Appendix

A. Reproducibility & Ethics Statements

Reproducibility We have specified the setup for all experiments in the paper including hyperparameters, presented the algorithm in detail, and will provided the source code to make sure that our results are reproducible.

Ethics Our work is related to federated learning and one of FL's goals is to preserve user's privacy. Considering this ethically sensitive topic, we have shown that differential privacy of our method FedDM can be guaranteed with Gaussian mechanism. On the other hand, potential negative impacts to users like data leakage must be taken into account carefully and cautiously if such differentially private algorithms are deployed in real-world sensitive applications. Whereas, it should be noted that our work do not directly leverage real-world sensitive data, and all experiments are conducted on synthetic data, MNIST, CIFAR10 or CIFAR100, all of which are standard non-private datasets.

B. Synthetic Binary Classification

We design a synthetic 1-D binary classification problem to better illustrate the advantage of learning a surrogate function for the training objective. Specifically, we construct a dataset $\mathcal{D}_s = \{(x_i, y_i)|i = 1, \dots, n\}$ with $n = 100$ synthetic pairs in the following way:

$$x_i \sim \mathcal{N}(0, 1), y_i = \begin{cases} 1 & (x_i \geq 0 \text{ and } p_i \geq 0.9) \text{ or } (x_i < 0 \text{ and } p_i < 0.1) \\ 0 & \text{otherwise} \end{cases}, \tag{16}$$

where p_i is a random value sampled from Uniform$(0, 1)$. A prediction is made by $\hat{y}_i = \text{Sigmoid}(wx_i)$ with the weight w as the trainable parameter. We use the binary cross entropy as the training objective:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(\hat{y}_i) + (1 - y_i) \log((1 - \hat{y}_i)). \tag{17}$$

Then we use $n' = 20$ randomly initialized examples $\{(\tilde{x}_j, \tilde{y}_j)|j = 1, \dots, 20\}$ to match the objective around $w = 0$ as introduced in Section 3.2. We plot the original objective, the surrogate function, and the tangent line at $w = 0$ obtained by the gradient in Figure 1.

C. Message size of different FL methods

In this section, we provide specific message size under different data partitioning of FedDM. As discussed previously, the message size of all baseline methods are determined on the model size, while the message size varies from different scenarios and ipc values. When there are 10 clients, we set ipc=10 for MNIST and CIFAR10, and ipc=5 for CIFAR100. We present the results in Table 5. It can be observed that FedDM are more advantageous for unbalanced data partitioning, such as $\text{Dir}_{10}(0.1)$ and $\text{Dir}_{10}(0.01)$. For the experiment of $\text{Dir}_{50}(0.5)$ on CIFAR10 with ConvNet, the message sizes of FedDM and baselines are 3.1×10^6 and 1.6×10^7 respectively, where our method saves about 80% costs per round. Moreover, if the underlying model are changed to ResNet-18 for $\text{Dir}_{10}(0.5)$, then the number of parameters is about 1.1×10^8.

Table 5. The size of message uploaded to the server (number of float parameters).

	MNIST	CIFAR10	CIFAR100
$\text{Dir}_{10}(0.5)$	635040	2672640	10045440
$\text{Dir}_{10}(0.1)$	368480	1351680	4761600
$\text{Dir}_{10}(0.01)$	109760	460800	2135040
Baseline	3177060	3200100	5044200

D. Proof of Theorem 3.2

According to Theorem 3.1, using DP-SGD to train a neural network can protect differential privacy. Back to FedDM, we first look at each client separately to investigate its differential privacy. We show that the gradient of $\mathcal{L}_c$ in equation 8 can be written as the average of individual gradients for each real example,

$$\nabla_{\mathcal{S}_k}\mathcal{L}_c = \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x_i,y_i)\in B_c^{\mathcal{D}_k}} \tilde{g}(x_i), \tag{18}$$

where $\tilde{g}(x_i)$ is the modified gradient for x_i. Recall the equation of $\mathcal{L}_c$, we have

$$\begin{aligned}\mathcal{L}_c = &\overbrace{\| \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(\boldsymbol{x},y)\in B_c^{\mathcal{D}_k}} h_w(x) - \frac{1}{|B_c^{\mathcal{S}_k}|}\sum_{(\tilde{x},\tilde{y})\in B_c^{\mathcal{S}_k}} h_w(\tilde{x})\|^2}^{\mathcal{L}_{c,h}} \\ &+ \underbrace{\| \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x,y)\in B_c^{\mathcal{D}_k}} z_w(x) - \frac{1}{|B_c^{\mathcal{S}_k}|}\sum_{(\tilde{x},\tilde{y})\in B_c^{\mathcal{S}_k}} z_w(\tilde{x})\|^2}_{\mathcal{L}_{c,z}}.\end{aligned} \tag{19}$$

$\mathcal{L}_c$ are divided into two similar parts, $\mathcal{L}_{c,h}$ and $\mathcal{L}_{c,z}$. Then we take a look at the gradient of $\mathcal{L}_{c,h}$ with respect to $\mathcal{S}_k$ below:

$$\begin{aligned}\nabla_{\mathcal{S}_k}\mathcal{L}_{c,h} &= 2(\overbrace{\frac{\partial \frac{1}{|B_c^{\mathcal{S}_k}|}\sum_{(\tilde{x},\tilde{y})\in B_c^{\mathcal{S}_k}} h_w(\tilde{x})}{\partial \mathcal{S}_k}}^{J_{\mathcal{S}_k}})^T(\overbrace{\frac{1}{|B_c^{\mathcal{S}_k}|}\sum_{(\tilde{x},\tilde{y})\in B_c^{\mathcal{S}_k}} h_w(\tilde{x})}^{h_w(\mathcal{S}_k)} - \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x,y)\in B_c^{\mathcal{D}_k}} h_w(x)) \\ &= J_{\mathcal{S}_k}(h_w(\mathcal{S}_k) - \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x,y)\in B_c^{\mathcal{D}_k}} h_w(x)) \\ &= \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x,y)\in B_c^{\mathcal{D}_k}} \underbrace{J_{\mathcal{S}_k}(h_w(\mathcal{S}_k) - h_w(x))}_{\tilde{h}_w(x)} = \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x,y)\in B_c^{\mathcal{D}_k}} \tilde{h}_w(x).\end{aligned} \tag{20}$$

Similarly, we have

$$\nabla_{\mathcal{S}_k}\mathcal{L}_{c,z} = \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x,y)\in B_c^{\mathcal{D}_k}} \tilde{z}_w(x). \tag{21}$$

Then the final gradient of $\mathcal{L}_c$ is

$$\nabla_{\mathcal{S}_k}\mathcal{L}_c = \nabla_{\mathcal{S}_k}\mathcal{L}_{c,h} + \nabla_{\mathcal{S}_k}\mathcal{L}_{c,z} = \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x,y)\in B_c^{\mathcal{D}_k}} \overbrace{(\tilde{h}_w(x) + \tilde{z}_w(x))}^{\tilde{g}(x)} = \frac{1}{|B_c^{\mathcal{D}_k}|}\sum_{(x,y)\in B_c^{\mathcal{D}_k}} \tilde{g}(x). \tag{22}$$

It indicates that the synthetic set can be regarded as an equivalence to the network parameter in DP-SGD, and leads to the conclusion that for each client k, Theorem 3.1 holds during optimizing $\mathcal{S}_k$. Furthermore, since the synthetic dataset is initialized from random noise, it would not leak privacy at the beginning of the optimization.

Next, to extend DP guarantee to a system with K clients, we use parallel composition [34]:

Theorem D.1 (**Parallel Composition [34]**). If there are K mechanisms $M_1,\ldots,M_K$ computed on disjoint subsets whose privacy guarantees are $(\epsilon_1,\delta_1),\ldots,(\epsilon_K,\delta_K)$ respectively, then any function of $M_1,\ldots,M_K$ is $(\max_i \epsilon_i, \max_i \delta_i)$-differential private.

We can see that different clients maintain their own local datasets, which satisfies disjoint property. Then this Gaussian mechanism is still (ϵ, δ)-differentially private for the whole system if each client satisfies (ϵ, δ)-differential privacy. In addition, to quantify how much noise is required for each client, we can make use of the Tail bound in [1]:

$$\delta = \min_{\lambda} \exp(\alpha_M(\lambda) - \lambda\epsilon). \tag{23}$$

Based on [1], $\alpha_M(\lambda) \leq Tq^2\lambda^2/\sigma^2$, without loss of generality, set $\lambda = \sigma^2$, it holds that $\delta \leq \exp(Tq^2\sigma^2 - \epsilon\sigma^2)$, and $\sigma \geq \sqrt{\frac{\log(\delta)}{Tq^2-\epsilon}}$. When $Tq^2 \leq \epsilon/2$, we have $\sigma \geq \sqrt{\frac{2\log(1/\sigma)}{\epsilon}}$.

To guarantee differential privacy when leveraging the synthetic dataset for downstream tasks, we introduce the property of post-processing below:

Lemma D.1 (Robustness to post-processing [15]). Let $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ be a randomized mechanism that is (ϵ, δ)-differentially private. If $\mathcal{F} : \mathcal{R} \to \mathcal{R}'$ be an arbitrary deterministic or randomized mapping, $\mathcal{F}(\mathcal{M})$ is also (ϵ, δ)-differentially private.

Training a network on the synthetic dataset $\mathcal{S}$ is a post-processing operation, and Lemma D.1 ensures that any post-processing computation on $\mathcal{S}$ is differentially private as long as the generation of $\mathcal{S}$ satisfies (ϵ, δ)-differential privacy. This property has also been deployed in similar applications such as [52], where the synthetic text generated from a DP-trained language model was used in various downstream tasks and still preserved privacy.

Finally, with Theorem 3.1, Theorem D.1 and Lemma D.1, we complete the proof of Theorem 3.2.

For the total DP budget of R rounds, suppose in one round, the Gaussian mechanism can guarantee (ϵ, δ)-DP. Querying for R rounds leads to $(O(\sqrt{R\log(1/\delta')}) \cdot \epsilon), R\delta + \delta')$-DP guarantee $\forall \delta' \in (0, 1/2)$, based on the advanced composition [21], which guarantees a moderate privacy budget increase for R rounds.

E. FedDM with DP-SGD

Algorithm 2 FedDM: Federated Learning with Iterative Distribution Matching

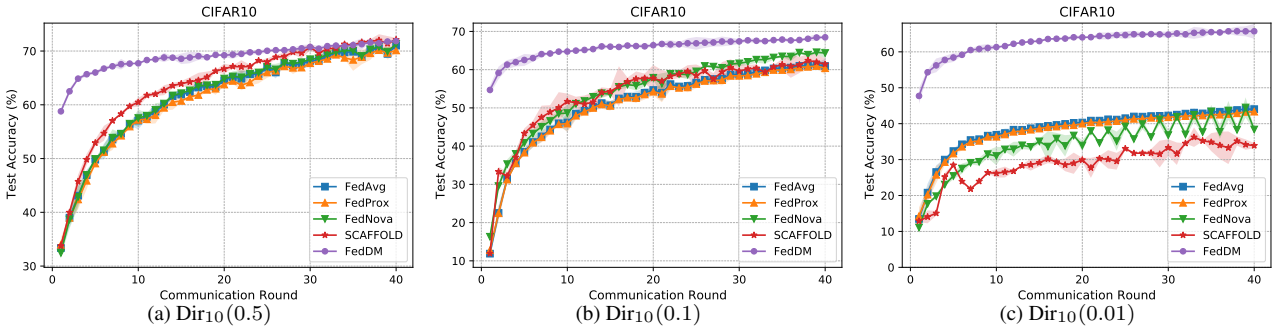
Input: Training set $\mathcal{D}$, set of synthetic samples $\mathcal{S}$, deep neural network parameterized with w, probability distribution over parameters $\mathcal{P}_w$, Gaussian noise level σ, gradient norm bound $\mathcal{C}$, training iterations of distribution matching T, learning rate η_c and η_s.
Server executes:
for each round $r = 1, \ldots, R$ **do**
  for client $k = 1, \ldots, K$ **do**
    $\mathcal{S}_k \leftarrow$ ClientUpdate(k, w_r, σ)
    Transmit $\mathcal{S}_k$ to the server
  end for
  Aggregate synthesized data from each client and build the surrogate function by Equation 9
  Update weights to w_{r+1} on $\mathcal{S}$ by SGD with the learning rate η_s
end for
ClientUpdate(k, w_r, σ):
Initialize $\mathcal{S}_k$ with random noise.
for $t = 0, \cdots, T-1$ **do**
  Sample $w \sim P_w(w_r)$
  Sample mini-batch pairs $B_c^{\mathcal{D}_k} \sim \mathcal{D}_k$ and $B_c^{\mathcal{S}_k} \sim \mathcal{S}_k$ for each class c
  Compute $\mathcal{L}_c$ based on Equation 8, $\mathcal{L} \leftarrow \sum_{c=0}^{C-1} \mathcal{L}_c$
  Obtain the clipped gradient: $\nabla_{\mathcal{S}_k}\mathcal{L}_c \leftarrow \nabla_{\mathcal{S}_k}\mathcal{L}_c / \max\left(1, \frac{\|\nabla_{\mathcal{S}_k}\mathcal{L}_c\|_2}{\mathcal{C}}\right)$
  Add Gaussian noise: $\nabla_{\mathcal{S}_k}\mathcal{L}_c \leftarrow \nabla_{\mathcal{S}_k}\mathcal{L}_c + \frac{1}{|B_c^{\mathcal{D}_k}|}\mathcal{N}(0, \sigma^2\mathcal{C}^2\mathbf{I})$
  Update $\mathcal{S}_k \leftarrow \mathcal{S}_k - \eta_c\nabla_{\mathcal{S}_k}\mathcal{L}$
end for

F. Additional Experimental Results

F.1. Learning curves of FL methods

We show a complete set of learning curves for all of our experiments.

Increasing the number of communication rounds to 40. Even though we focus on the limited budget of 20 communication rounds in the main paper, we conduct an experiment to evaluate the performance of all considered methods with 40 rounds. In Figure 6, it can be observed that FedDM still outperforms other model averaging based methods in terms of final test accuracy and convergence rate especially under unbalanced distribution, showing the effectiveness of our method.

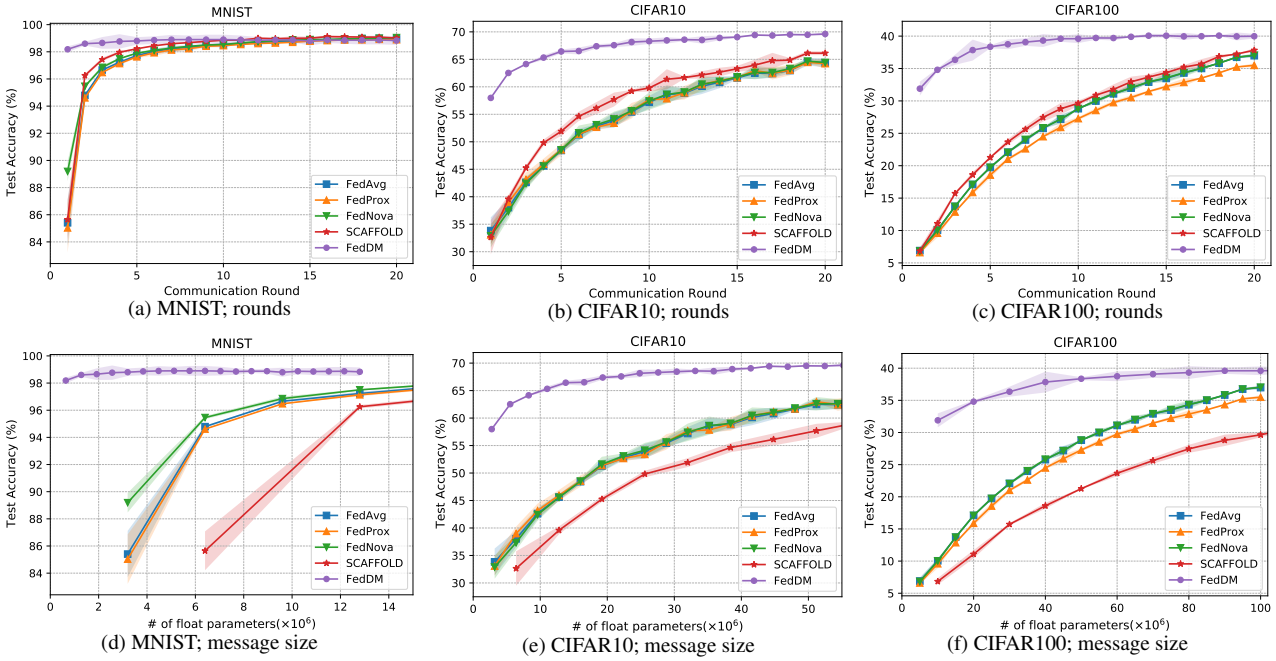


(a) $\text{Dir}_{10}(0.5)$ (b) $\text{Dir}_{10}(0.1)$ (c) $\text{Dir}_{10}(0.01)$

Figure 6. Test accuracy of different methods with 40 communication rounds on CIFAR10.

Different data partitioning. Here we present curves for different data partitioning. We observe that FedDM still outperforms all other baselines under scenarios of $\text{Dir}_{10}(50)$ in Figure 10 which is almost an i.i.d. data partitioning, and $\text{Dir}_{50}(0.5)$ in Figure 11 which has more clients.

- $\text{Dir}_{10}(0.5)$

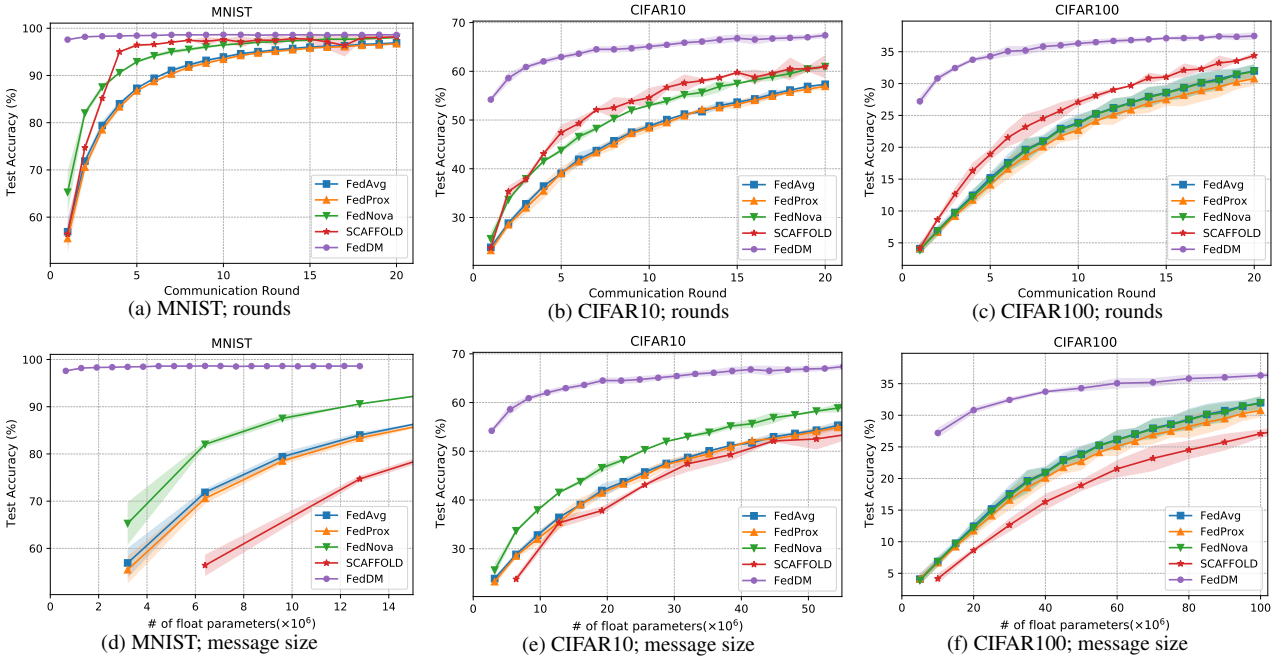


(a) MNIST; rounds (b) CIFAR10; rounds (c) CIFAR100; rounds

(d) MNIST; message size (e) CIFAR10; message size (f) CIFAR100; message size

Figure 7. Test accuracy under $\text{Dir}_{10}(0.5)$.

- $\text{Dir}_{10}(0.1)$

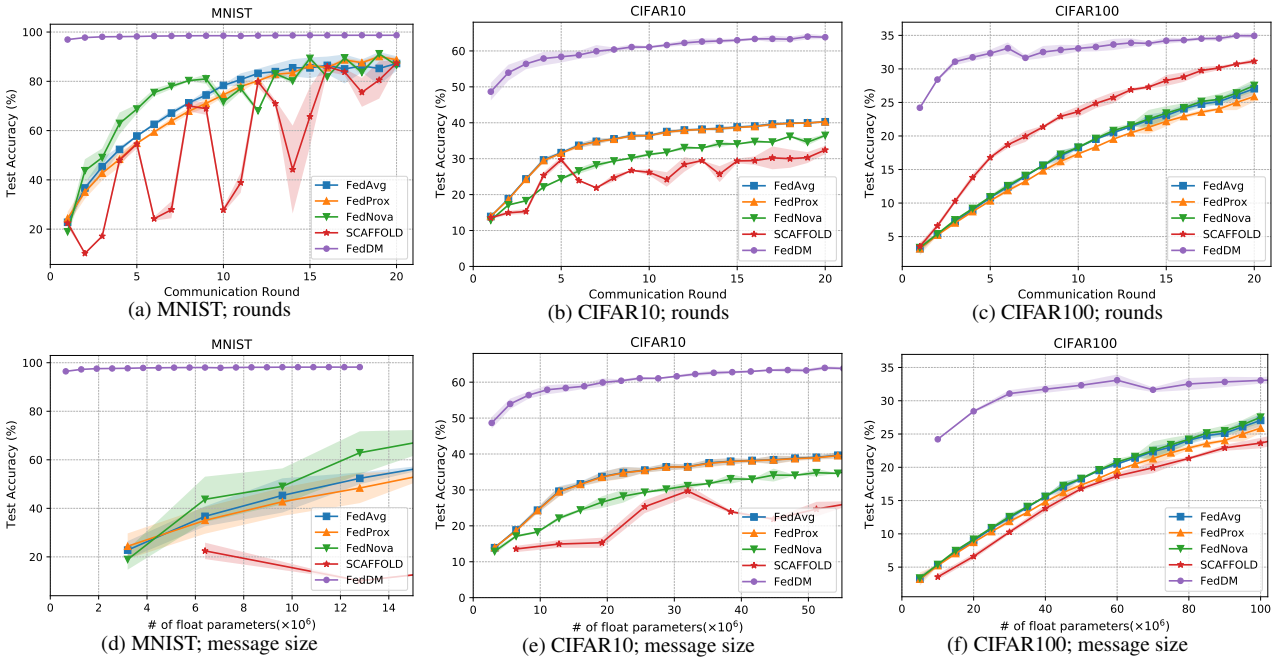


(a) MNIST; rounds (b) CIFAR10; rounds (c) CIFAR100; rounds

(d) MNIST; message size (e) CIFAR10; message size (f) CIFAR100; message size

Figure 8. Test accuracy under $\text{Dir}_{10}(0.1)$.

- $\text{Dir}_{10}(0.01)$

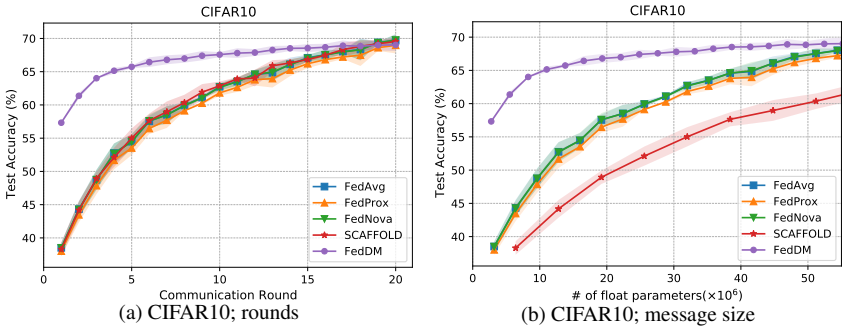


(a) MNIST; rounds (b) CIFAR10; rounds (c) CIFAR100; rounds

(d) MNIST; message size (e) CIFAR10; message size (f) CIFAR100; message size

Figure 9. Test accuracy under $\text{Dir}_{10}(0.01)$.

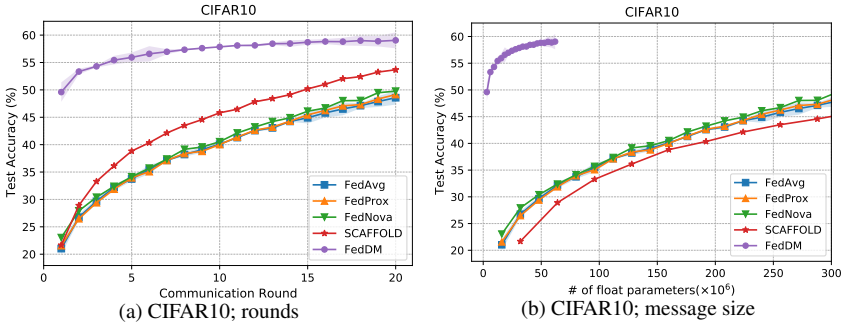
- i.i.d., $\text{Dir}_{10}(50)$



(a) CIFAR10; rounds (b) CIFAR10; message size

Figure 10. Test accuracy under $\text{Dir}_{10}(50)$.

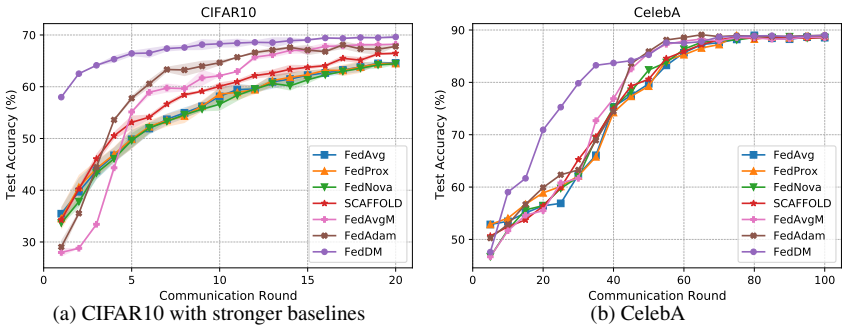
- $\text{Dir}_{50}(0.5)$



(a) CIFAR10; rounds (b) CIFAR10; message size

Figure 11. Test accuracy under $\text{Dir}_{50}(0.5)$.

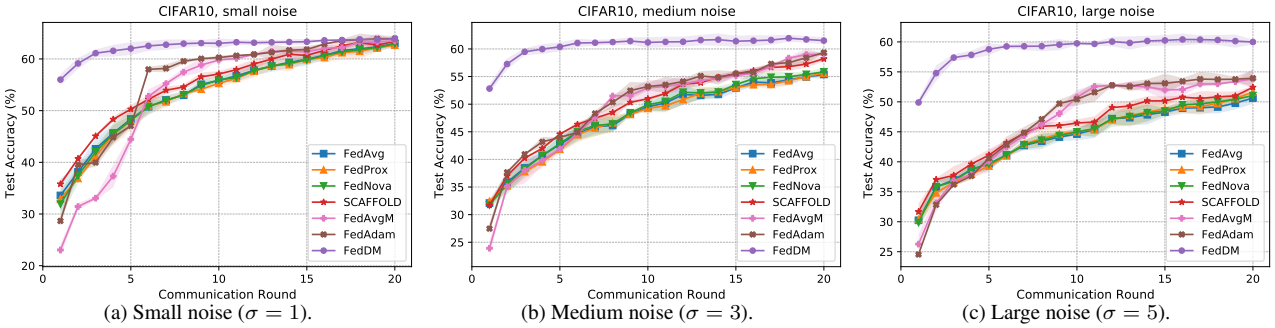
Comparison with stronger baselines and a realistic dataset. We include two stronger baselines, FedAvgM and FedAdam, and evaluate their performance on CIFAR10 under the default setting $\text{Dir}_{10}(0.5)$. As shownin Figure 12a, FedDM still outperforms selected methods. In addition, we conduct an experiment on a realistic dataset, CelebA (2 classes) with a pre-trained ViT-small model. 10% of 9343 clients are used and 10 of them are sampled each round. With the batch size of 5 and the local epoch of 1 suggested by LEAF [4], we only tune learning rates. We set ipc$=2, T=50$ in FedDM. Performance of R=100 rounds is shown in Figure 12b, where FedDM can reach a satisfactory accuracy with fewer rounds. The message size of FedDM per client is 6.0×10^5, significantly smaller than other methods whose size is 2.1×10^7.



(a) CIFAR10 with stronger baselines (b) CelebA

Figure 12. Comparison with stronger baselines and a realistic dataset.

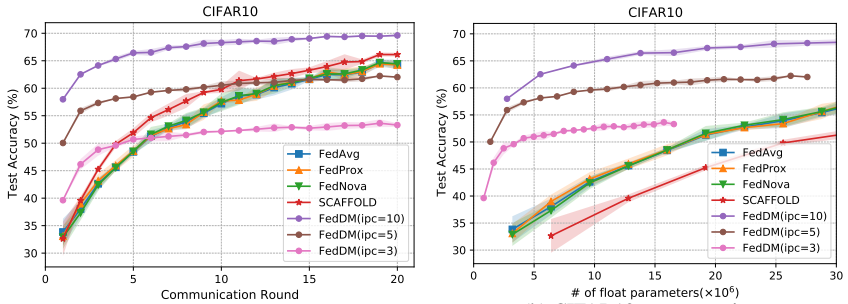
Different noise levels. Figure 13 displays learning curves of different σ.



(a) Small noise ($\sigma = 1$). (b) Medium noise ($\sigma = 3$). (c) Large noise ($\sigma = 5$).

Figure 13. Performance of FL methods with different levels of noise.

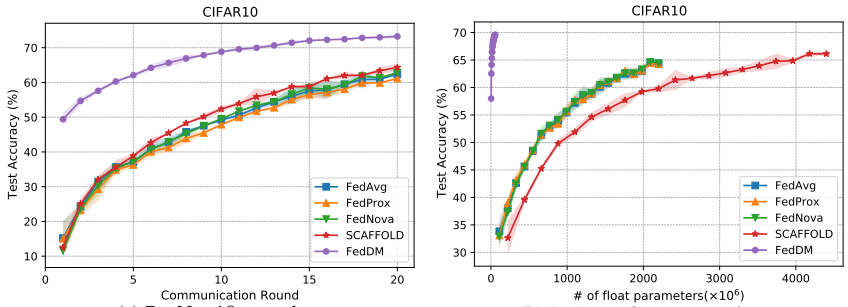
Effects of ipc. We show test accuracy curves to analyze effects of ipc in Figure 14.



(a) CIFAR10; rounds (b) CIFAR10; message size

Figure 14. Performance of FedDM with different values of ipc.

Performance on ResNet-18. Learning curves of test accuracy along with rounds and message size are shown in Figure 15.



(a) ResNet-18; rounds (b) ResNet-18; message size

Figure 15. Performance of FL methods on ResNet-18.

Transmitting real data. We present a comparison with sending real images (REAL) in Figure 16.

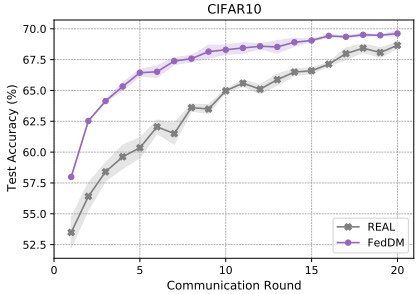


Figure 16. Test accuracy of FedDM and REAL.

F.2. Visualization of the synthetic dataset

By randomly picking a client under data partitioning of $\text{Dir}_{10}(0.5)$, we provide the visualization of our synthetic dataset under different noise levels in Figure 17, 18, 19, and 20 when $\mathcal{S}$ is initialized from random Gaussian noise $\mathcal{N}(0, 1)$. It can be observed that even when there is no noise added to the gradient during optimization of the synthetic dataset, those images are still illegible from their original classes in Figure 17. Furthermore, as σ increases, synthesized data become harder to recognize, which protects the client's privacy successfully.

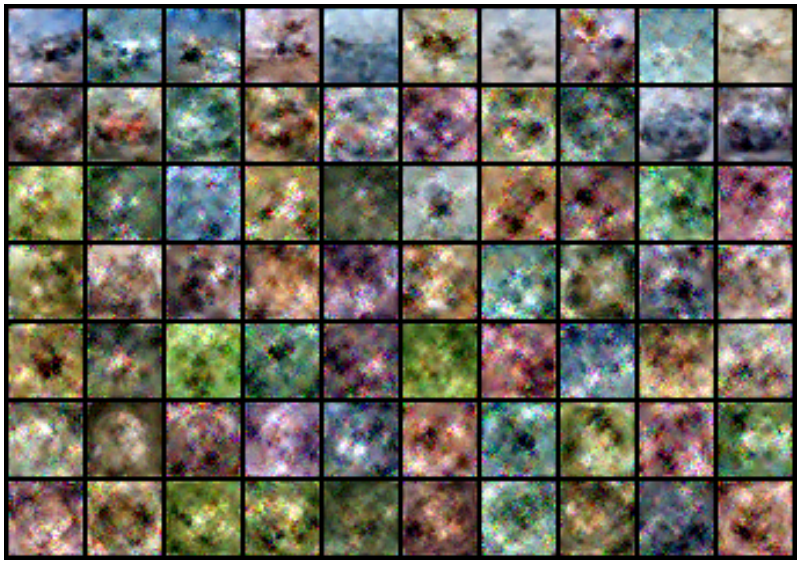


Figure 17. Synthesized images when no noise is added.

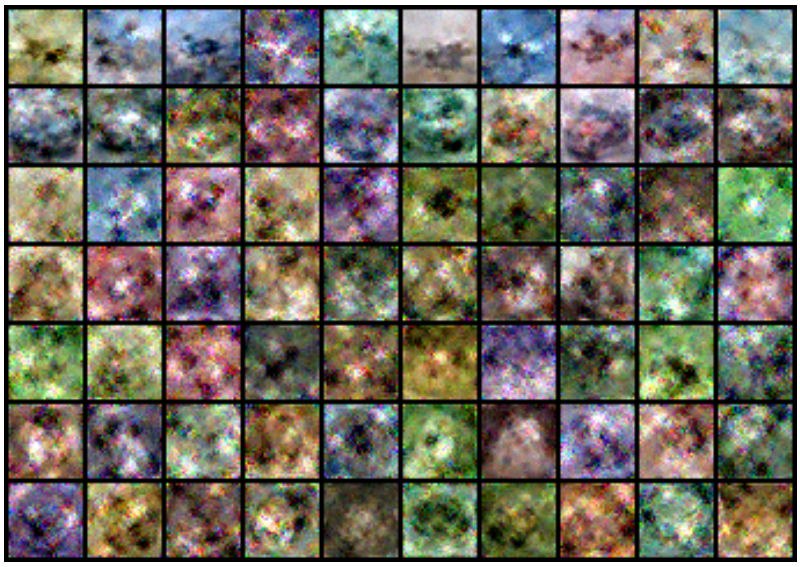


Figure 18. Synthesized images with $\sigma = 1$

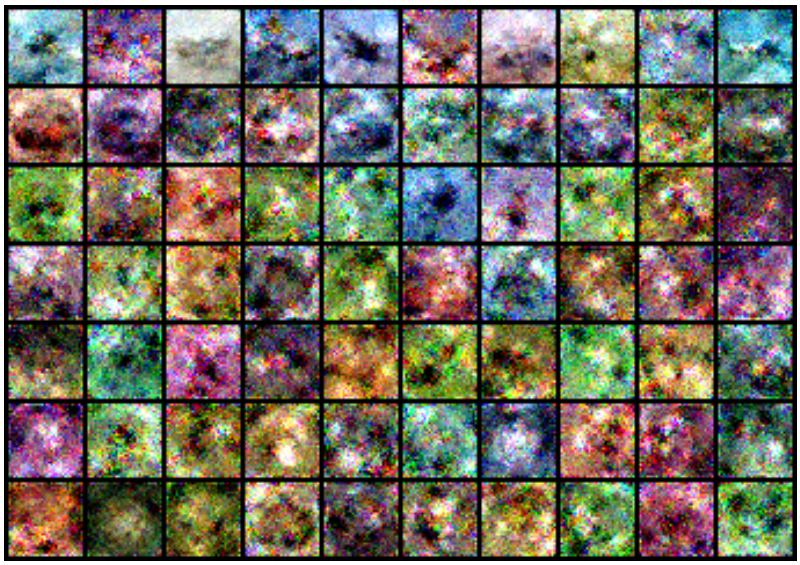


Figure 19. Synthesized images with $\sigma = 3$.

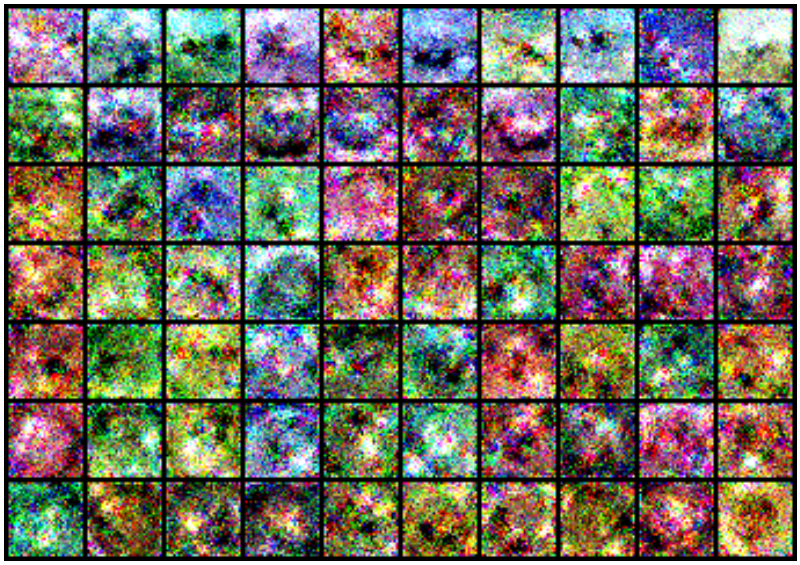


Figure 20. Synthesized images with $\sigma = 5$.