the number of preliminary training epochs.

8.1. Communication costs

Compared to the baselines (e.g., FedProx [4] and Moon [3]), FedDPMS requires additional communication resources for two reasons. First, to enable diversification of local data, the server acquires noisy latent representation means from an assisting client and shares them with a matching benefitting clients. We emphasize that each client can assist or benefit from another at most once during the entire training process. The worst-case scenario from the communications perspective is if every client in the system fulfills both roles. In that case, the relative overhead introduced by transmitting the means of latent representation is $r_1 = \alpha nl/\Theta$. Typically, this overhead is rather small; for instance, in our experiments with CIFAR10, $n = 3$, $\alpha = 200$, $l = 128$, $\Theta > 1000000$, and thus $\alpha nl/\Theta \approx 7.68\%$. The second part of the additional communication cost is due to uploading local decoder model (a single event at the end of preliminary training) and the download of the global decoder by the clients seeking to diversify local dataset during the secondary training phase. To compute E, the expected number of selected clients who download the global decoder during secondary training (lasting $T - T_p$ rounds), we denote by $\mathcal{E}_t$ an event that a client selected at round t has not benefitted from latent representation information in the previous $t-1$ rounds of secondary training. Clearly,

$$\Pr(\mathcal{E}_t) = (1-\nu)^{t-1}. \tag{2}$$

Then, the expected number of the selected clients is found as

$$E = \left((1-\nu)^0 + \cdots + (1-\nu)^{T-T_p-1}\right) k = \frac{1-(1-\nu)^{T-T_p}}{\nu} k. \tag{3}$$

The ratio between the communication overhead and the baseline is

$$r_2 = \frac{E\Theta}{2k\Theta T} = \frac{1-(1-\nu)^{T-T_p}}{2\nu T}. \tag{4}$$

If $\nu \ll 1$ then $r_2 = \frac{1}{2} - \frac{T_p}{2T}$, implying that all of the clients selected in a secondary training round actually download the global decoder. For large T, $r_2 = \mathcal{O}(\frac{1}{2\nu T})$. For convenience, we specify the communication cost of FedDPMS in Table 4 as $UB = r_1 + r_2$.

8.2. Computational cost

While FedAvg schemes deploy and train only a classifier (i.e., a single neural network), FedDPMS relies on VAEs and in the preliminary phase requires training of encoder/classifier/decoder. Specifically, the computational overhead of FedDPMS relative to FedAvg is $\frac{T_p}{T}$. In our experiments, FedDPMS achieves the best performance when $T_p/T = 0.4$, implying 40% more computation; clearly, varying T_p/T explores the trade-off between performance and computational cost. Note that when training large models, FedDPMS requires more secondary training rounds and thus ratio $\frac{T_p}{T}$ becomes small, implying less computational overhead.

8.3. Memory requirements

Since FedDPMS relies on VAEs, in the preliminary training it requires approximately twice as much memory as FedAvg schemes that deploy the same neural network architectures. In the secondary training, each client stores the global decoder's parameters for at most one round, and hence the average additional memory is negligible. FedDPMS further requires additional memory to store synthetic data; that overhead is typically negligible compared to the total memory use. Note that among the competing methods, FedProx and Moon each introduce a regularization term and thus they too need additional memory – the former requires twice as much memory as FedAvg while the latter needs three times as much.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 4

[2] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[3] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 5

Table 4. A comparison of the communication, computation and memory requirement for FedDPMS and the competing methods.

SCHEME	COMMUNICATION	COMPUTATION	MEMORY
FEDAVG	1	1	1
FEDPROX	1	$1 + \frac{t_{prox}}{t_{avg}}$	2
FEDMIX	$1 + G/\Theta$	1	1
MOON	1	$1 + \frac{t_{moon}}{t_{avg}}$	3
FEDDPMS	$1 + UB$	$1+T_p/T$	$1+T_p/T$

[4] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018. 5

[5] Quande Liu, Cheng Chen, Jing Qin, Qi Dou, and Pheng-Ann Heng. Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1013–1023, 2021. 3

[6] Tehrim Yoon, Sumin Shin, Sung Ju Hwang, and Eunho Yang. Fedmix: Approximation of mixup under mean augmented federated learning. *arXiv preprint arXiv:2107.00233*, 2021. 3