Supplementary Materials: FedDEO: Description-Enhanced One-Shot Federated Learning with Diffusion Models

Anonymous Authors

In the supplementary materials, we provide a substantial amount of content that couldn't be included in the main text due to space limitations, which can mainly be divided into three sections: 1) **Method Details.** We further elaborate on the details of the proposed method, including proofs of theoretical analyses and pseudo code of the proposed method. 2) **Experimental Setting Details.** More detailed information about the datasets and client partitions are provided, along with thorough implementation details. 3) **Supplementary Experiments.** We further demonstrate the performance of the proposed method through supplementary quantitation and visualization experiments, such as ablation experiments regarding the number of clients and the use of different diffusion models, quantitation experiments concerning communication, computation and privacy, more visualization experiments of synthetic datasets, privacy-related visualization experiments, etc.

1 METHOD DETAILS

Our method focuses on the diffusion-based OSFL methods [17–19], following the general framework depicted in Figure 1. The core idea of such methods lies in the guidance regarding the local data distribution provided by clients. The server's DM conducts conditional generation based on this guidance, resulting in the synthetic dataset complies with the client distributions, which is used to train the aggregated model. Despite their prevalence, such methods often lack rigorous theoretical analyses. In the main text, we delve into the relationship between the synthetic data distribution and the client distribution and provide theoretical analyses. In this section, we detail our theoretical analyses and provide the pseudocode of the proposed method in Algorithm 1 and 2.

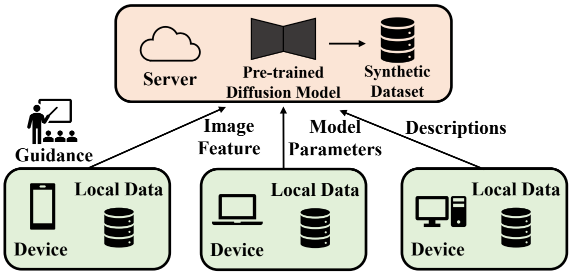

Figure 1: General framework for diffusion-based FL methods.

1.1 Proofs

In this section, we provide a proof of the theoretical analyses in the main text, regrading the upper bound of the KL divergence between the distributions of the synthetic dataset and the client datasets. Firstly, in section 3.1 of the main text, we introduce an assumption: for the data distribution of the client's local dataset $p_n(\mathbf{x})$ and the data distribution of the DMs $p_{\epsilon_\theta}(\mathbf{x})$, we have:

Assumption 1 *There exists $\lambda > 0$ such that the Kullback-Leibler divergence from $p_n(\mathbf{x})$ to $p_{\epsilon_\theta}(\mathbf{x})$ is bounded above by λ:*

$$KL(p_{\epsilon_\theta}(\mathbf{x})\|p_u(\mathbf{x})) < \lambda \tag{1}$$

Based on this assumption, considering the KL divergence between the distributions of the original client dataset $p_n(\mathbf{x})$ and the synthetic dataset $p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})$, which is the conditional distribution of the DMs conditioned on the trained discriptions $\mathbf{d}$, we have:

Theorem 1 *For the distribution of client data $p_n(\mathbf{x})$ and the conditional distribution $p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})$ of the DM ϵ_θ conditioned the description $\mathbf{d}$ trained on the clients, we have:*

$$\begin{aligned} KL(p_n(\mathbf{x})\|p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})) &= \int p_n(\mathbf{x}) \log \frac{p_n(\mathbf{x})p_{\epsilon_\theta}(\mathbf{d})}{p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x})p_{\epsilon_\theta}(\mathbf{x})} d\mathbf{x} \\ &< \lambda + \mathbb{E}(\log p_{\epsilon_\theta}(\mathbf{d})) - \int p_n(\mathbf{x}) \log p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x}) dx \end{aligned} \tag{2}$$

Proof. Firstly, based on the definition of KL divergence, we have:

$$KL(p_n(\mathbf{x})\|p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})) = -\int p_n(\mathbf{x}) \log \frac{p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})}{p_n(\mathbf{x})} d\mathbf{x} \tag{3}$$

Based on the Bayes' theorem, we have:

$$p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d}) = \frac{p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x})p_{\epsilon_\theta}(\mathbf{x})}{p_{\epsilon_\theta}(\mathbf{d})} \tag{4}$$

From Eq. 3 and Eq. 4, we have:

$$\begin{aligned} &KL(p_n(\mathbf{x})\|p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})) = -\int p_n(\mathbf{x}) \log \frac{p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})}{p_n(\mathbf{x})} d\mathbf{x} \\ &= -\int p_n(\mathbf{x}) \log \frac{p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x})p_{\epsilon_\theta}(\mathbf{x})}{p_n(\mathbf{x})p_{\epsilon_\theta}(\mathbf{d})} d\mathbf{x} \\ &= \int p_n(\mathbf{x}) \log \frac{p_n(\mathbf{x})p_{\epsilon_\theta}(\mathbf{d})}{p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x})p_{\epsilon_\theta}(\mathbf{x})} d\mathbf{x} \\ &= \int p_n(\mathbf{x}) \log \frac{p_n(\mathbf{x})}{p_{\epsilon_\theta}(\mathbf{x})} d\mathbf{x} + \int p_n(\mathbf{x}) \log \frac{p_{\epsilon_\theta}(\mathbf{d})}{p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x})} d\mathbf{x} \\ &= KL(p_n(\mathbf{x})\|p_{\epsilon_\theta}(\mathbf{x})) + \int p_n(\mathbf{x}) \log \frac{p_{\epsilon_\theta}(\mathbf{d})}{p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x})} d\mathbf{x} \end{aligned} \tag{5}$$

From Eq. 1 and Eq. 5, we have:

$$KL(p_n(\mathbf{x})\|p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})) < \lambda + \int p_n(\mathbf{x}) \log \frac{p_{\epsilon_\theta}(\mathbf{d})}{p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x})} d\mathbf{x} \tag{6}$$

Algorithm 1 Client Description Training

```
Input: The client datasets $\mathcal{D}_n, n = 1, \ldots, N$, a pre-trained DM $\epsilon_\theta$,
       text features of categories $f_c, c \in C_n$
Output: The descriptions of each category within the clients $\mathbf{d}_{n,c}$
for client $n = 1, \ldots, N$ do
    for category $c \in C_n$ do
        $\mathbf{d}_{n,c} \leftarrow f_c$
        for epoch $i = 1, \ldots, S$ do
            for $\mathbf{x}$ in $\mathcal{D}_n$ do
                $\mathbf{x}_0 := \mathbf{x}$
                $t \sim Uniform(\{1, ..., T\})$
                $\epsilon \sim \mathcal{N}(0, \mathcal{I})$
                $\mathbf{x}_t = \sqrt{a_t}\mathbf{x}_0 + \sqrt{1 - a_t}\epsilon$
                $\mathcal{L}(\mathbf{x}_t, \mathbf{d}_{n,c}, t) = \mathcal{L}_{MSE}(\epsilon, \epsilon_\theta(\mathbf{x}_t, t|\mathbf{d}_{n,c}))))$
                update $\mathbf{d}_{n,c}$
            end for
        end for
    end for
end for
return the descriptions $\mathbf{d}_{n,c}, n = 1, \ldots, N, c \in C_n$
```

Algorithm 2 Server Image Generation

```
Input: The Diffusion Model $\epsilon_\theta$, the text features of categories $f_c$,
       the trained local descriptions $\mathbf{d}_{n,c}, n = 1, \ldots, N, c \in C_n$
Output: The aggregated model $\mathbf{w}_g$, the synthetic dataset $\{\hat{x}_i^{n,c}\}$.
for client $n = 1, \ldots, N$ do
    for category $c \in C_n$ do
        for $i = 1, \ldots, R$ do
            $\mathbf{x}_T \sim \mathcal{N}(0, \mathcal{I})$
            for timestep $t$ = T, ..., 1 do
                $\epsilon_t \sim \mathcal{N}(0, \mathcal{I})$
                $\hat{\epsilon}_\theta(\mathbf{x}_t|f_c, \mathbf{d}_{n,c}) = \epsilon_\theta(\mathbf{x}_t|\mathbf{d}_{n,c})) + \epsilon_\theta(\mathbf{x}_t|f_c))$
                update $\mathbf{x}_{t-1}$ with Eq. 6 of the main text
            end for
            $\hat{\mathbf{x}}_i^{n,c} \leftarrow \mathbf{x}_0$
        end for
    end for
end for
train the aggregated model $\mathbf{w}_g$ with $\mathcal{L}_{agg}(\hat{\mathbf{x}}_i^{n,c}, c)$.
return $\mathbf{w}_g$ and $\{\hat{\mathbf{x}}_i^{n,c}\}$
```

, where λ is defined in Assumption 1. Next, we focus on the integral term within Eq. 6:

$$\begin{aligned}&\int p_n(\mathbf{x}) \log \frac{p_{\epsilon_\theta}(\mathbf{d})}{p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x})} d\mathbf{x} \\ &= \int p_n(\mathbf{x}) \log p_{\epsilon_\theta}(\mathbf{d}) d\mathbf{x} - \int p_n(\mathbf{x}) \log p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x}) d\mathbf{x} \\ &= \mathbb{E}(\log p_{\epsilon_\theta}(\mathbf{d})) - \int p_n(\mathbf{x}) \log p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x}) dx \end{aligned} \tag{7}$$

Therefore, from Eq. 6 and Eq. 7, we have :

$$\begin{aligned} KL(p_n(\mathbf{x}) \| p_{\epsilon_\theta}(\mathbf{x}|\mathbf{d})) <&\lambda + \mathbb{E}(\log p_{\epsilon_\theta}(\mathbf{d})) \\ &- \int p_n(\mathbf{x}) \log p_{\epsilon_\theta}(\mathbf{d}|\mathbf{x}) dx \end{aligned} \tag{8}$$

2 EXPERIMENTAL SETTING DETAILS

In this section, we detail the experimental settings of the proposed method that couldn't be elaborated on in the main text due to the space limitations, primarily comprising three parts: 1) **Dataset Details.** 2) **Client Partition Details.** 3) **Implementation Details.**

2.1 Dataset Details

Our experiments are conducted on three datasets: **DomainNet** [8], **OpenImage** [4] and **NICO++** [20]. **DomainNet** comprises six domains: *clipart, infograph, painting, quickdraw, real*, and *sketch*. Each domain has 345 categories. Following the partition in FedDISC [17], according to the hierarchy of categories provided by OpenImage, **OpenImage** is partitioned into 20 supercategories, with each supercategory comprising 6 fine-grained subclasses, serving as the six data domains for each category. The detailed partition of OpenImage is demonstrated in Table 1. **NICO++** involves 60 categories, with each category having six common domains shared across categories and six unique domains specific to each category. These two scenarios are respectively referred to as the **Unique NICO++** (NICO++_U) and **Common NICO++** (NICO++_C) datasets. For instance, in Common NICO++, both the *Cat* and *Dog* categories encompass 6 data domains: *autumn, dim, grass, outdoor, rock*, and *water*. In Unique NICO++, the *Cat* class comprises 6 unique data domains: *Eating, In Cave, In Mud, Jumping, Maine Cat*, and *Walking*, while the *Dog* class comprises 6 distinct data domains: *Lying, Pug Dog, Running, Sticking Out Tongue, Teddy Dog*, and *Wearing Clothes*.

The example images of each dataset are presented in figure 2. As mentioned in the main text, this figure clearly illustrates the emphases on the partition of data domains across different datasets is different. DomainNet primarily focuses on image style, OpenImage concentrates on fine-grained subcategories within each supercategory, Common NICO++ prioritizes image backgrounds, and Unique NICO++ places its emphasis on specific object attributes. The datasets we employ comprehensively simulate various types of differences that may exist among clients, thereby further enhancing the practicality of the proposed method.

2.2 Client Partition Details

Client Partition. The client partitioning is primarily aimed at reflecting the non-IID of data across various clients. In federated learning, there are primarily two types of non-IID data: feature distribution skew and label distribution skew [3]. We address these two scenarios separately in our client partition. For feature distribution skew, we conduct experiments on all four datasets. For each dataset, we allocate the six data domains of all categories to six clients, with each client possessing data from one unique domain for all categories. Regarding label distribution skew, experiments are conducted on Common NICO++ and Unique NICO++ datasets. Each 10 categories of the total 60 categories is grouped, resulting in six clients. Each client owns all data from 10 categories. As mentioned in the main text, there is no data overlap between clients in all partitions. Therefore, our partitioning maximizes the degree of non-IIDness among client data and considers various non-IID scenarios.

Supercategory	Baked Goods	Bird	Building	Carnivore	Clothing	Drink	Fruit	Furniture	Home appliance	Human body
Client0	Pretzel	Woodpecker	Convenience Store	Bear	Shorts	Beer	Apple	Chair	Washing Machine	Human Eye
Client1	Bagel	Parrot	House	Leopard	Dress	Cocktail	Lemon	Desk	Toaster	Skull
Client2	Muffin	Magpie	Tower	Fox	Swimwear	Coffee	Banana	Couch	Oven	Human Mouth
Client3	Cookie	Eagle	Office Building	Tiger	Brassiere	Juice	Strawberry	Wardrobe	Blender	Human Ear
Client4	Bread	Falcon	Castle	Lion	Tiara	Tea	Peach	Bed	Gas Stove	Human Nose
Client5	Croissant	Sparrow	Skyscraper	Otter	Shirt	Wine	Pineapple	Shelf	Mechanical Fan	Human Foot
Supercategory	Kitchen Utensil	Land Vehicle	Musical Instrument	Office Supplies	Plant	Reptile	Sports Equipment (Ball)	Toy	Vegetable	Weapon
Client0	Spatula	Ambulance	Drum	Pen	Maple	Dinosaur	Football	Doll	Potato	Knife
Client1	Spoon	Cart	Guitar	Poster	Willow	Lizard	Tennis Ball	Balloon	Carrot	Axe
Client2	Fork	Bus	Harp	Calculator	Rose	Snake	Baseball	Dice	Broccoli	Sword
Client3	Knife	Van	Piano	Whiteboard	Lily	Tortoise	Golf Ball	Flying Disc	Cabbage	Handgun
Client4	Whisk	Truck	Violin	Box	Common Sunflower	Crocodile	Rugby Ball	Kite	Bell Pepper	Shotgun
Client5	Cutting Board	Car	Accordion	Envelope	Houseplant	Sea Turtle	Volleyball	Teddy Bear	Pumpkin	Dagger

Table 1: The client partition on the OpenImage dataset.

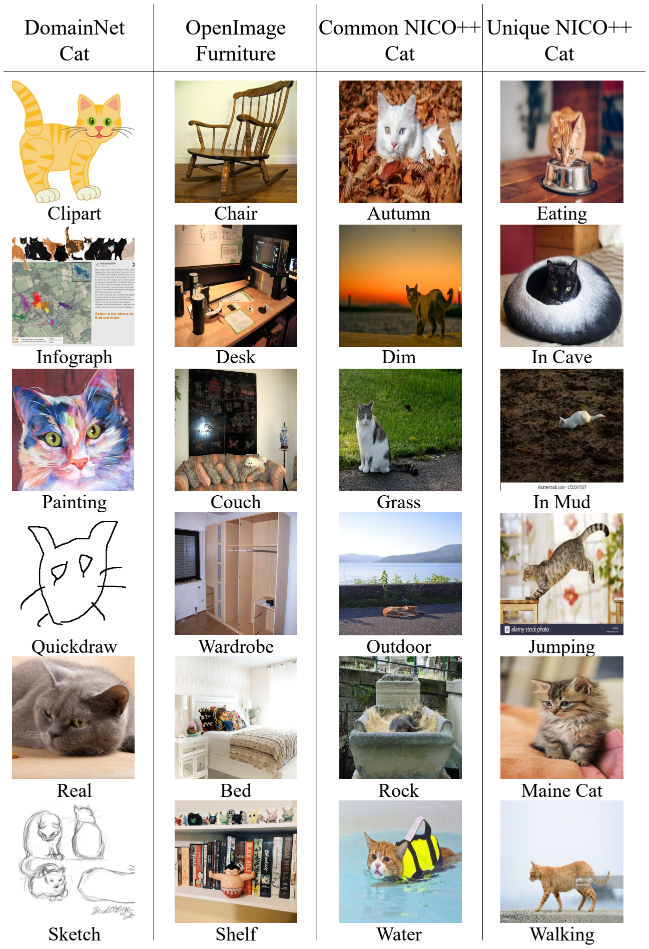

Figure 2: The example images of used datasets.

The Number of Images. The number of images on each client is important in our experimental setting, since we need to compare the performance of the proposed method with **Ceiling**, the performance ceiling of centralized training, involving the direct comparison between the synthetic dataset and original client dataset. Considering the cost of generation, we set the number of images generated with the guidance of each description to 30 in most experiments except the ablation experiments about the number of images in the Section 4.3 of the main text. The total number of images in each category of the synthetic dataset is 180. To ensure the fairness in comparing the synthetic dataset with the original client dataset, the maximum number of images for each category in each client local dataset is also set to 30, as same as the image number for each category in each data domain of the synthetic dataset.

The Number of Clients. Regarding the number of clients, it's worth noticing that in FedDEO, each client is entirely independent of the other clients. When the total number of images of the synthetic dataset is consistent, increasing the number of clients do not introduce interference between clients and affect the performance of the proposed method. Therefore, in the majority of our experiments, we set the number of clients to 6.

Nevertheless, we still conduct experiments related to the number of clients to demonstrate the practicality of the proposed method for a large number of clients. Following the commonly used Dirichlet distribution in partitioning non-IID clients [2], in the feature distribution skew scenario, for the 6 domains of the Common NICO++ and Unique NICO++ datasets, we sample 5, 10, and 30 clients per domain from $Dirichlet(\alpha = 1.0)$ according to the categories. Consequently, the total number of clients changes from the original 6 to 30, 60, and 180. The clients simultaneously exhibit feature distribution skew and label distribution skew. The number of images in each category of the client local dataset remains 30. To ensure fairness in comparison across different numbers of clients, the total number of images in each category of the synthetic dataset remains 180, shifting the number of the generated images guided by each description from 30 to 6, 3, and 1. For example, when the number of clients is 180, each description trained on each client guides the generation of 1 image in the synthetic dataset on the server. We present detailed experimental results and analysis in the subsequent supplementary experiments section.

2.3 Implementation Details

In our experiments, we mainly use ResNet-18 [1] as the model structure of the aggregated model. The pre-trained DM we mainly used is *Stable-diffusion-v1.5* from the *HuggingFace* model repository, which includes a corresponding CLIP text encoder used in our method to extract text features f_c for the name of each category c. We also use *Stable-diffusion-v2.1* from the *HuggingFace* model

	Common NICO++						
Number of Clients	autumn	dim	grass	outdoor	rock	water	average
6	**71.03**	58.02	73.33	**68.53**	68.16	63.04	67.01
30	69.37	**59.17**	**73.75**	68.44	**69.56**	63.07	**67.22**
60	70.48	57.69	72.65	67.18	68.49	**63.69**	66.69
180	69.13	58.91	72.52	67.64	68.77	62.39	66.56
	Unique NICO++						
Number of Clients	domain 0	domain 1	domain 2	domain 3	domain 4	domain 5	average
6	81.25	86.19	**82.94**	**79.94**	**83.85**	**80.27**	**82.40**
30	**82.45**	**86.93**	81.79	78.12	81.25	79.81	81.72
60	81.26	85.02	81.51	79.72	82.95	79.32	81.63
180	81.78	85.21	80.60	78.46	82.63	78.98	81.27

Table 2: The ablation experiments regarding the number of clients on Common NICO++ and Unique NICO++.

repository and the pre-trained *Latent Diffusion Model* [11] from *Github*. The *Stable-diffusion-v1.5* and *Stable-diffusion-v2.1* are pre-trained on the LAION-5B dataset [12] and the *Latent Diffusion Model* is pre-trained on the LAION-400M dataset [13]. Both datasets are large-scale image-text paired datasets, covering a wide range of image distributions encountered in daily life to satisfy Eq. 1. All experiments are conducted with four NVIDIA GeForce RTX 3090 GPUs. Regarding specific hyperparameters, we use the SGD optimizer with a learning rate of 10. When the learning rate is too low, the changes in the descriptions are minimal, making it challenging to achieve effective learning. As mentioned above, the number of images R generated per descriptor is set to 30, and he number S of training epochs for client descriptors is 10 in most of our experiments except the related ablation experiments. The relevant hyperparameters for the diffusion generation process are set to their default values. The number of inference steps is 50, and the guidance scale of the generation is 3.

3 SUPPLEMENTARY EXPERIMENTS

We primarily conduct supplementary experiments targeting three aspects that are not detailed in the main text because of the space limitation: 1) **Ablation Experiments.** Experiments regarding the number of clients and the used pre-trained diffusion models. 2) **Supplementary of the Discussions.** Experiments regarding the discussions in the main text, including experiments on communication , computation, and privacy. 3) **More visualization experiments.** Experiments to further illustrate the quality and diversity of the synthetic dataset.

3.1 Ablation Experiments

As mentioned in the main text, to further demonstrate the performance of the proposed method, we conduct ablation experiments about the number of images in the synthetic dataset, the number of training epochs for the local descriptions, the used pre-trained DMs, and the number of clients. The first two types of ablation experiments are already presented in the main text. In this section, we focus on discussing the remaining two factors that impact the performance of the proposed method.

The Number of Clients. Following the experimental setting in Section 2.2 of the supplementary materials, we conduct ablation experiments on the number of clients in Common NICO++ and Unique NICO++ datasets. The experimental results are presented in Table 2. From the results, it can be observed that increasing the number of clients has almost no effect on the performance of our method. This is because the proposed method independently treats each client. Each client trains its local description separately, and the generation of each image on the server is also guided separately by each local descriptions. These experimental results further underscore the practicality of the proposed method in scenarios with a large number of clients.

	Common NICO++						
	autumn	dim	grass	outdoor	rock	water	average
SD-v1.5	71.03	58.02	**73.33**	68.53	68.16	63.04	67.01
SD-v2.1	**71.62**	**58.51**	72.56	**69.93**	**69.24**	**63.16**	**67.50**
LDM	67.82	56.25	69.89	66.45	64.88	60.97	64.37
	Unique NICO++						
	domain 0	domain 1	domain 2	domain 3	domain 4	domain 5	average
SD-v1.5	**81.25**	86.19	82.94	79.94	83.85	80.27	82.40
SD-v2.1	81.12	**86.57**	**83.7**	**80.87**	**84.63**	**80.94**	**82.97**
LDM	76.21	84.75	81.77	79.93	82.91	79.42	80.83

Table 3: The ablation experiments regarding the used diffusion model on Common NICO++ and Unique NICO++.

Size of Downloaded Files			
CLIP-based Federated Fine-tuning	FedDISC	FGL	FedDEO
1.71*GB*	1.74*GB*	1.88*GB*	1.72*GB*

Table 4: Comparison about the download communication costs, where CLIP-based Federated Fine-tuning Methods have different problem settings and are only used for reference. We directly compare the total size of files that clients need to download from the server across different methods.

Client Computation Costs (GFLOPS)			
CLIP-based Federated Fine-tuning	FedDISC	FGL	FedDEO
493.5	334.73	**227.34**	365.72

Table 5: Comparison about the client computation costs, where CLIP-based Federated Fine-tuning Methods have different problem settings and are only used for reference.

The Used Diffusion Models. Due to the utilization of the synthetic datasets for training the aggregated model, the pre-trained diffusion model employed for generating the synthetic datasets constitutes a crucial component of our method. However, this does not imply that our method heavily relies on the specific diffusion model. On the one hand, as mentioned in Section 3.1 of the main text, we only require some overlap between the diffusion models' distributions and the client distributions, which is readily achievable with the pre-trained diffusion models fine-tuned on large-scale image datasets such as LAION-5B [12]. Even if the clients focus on some professional fields, such as medical images, it is entirely feasible to pre-train the specialized DMs on the server firstly. On the other hand, compared to other diffusion-based OSFL methods [17–19], the process of training client descriptions in our method serves as an effective means to facilitate the adaptation of the diffusion model to client data, resembling various federated fine-tuning methods [9, 14, 16]. Consequently, this significantly alleviates our method's reliance on the used diffusion models.

	FID between Different Datasets							
	Clipart	*Infograph*	*Painting*	*Quickdraw*	*Real_A*	*Real_B*	*Sketch*	*Synthetic*
Real_A	284.39	228.44	221.01	297.37	0	131.48	195.74	133.22

Table 6: Comparison of the FID between datasets.

To substantiate this claim, we conduct ablation experiments on three commonly used diffusion models, *Stable-diffusion-v1.5*, *Stable-diffusion-v2.1*, and *Latent Diffusion Model* (LDM). The experimental results are provided in Table 3. From the table, it is evident that: 1) Different diffusion models are capable of training high-performing aggregated models using our method. 2) Despite the utilization of LDM to train the aggregated models outperforming the Ceiling, its overall performance is inferior. This is primarily attributed to the relatively smaller scale of LAION-400M used during the pre-training of LDM, resulting in lower overlap with the client distributions. 3) The best performance is achieved with *Stable-diffusion-v2.1*, mainly because, although both *Stable-diffusion-v1.5* and *Stable-diffusion-v2.1* are pre-trained on LAION-5B, *Stable-diffusion-v2.1* has more parameters and relatively better generation quality. These results demonstrate that the proposed method is not limited to the specific used diffusion model, thereby validating its practicality.

3.2 Supplementary of the Discussions

In the discussion section of the main text, we present quantitation of the uploaded communication and visualization experiments about the privacy-sensitive information of the clients. In this section, we conduct supplementary experiments to further discuss the performance of the proposed method in terms of communication costs, computation costs, and privacy concerns.

Communication Costs. In the main text, we categorize the communication costs involved in the federated learning into the upload communication costs and the download communication costs. We clearly demonstrate that our method exhibits significant advantages concerning the upload communication costs through the quantification in the main text. As for downloading, the analysis in the main text also demonstrate that the download communication costs introduced by the proposed method is entirely acceptable. Here, we delve deeper into the discussion regarding the download communication costs and further prove the performance of the proposed method regarding communication. In Table 4, we quantify the download communication costs in our proposed method and compared it with other federated learning methods based on foundation models. It is evident from the table that the download communication cost in FedDEO is less than FGL [19] and FedDISC [17], which need to download BLIP [5], CLIP [10], and some prototypes, with almost no difference compared to other CLIP-based federated fine-tuning methods such as FedAPT [14]. It is worth noting that lightweight diffusion models have become a major research focus regarding the diffusion model [6, 7, 21], which can be readily applied in our method to further reduce the download communication costs. These results demonstrate the performance of the proposed method in terms of communication cost.

Computation Costs. As mentioned in the main text, we divide the computation costs into the server computation costs and the client computation costs. Regarding the server computation costs, our method is essentially consistent with other diffusion-based OSFL methods, such as FedDISC, FedCADO, and FGL, hence we do not perform quantitative comparisons. Although these methods entail significant server computational costs, in the federated learning framework, it is typically assumed that the server possesses sufficient computational resources to complete the model aggregation task. Therefore, the increase in server computation costs in the diffusion-based OSFL methods is acceptable. Regarding clients, although the client computation costs involved in FedDEO is inevitably higher than that of FedDISC and FGL due to the necessity of training descriptions on the clients. Since FedDEO fixes all parameters of the DM and only trains the local descriptions, the client computation costs of FedDEO is comparable to various federated fine-tuning methods [9, 14, 16], making it entirely acceptable, which is demonstrated by the detailed quantitation of client computational costs in Table 5. Furthermore, there are substantial researches regarding how to perform the inference and training of diffusion models in scenarios with severely limited computation resources, such as on mobile devices [7, 21]. These efforts can also be readily integrated into our proposed method, effectively reducing the client computation costs. Therefore, our method does not exhibit disadvantages in terms of computation costs compared to other methods.

	Common NICO++						
μ	autumn	dim	grass	outdoor	rock	water	average
0	**71.03**	**58.02**	**73.33**	**68.53**	68.16	**63.04**	**67.01**
0.1	70.45	57.68	73.17	68.21	**68.26**	62.97	66.79
0.3	68.53	55.02	71.34	62.30	64.39	61.57	63.85
0.5	60.28	49.14	62.80	57.36	58.62	52.47	56.77
	Unique NICO++						
μ	domain 0	domain 1	domain 2	domain 3	domain 4	domain 5	average
0	81.25	**86.19**	**82.94**	**79.94**	83.85	80.27	**82.40**
0.1	**81.5**	85.71	82.73	79.74	**83.31**	**80.38**	82.22
0.3	77.87	82.22	79.08	75.75	80.86	78.79	79.095
0.5	73.58	75.66	75.01	71.55	76.25	72.43	74.08

Table 7: Ablation experiments of the noised descriptions.

Privacy Concerns. Based the discussion in the main text regarding privacy, in this section, we make further discussions regarding the privacy concerns, including the quantitation of the datasets based on FID, the upload of noised descriptions, and additional visualization experiments.

To quantify the privacy-leakage risk of the synthetic datasets, we compared the Frechet Inception Distance (FID) between the synthetic dataset and the client local datasets for categories that may contain privacy-sensitive information, such as the *"Face"* category in DomainNet. Specifically, because the *"Real"* domain is most relevant to privacy leakage, we divided the dataset of the *"Real"* domain into two non-overlapping parts, denoted as *Real_A* and *Real_B*, to represent the FID between the datasets without privacy leakage but with the same style. And we calculated the FID between *Real_A* and datasets from other domains, as well as the *"Real"* domain of the *Synthetic* dataset. Additionally, we collect photos of the same person with different styles from the internet, divide these photos into two groups, and compute the FID between these groups, serving as a threshold for potential privacy leakage. We conduct several experiments and find that this threshold of FID

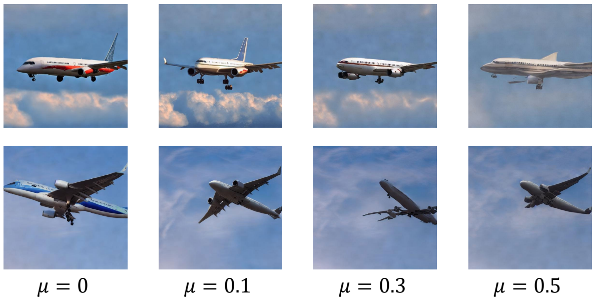

Figure 3: Visualization of samples generated by adding varying scales of noise to descriptions. Samples within the same row are generated with the same initial noise.

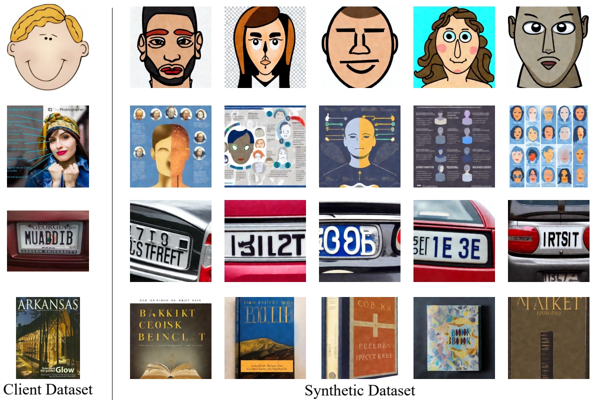

Figure 4: The visualization of privacy-sensitive information-related categories.

is approximately around 80. These results are presented in Table 6. From the table, it can be observed that the FID between Real_A and the synthetic dataset is neither too low, which could suggest the presence of identical images leading to privacy leakage, nor too high, indicating the generation of images with significantly different styles. These quantitation results demonstrate that with the guidance of local descriptions, the server can generate the synthetic dataset complies with the client distributions without leaking the the privacy-sensitive information.

Furthermore, as a commonly used method to implement differential privacy [15], we add noise to the local descriptions before uploading them. To obtain the noised description $\hat{\mathbf{d}}_{n,c}$ from the trained local description $\mathbf{d}_{n,c}$, we use the following formula:

$$\hat{\mathbf{d}}_{n,c} = \frac{\mathbf{d}_{n,c} + \mu\epsilon}{1+\mu} \tag{9}$$

where ϵ is the noise sampled from the standard Gaussian distribution $\mathcal{N}(0, \mathcal{I})$, and μ is a hyperparameter used to control the scale of the added noise. We conduct ablation experiments on Common NICO++ and Unique NICO++ for different μ, and the results are presented in Table 7. The experimental results show that the minor scales of noise addition do not significantly affect the guidance provided by the descriptions. Even with increased scale of noise addition, the performance of the trained aggregated model is close to the performance ceiling of centralized training. Additionally, we visualized the generation results under different μ, as shown in Figure 3. The results indicate that although increasing the scale of noise addition result in the partial loss of information in the descriptions and erroneous guidance, consequently resulting in relatively indistinct generated images, the descriptions still guide the diffusion model in generating synthetic data with accurate semantics and predominantly consistent style. These quantitation and visualization results demonstrate that the proposed method has strong robustness against noise addition, enabling the differential privacy through the noise-added local descriptions without significantly compromising performance.

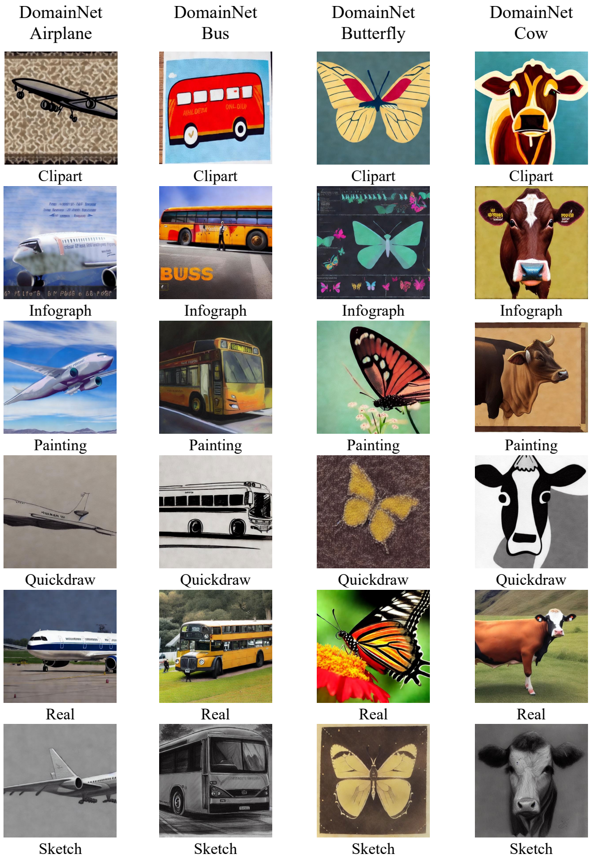

Figure 5: Visualization of generated samples on DomainNet.

Besides, similar to the visualization experiments in the main text, we conduct additional visualization experiments to illustrate the performance of the proposed method in privacy protection. The visualization results are shown in Figure 4, further illustrate that the synthetic datasets only share similar styles and identical semantics with the original client datasets. It is almost impossible to extract specific privacy-sensitive information from the descriptions, which aiming to characterize the overall distribution.

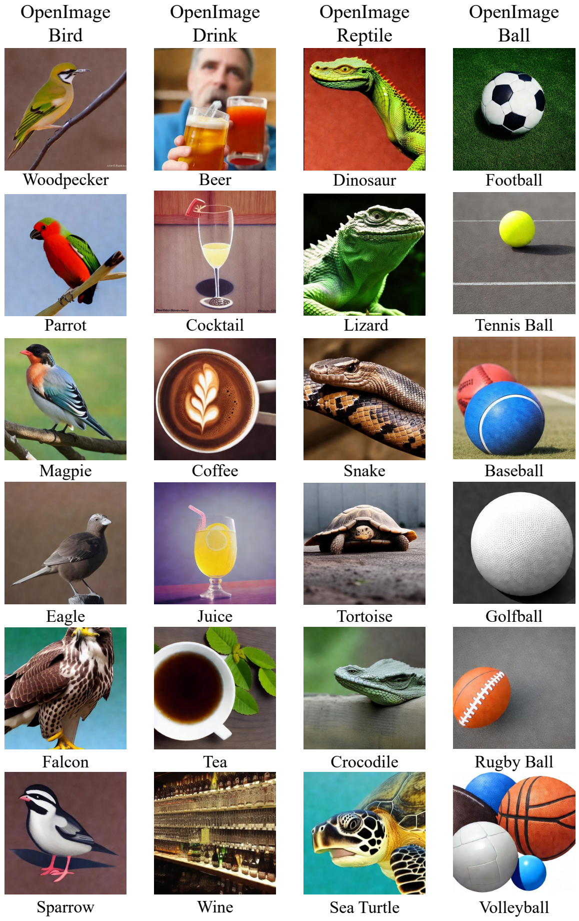

Figure 6: Visualization of generated samples on OpenImage.

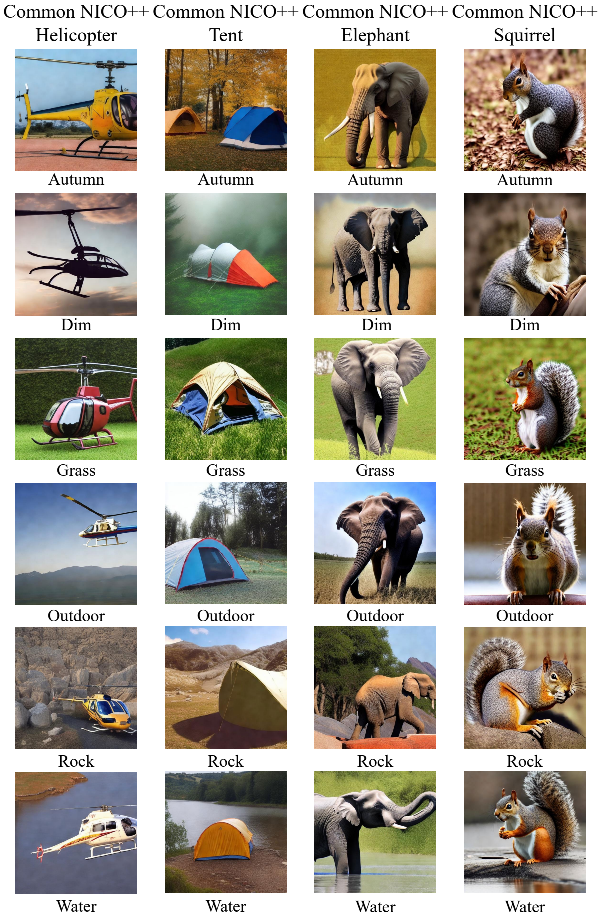

Figure 7: Visualization of generated samples on Common NICO++.

3.3 More Visualization Experiments

Similar to the main text, we conduct additional visualization experiments to illustrate the quality and diversity of the synthetic dataset generated by the proposed method. The experimental results are presented in Figure 5, Figure 6, Figure 7, and Figure 8. These visualizations further illustrate that the generated synthetic dataset complies with different client distributions when there are differences in style, subcategory, or background among clients, while being semantically correct. The synthetic dataset demonstrates diversity and quality comparable to the original client datasets, which directly contributes to the performance of the trained aggregated model outperforming the performance ceiling of centralized training.

REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[2] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. 2019. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335* (2019).

[3] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning* 14, 1–2 (2021), 1–210.

[4] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. 2020. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision* 128, 7 (2020), 1956–1981.

[5] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597* (2023).

[6] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. 2023. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 17535–17545.

[7] Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. 2024. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. *Advances in Neural Information Processing Systems* 36 (2024).

[8] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*. 1406–1415.

[9] Chen Qiu, Xingyu Li, Chaithanya Kumar Mummadi, Madan Ravi Ganesh, Zhenzhen Li, Lu Peng, and Wan-Yi Lin. 2023. Text-driven Prompt Generation for Vision-Language Models in Federated Learning. *arXiv preprint arXiv:2310.06123* (2023).

[10] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark,

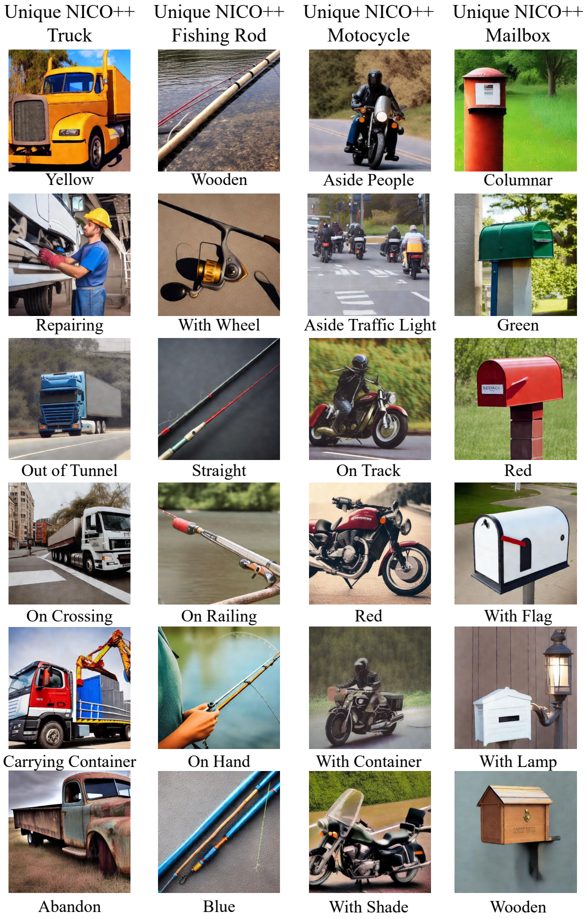

Figure 8: Visualization of generated samples on Unique NICO++.

et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

[11] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10684–10695.

[12] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. 2022. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems* 35 (2022), 25278–25294.

[13] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. 2021. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114* (2021).

[14] Shangchao Su, Mingzhao Yang, Bin Li, and Xiangyang Xue. 2024. Federated Adaptive Prompt Tuning for Multi-Domain Collaborative Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 15117–15125.

[15] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE transactions on information forensics and security* 15 (2020), 3454–3469.

[16] Fu-En Yang, Chien-Yi Wang, and Yu-Chiang Frank Wang. 2023. Efficient model personalization in federated learning via client-specific prompt generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 19159–19168.

[17] Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. 2023. Exploring One-shot Semi-supervised Federated Learning with A Pre-trained Diffusion Model. *arXiv preprint arXiv:2305.04063* (2023).

[18] Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. 2023. One-Shot Federated Learning with Classifier-Guided Diffusion Models. *arXiv preprint arXiv:2311.08870* (2023).

[19] Jie Zhang, Xiaohua Qi, and Bo Zhao. 2023. Federated generative learning with foundation models. *arXiv preprint arXiv:2306.16064* (2023).

[20] Xingxuan Zhang, Linjun Zhou, Renzhe Xu, Peng Cui, Zheyan Shen, and Haoxin Liu. 2022. Nico++: Towards better benchmarking for domain generalization. *arXiv preprint arXiv:2204.08040* (2022).

[21] Yang Zhao, Yanwu Xu, Zhisheng Xiao, and Tingbo Hou. 2023. MobileDiffusion: Subsecond Text-to-Image Generation on Mobile Devices. *arXiv preprint arXiv:2311.16567* (2023).