CollaFuse: Collaborative Diffusion Models

SIMEON ALLMENDINGER*†, University of Bayreuth, Germany and Fraunhofer FIT, Germany
DOMENIQUE ZIPPERLING*, University of Bayreuth, Germany and Fraunhofer FIT, Germany
LUKAS STRUPPEK, Technical University of Darmstadt, Germany
NIKLAS KÜHL, University of Bayreuth, Germany and Fraunhofer FIT, Germany

In the landscape of generative artificial intelligence, diffusion-based models have emerged as a promising method for generating synthetic images. However, the application of diffusion models poses numerous challenges, particularly concerning data availability, computational requirements, and privacy. Traditional approaches to address these shortcomings, like federated learning, often impose significant computational burdens on individual clients, especially those with constrained resources. In response to these challenges, we introduce the novel approach `CollaFuse` for distributed collaborative diffusion models inspired by split learning. Our approach facilitates collaborative training of diffusion models while alleviating client computational burdens during image synthesis. This reduced computational burden is achieved by retaining data and computationally inexpensive processes locally at each client while outsourcing the computationally expensive processes to shared, more efficient server resources. Through experiments on the common datasets CelebA, CIFAR-10, and Animals-with-Attributes2, our approach demonstrates enhanced performance while decreasing information disclosure as it reduces the necessity for sharing raw data. These capabilities hold significant potential across various application areas, including the design of edge computing solutions. Thus, our work advances distributed machine learning by contributing to the evolution of collaborative diffusion models.

JAIR Associate Editor: Chang Xu

JAIR Reference Format:
Simeon Allmendinger, Domenique Zipperling, Lukas Struppek, and Niklas Kühl. 2026. CollaFuse: Collaborative Diffusion Models. *Journal of Artificial Intelligence Research* 86, Article 55 (August 2026), 23 pages. DOI: 10.1613/jair.1.20934

1 Introduction

Generative artificial intelligence (GenAI) methods exhibit astonishing results in generating images, among other modalities like music (Mariani et al. 2024) and video (Brooks et al. 2024; Singer et al. 2023) with promising applications in areas like healthcare (Sun et al. 2023) to help reduce biases (Ramachandranpillai et al. 2024). Recent computer vision advancements primarily rely on diffusion models (Ho et al. 2020; Y. Song and Ermon 2020) that generate synthetic images from random noise through iterative denoising steps. We formally introduce diffusion models and related work in section 2.1. In contrast to more traditional approaches (Goodfellow et al. 2020; Kingma and Welling 2014), diffusion models excel in providing high sample quality and strong mode coverage (Dhariwal and Nichol 2021; Kazerouni et al. 2023; Nichol and Dhariwal 2021). However, the strides in GenAI require large

*Both authors contributed equally to this research.
†Corresponding author.

Authors' Contact Information: Simeon Allmendinger, ORCID: 0009-0005-8741-7734, simeon.allmendinger@uni-bayreuth.de, University of Bayreuth, Bayreuth, Germany and Fraunhofer FIT, Bayreuth, Germany; Domenique Zipperling, ORCID: 0009-0003-4598-9051, domenique.zipperling@uni-bayreuth.de, University of Bayreuth, Bayreuth, Germany and Fraunhofer FIT, Bayreuth, Germany; Lukas Struppek, ORCID: 0000-0003-0626-3672, lukas.struppek@tu-darmstadt.de, Technical University of Darmstadt, Darmstadt, Germany; Niklas Kühl, ORCID: 0000-0001-6750-0876, kuehl@uni-bayreuth.de, University of Bayreuth, Bayreuth, Germany and Fraunhofer FIT, Bayreuth, Germany.

DOI: 10.1613/jair.1.20934

amounts of data, and the generation process itself is computationally expensive due to the multiple required denoising steps.

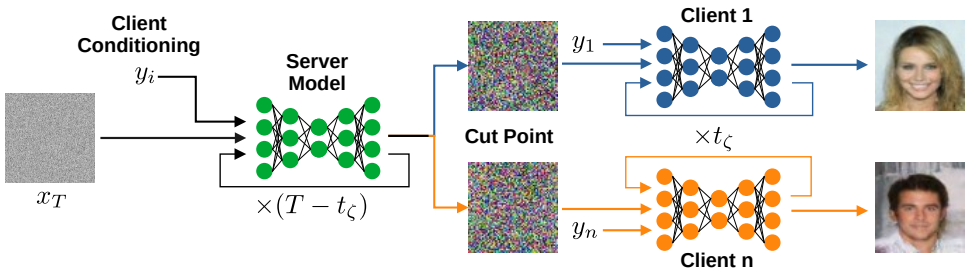

Fig. 1. Overview of CollaFuse for collaborative image synthesis by splitting the denoising process between the server and clients. Based on client-specific conditioning y_i, the first $T - t_\zeta$ denoising steps are run on a trusted server model, while the remaining t_ζ denoising steps are run locally by the i-th client. Thereby, external computing resources can be utilized while keeping the clients' raw data private.

While large companies possess the necessary data and computational resources for training diffusion models, smaller organizations and private clients may face challenges in providing the required resources, limiting their ability to train and implement recent GenAI models (Zajac et al. 2023). This might lead to high dependency on a few key players and even prevent the application of such models completely due to local data protection regulations. For example, the popular Stable Diffusion v2 has been trained on 256 A100 GPUs for about 200,000 hours (Rombach et al. 2022). But even smaller models still set notably high hardware and data requirements.

To address data and resource availability constraints, organizations and clients can join forces to train machine learning models collaboratively with other clients in a decentralized way. A prominent representative of collaborative training is federated learning (McMahan et al. 2017). Conceptually, each client trains an individual model locally on their private data. After several training steps, the model parameters are sent to the server to build a global model by aggregating the individual weights. The server distributed this global model back to all clients. As the necessity for each client to train and share an entire model remains, high computational resources for each client are entailed (Thapa et al. 2022). Furthermore, in addition to privacy risk from the generative GenAI models in general (Hintersdorf, Struppek, Brack, et al. 2024), federated learning introduces new potential privacy risks (Shokri et al. 2017; Zhu et al. 2019). As an alternative learning paradigm, split learning (Gupta and Raskar 2018) supports collaborative model training by splitting the model into server-side and client-side components. In the conventional split learning setup, clients share only intermediate network activations with the server side, where the final computations are performed. Unlike federated learning, split learning reduces the computational load on clients and enhances privacy protection by sharing only intermediate representations instead of raw data or model weights (Duan et al. 2023). We provide an overview of collaborative training for generative models in section 2.2. Our analysis reveals that current approaches rely mostly on federated learning. By not utilizing split learning with diffusion models, they overlook the privacy and resource benefits of split learning, especially from the clients' perspective.

Our Approach: In addressing the challenges posed by the data, computation, and privacy requirements of diffusion-based GenAI, we present our collaborative diffusion models in section 3. We introduce the novel collaborative learning and inference approach `CollaFuse` tailored specifically for diffusion models. Drawing inspiration from the split learning framework, our approach divides the iterative denoising steps of diffusion models into two components. The computation of the initial denoising steps is carried out by a shared model on a server, with limited information disclosure due to the inherent noise in the training data and intermediate

generated samples. Subsequently, the client's model then performs the remaining denoising steps, which are usually significantly fewer than the denoising steps on the server side. `CollaFuse` also allows for personalized image conditioning by incorporating attribute labels during the generation. Our empirical results demonstrate that our collaborative diffusion approach improves the image quality compared to a setting where each client trains its own local diffusion model. By sharing a server model that performs most of the computationally heavy denoising steps, the computational burdens for each client are comparably small. At the same time, clients can better approximate their individual data distribution, which enables them to generate better characteristic features. Beyond these performance improvements, we also conduct dedicated privacy analyses, including attribute inference (Fredrikson, Lantz, et al. 2014) and inversion attack (Fredrikson, Jha, et al. 2015) experiments, to quantify information disclosure at different collaboration cut points. This allows us to empirically evaluate how collaborative diffusion can balance image fidelity with privacy protection.

The proportion of denoising steps carried out on the server and client sides, respectively, is controlled by a single parameter called the cut point. The higher the cut point, the more steps are computed on the client side. Whereas our approach is in principle also applicable to other diffusion model architectures, we focus our experiments in section 4 on the common Denoising Diffusion Probabilistic Models (DDPM) (Ho et al. 2020) and Latent Diffusion Models (LDMs) (Rombach et al. 2022). In summary, we make the following contributions:

(1) We introduce `CollaFuse`, to the best of our knowledge the first split-denoising (timestep-partitioned) collaborative diffusion framework, which consists of a shared server component trained by multiple clients without revealing their original training data to other clients or the server.
(2) `CollaFuse` allows clients to outsource most of their computationally expensive denoising steps during training and inference to a shared server model.
(3) `CollaFuse` improves the image quality compared to the setting where each client trains a local diffusion model on its own data.

2 Background and Related Work

We start by introducing diffusion models for generative image synthesis. We further describe related distributed collaborative machine learning approaches, such as federated and split learning, and their utilization for image synthesis or generative AI more generally.

2.1 Diffusion Models

The Denoising Diffusion Probabilistic Models (DDPMs) (Ho et al. 2020) mark a significant advancement in generative image synthesis consisting of a diffusion and denoising process. The diffusion process is a Markov chain with T timesteps that transforms a training image x_0 to a noisy image x_T that follows a random Gaussian distribution. The diffusion process of an image x_t at time step t is mathematically defined as

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t}\epsilon, \text{ with } t = 1, \ldots, T. \tag{1}$$

Here, α_t denotes the variance schedule, and ϵ is the added Gaussian noise. A denoising network $\epsilon_\theta(x_t, t)$ with parameters θ is then trained to reverse the diffusion process and predict the noise added to the sample x_t during time step t. Most denoising networks are built upon the common U-Net (Ronneberger et al. 2015) architecture. With the denoising network, the image generation process, which iteratively removes the predicted noise $\epsilon_\theta(x_t, t)$ from the noisy sample x_t, can be defined as

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) \tag{2}$$

with $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. Based on the idea of Markov chains, the distribution of the intermediate noise predictions in the denoising process $p_\theta(\cdot)$ is defined by

$$p_\theta(x_{0:T}) = p(x_T) \cdot \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \tag{3}$$

with $p(x_T) = \mathcal{N}(x_T; \mathbf{0}, \mathbf{I})$. In DDPMs, the iterative application of U-Nets across T timesteps is a fundamental characteristic, enabling the model to refine noisy data into structured outputs progressively. The Imagen model (Saharia, Chan, Saxena, et al. 2022), as one of the most recognized text-conditioned diffusion models in the community, builds upon DDPMs. The authors employ a frozen text encoder and dynamic thresholding to generate photorealistic images conditioned by the text prompt y. The loss function of DDPMs can be expressed by

$$\mathcal{L} = \mathbb{E}_{t\sim\mathcal{U}[1,T],\ \epsilon\sim\mathcal{N}(0,\mathbf{I})}\left[\ \|\epsilon_\theta(x_t, t, y) - \epsilon\|_2^2\ \right] \tag{4}$$

The Imagen model additionally adds a guidance weight ω_t, which is integral to the denoising process. This guidance weight modulates the influence of the predicted noise $\epsilon_\theta(x_t, t, y)$ at each timestep t, enabling precise control over the image generation process, particularly in maintaining fidelity to the target distribution. For simplicity, we leave the explicit embedding process out of our notation and implicitly assume that all text labels have already been embedded before feeding the embeddings into the U-Net $\epsilon_\theta(\cdot)$.

LDMs (Rombach et al. 2022) extend DDPMs to the latent space of image representations, enhancing computational efficiency and scalability (Rombach et al. 2022). By applying the diffusion process in a lower-dimensional latent space, LDMs generate high-quality images with less computational overhead. Autoencoders encode images into latent representations, where denoising occurs, and then reconstruct the final image.

2.2 Collaborative Learning in Generative AI

Collaborative learning has become an essential paradigm for training models on distributed data while preserving privacy and reducing local computational costs. It enables multiple clients to jointly optimize model parameters without sharing raw data, forming the basis for methods such as federated and split learning.

Collaborative learning: Federated learning (FL) (McMahan et al. 2017) and split learning (SL) (Gupta and Raskar 2018) are among the most prominent approaches for training machine learning models collaboratively on distributed data sources. FL utilizes distributed data, with clients independently training models on their unique datasets. These models are subsequently shared, aggregated, and redistributed. The cycle repeats until the models converge. Conversely, SL divides a model among clients and a central server, decreasing the computational load on clients. Moreover, clients have the option to use FL for model aggregation, leading to the development of SplitFed learning (Thapa et al. 2022). These techniques have found applications in diverse fields such as the automotive industry (Schwermer, Bicer, et al. 2023), energy management (Schwermer, Buchberger, et al. 2022), and healthcare (Joshi et al. 2022), where they are combined with both discriminative (Nazir and Kaleem 2023) and generative AI approaches (Shen et al. 2023).

Collaborative learning in generative AI: Especially for image synthesis, FL and SL possess potential owing to the extensive volume of data involved. Before diffusion models took over as the predominant architecture for image synthesis, *Generative Adversarial Networks (GANs)* (Goodfellow et al. 2020) represented the most widely used generative architecture. GANs are composed of two components: a generator for synthesizing images and a discriminator trained to distinguish between real and synthetic images. Existing research on collaborative training of GANs demonstrates different ways to integrate these two components within the FL process. Hardy et al. (2019) introduced FL-GAN, adopting the standard FL procedure for both discriminator and generator. This vanilla approach was compared to MD-GAN, where FL is applied only to the discriminator, while the generator is trained directly by a central server. Building upon this foundation, Fan and P. Liu (2020) empirically analyzed

several strategies for synchronizing discriminators and generators across clients, demonstrating that the best results are achieved when synchronizing both components jointly. W. Li et al. (2022) improved FL-GANs by employing maximum mean discrepancy for generator updates, while more recent work addresses heterogeneity issues (Ma et al. 2024; Zhao et al. 2023). Because FL places substantial strain on computational and communication resources, new methods have been developed to alleviate these burdens. Lai et al. (2024) proposed an on-demand, quantized, energy-efficient federated diffusion approach to enhance the training efficiency of generative AI models in mobile edge networks by dynamically adjusting quantization, reducing energy and communication consumption while preserving data quality and diversity.

In addition, several studies focus on improving privacy in FL-GANs. Augenstein et al. (2020) proposed a differentially private federated GAN framework, while Veeraragavan and Nygård (2023) combined consortium blockchains with an efficient secret sharing algorithm to address trust-related weaknesses in existing solutions. Beyond FL, GANs have also been collaboratively trained using SL (Feng et al. 2023). Although not explicitly focused on distributed learning, Ohta and Nishio (2023) presented a privacy-preserving SL approach for GANs that could be extended to collaborative learning. Other work combines both paradigms, known as SplitFed learning, to leverage the strengths of each (Kortoçi et al. 2022; Yin et al. 2023).

While these studies demonstrate the feasibility of collaborative learning for GAN-based architectures, their direct transfer to diffusion models remains challenging. Diffusion models rely on iterative denoising and stochastic sampling processes that are computationally intensive and require consistent synchronization of parameters over thousands of steps. These characteristics complicate straightforward adaptation of existing FL and SL schemes, motivating the development of new collaborative frameworks tailored to diffusion-based generative models.

Collaborative learning of diffusion models: In the domain of diffusion models, research on collaborative training methods is still scarce. Jothiraj and Mashhadi (2024) made a first step and introduced the Phoenix technique for training unconditional diffusion models in a horizontal FL setting. Their objective is to address mode coverage issues often seen in distributed datasets that are not independent and identically distributed. Their data-sharing approach boosts performance by sharing only 4-5% of the data among clients, minimizing communication overhead. Personalization and threshold filtering techniques outperform comparison methods in terms of precision and recall, but fall short in image quality compared to the proposed technique. The paper suggests further exploration to enhance image quality in future work. Y. Song, H. Liu, et al. (2024) proposed enhancing FL with knowledge distillation by dynamically adjusting distillation weights, using data generated by a collaboratively trained diffusion model (FedAvg), thus addressing data heterogeneity and preserving privacy by eliminating the need for public datasets.

Moreover, the potential of FL for AI-generated content, especially for DDPMs, was demonstrated by Huang et al. (2024). The authors discuss three different approaches for diffusion models in FL settings. A parallel approach mimicking the conventional FL. A separate split approach combines FL with SL. As a third solution, the authors discussed a sequential approach in which one client receives the current model from the server, trains the model on its data, and then transmits the current version to the next client. The trained model returns to the server only after every client has trained the model once. Based on the sequential FL, a LoRA-based (Hu et al. 2022) federated fine-tuning scheme is designed and examined in more detail, demonstrating the advantages of faster convergence time and reduced memory consumption during the tuning process. Further, Goede et al. (2024) adapt the federated averaging algorithm for training DDPMs. Mendieta et al. (2025) trained diffusion models with one-shot FL under provable privacy budgets with differential privacy.

By mainly focusing on FL for GANs (Little et al. 2023), current literature has neglected the benefits of different collaborative paradigms and GenAI architectures so far. Combining DDPMs and SL promises various benefits, including reducing local resource requirements and increasing data privacy. Huang et al. (2024) is the only identified study utilizing SL for diffusion models. However, it combines SL with FL, which increases privacy risks compared to a standard SL approach, as client models are still shared. Moreover, it overlooks the impact of

model split configuration on factors such as performance. Our proposed approach for distributed collaborative image synthesis with diffusion models taps into these advantages and combines the research areas of diffusion models and collaborative learning. Focusing exclusively on split learning mitigates privacy risks by sharing only intermediate results. Additionally, we treat the timestep split configuration (the cut point determining how many denoising steps are performed client-side) as a hyperparameter and analyze its impact. We emphasize that CollaFuse is inspired by split learning but does not split a network by layers or activations; instead it partitions the reverse denoising trajectory across timesteps, so the privacy and communication intuitions of conventional split learning do not transfer directly.

3 Collaborative Diffusion Models

We now formally introduce our novel approach `CollaFuse` for enabling collaborative image generation with diffusion models. In our setting, a certain number k of clients $c \in C = \{c_1, c_2, ..., c_k\}$ wants to collaboratively train a diffusion model for image synthesis. Although we assume that each client has a dataset from a similar domain, e.g., facial images, the specific feature distribution may differ. To stay with the facial image example, client `A` may have a dataset of facial images with eyeglasses, whereas client `B`'s dataset consists only of faces without eyeglasses. All clients now want to train a shared U-Net ϵ_θ^s on the server that is available to each client and computes the initial denoising steps. Additionally, each client $c \in C$ trains an individual U-Net ϵ_θ^c that is maintained locally and computes solely the remaining denoising steps. For notation simplicity, we assume that θ denotes the weights of each individual model, so there exist no shared weights between the client models and the server model.

The computational split between server and clients is manually set by the cut point $t_\zeta \in [0, T]$ that specifies the number of denoising steps performed on the client side after $T - t_\zeta$ steps were computed by the shared server model. The cut point is set as a hyperparameter and kept fixed during training and inference. For $t_\zeta = 0$, all denoising steps are computed by the server, which is trained on the joint set of all clients' data. For $t_\zeta = T$, each client trains an individual diffusion model on its data that performs all denoising steps without any shared server model. The approximated data distribution of our collaborative denoising approach is formalized in Equation (5) with $p(x_T) = \mathcal{N}(x_T; 0, \mathbf{I})$:

$$p_{\theta^s,\theta^c}(x_{0:T}) = p(x_T) \prod_{t=t_\zeta+1}^{T} p_{\theta^s}(x_{t-1} \mid x_t) \cdot \prod_{t=1}^{t_\zeta} p_{\theta^c}(x_{t-1} \mid x_t) \tag{5}$$

Here, the first product operator describes the distribution approximated by the server model with weights θ^s, and the second product operator consequently defines the distribution approximated by the client model ϵ_θ^c.

3.1 Collaborative Training

During training, client models ϵ_θ^c and the server model ϵ_θ^s are updated independently. Alg. 1 provides our collaborative training procedure in pseudocode, which we now describe in more detail. For training, each client $c \in C$ has access to a private dataset $D_c = \{(x_0^i, y^i)\}$ of images x_0^i with optional textual attribute labels y^i. In principle, our approach also works with unlabeled data and other kinds of labels, e.g., one-hot encoded label vectors and segmentation maps. However, we focus on the use case of attribute-conditioned image generation in this work. By using textual feature descriptions as labels, our implementation can easily be extended to more elaborate text-guided image synthesis. As for the standard diffusion training process, each client samples a training batch $\{(x_0, y)\}^b \subseteq D_c$ of batch size b together with client time steps $t^c \sim \mathcal{U}[1, t_\zeta]^b$ during each training step. Gaussian noise is added to each training sample based on t^c following the diffusion process defined in eq. (1). Note that x_{t^c} and the cut-point image x_{t_ζ} are diffused from the same noise draw ϵ^c (Alg. 1, lines 9–10), so that the client trajectory and the sample handed to the server share a consistent noise realisation. All noisy images

Algorithm 1 Collaborative Training

Require: training dataset D; batch size b; number of time steps T, cut point t_ζ,
variance scheduler α, noise scheduler σ, clients C, server s; rounds R, local epochs E

1: client dataset $D_c \subseteq D$
2: **for** round $r = 1$ **to** R **do**
3: **for** $c \in C$ **do**
4: **for** local epoch $e = 1$ **to** E **do**
5: **for** Each batch $\{(x_0, y)\}^b \subseteq D_c$ **do**
6: **** CLIENT NODE ****
7: $t^c \sim \mathcal{U}[1, t_\zeta]^b$ and $t^s \sim \mathcal{U}[t_\zeta, T]^b$
8: $\epsilon^c \sim \mathcal{N}(0, \mathbf{I})$, $\epsilon^s \sim \mathcal{N}(0, \mathbf{I})$
9: $x_{t^c} \leftarrow \alpha(t^c) \cdot x_0 + \sigma(t^c) \cdot \epsilon^c$ } Diffusion Process
10: $x_{t_\zeta} \leftarrow \alpha(t_\zeta) \cdot x_0 + \sigma(t_\zeta) \cdot \epsilon^c$ } Diffusion Process
11: $x_{t^s} \leftarrow \alpha(t^s) \cdot x_{t_\zeta} + \sigma(t^s) \cdot \epsilon^s$ } Diffusion Process
12: $\mathcal{L}_{t^c} = ||\epsilon_{\theta^c}(x_{t^c}, t^c, y) - \epsilon^c||_2^2$ } Denoising Process
13: Update θ^c } Denoising Process
14: Pass x_{t^s} and ϵ^s to server s } Denoising Process
15: **** SERVER NODE **** } Denoising Process
16: $\mathcal{L}_{t^s} = ||\epsilon_{\theta^s}(x_{t^s}, t^s, y) - \epsilon^s||_2^2$ } Denoising Process
17: Update θ^s } Denoising Process
18: **end for**
19: **end for**
20: **end for**
21: **end for**

are fed into the client's model to predict the added noise and update the model's parameters according to the loss function defined in eq. (4). In addition, each client uses the diffused image x_{t_ζ} from the cut point and samples additional server time steps $t^s \sim \mathcal{U}[t_\zeta, T]^b$ to provide the noisy images x_{t^s} for the server. The final noise image and the noise added to x_{t_ζ} are then used to update the server model's weights analogously. We note that the process of adding additional noise for the server could, in principle, also be performed on the server side. Line 11 constructs the server's training input by treating the cut-point sample x_{t_ζ} as a starting point and re-diffusing it to level t^s with the standard marginal coefficients $\alpha(t^s), \sigma(t^s)$, rather than applying the exact conditional transition $q(x_{t^s} \mid x_{t_\zeta})$. This is a deliberate design choice rather than an approximation error: the server is trained on exactly the marginal it is later asked to reverse, so the construction is internally consistent. Because the server's reverse trajectory at inference is initialised from pure noise and run over $[t_\zeta, T]$ on the same schedule (Alg. 2), the training and sampling marginals coincide. We use a single shared noise schedule throughout; the only distinction between client and server is the timestep interval each is responsible for.

Regarding privacy, the client passes both x_{t^s} and ϵ^s, so the server can algebraically recover the cut-point representation x_{t_ζ}. However, under our threat model it does not observe the clean image x_0. We therefore interpret the privacy benefit as follows: the server is limited to the noisy cut-point representation x_{t_ζ}, and we evaluate the residual leakage from that representation empirically in section 4.2.

Algorithm 2 Collaborative Inference

Require: number of time steps T, label y, cut point t_ζ, client c, server s

Sample initial noise: $x_T \sim \mathcal{N}(0, \mathbf{I})$
Choose client start $M \in [t_\zeta, T]$, e.g. $M = \left\lfloor t_\zeta + \frac{t_\zeta}{T}(T - t_\zeta) \right\rfloor$
t^c_{list} = linearly spaced array generator$(1, M, t_\zeta)$
$t \leftarrow T$
while $t \geq 1$ **do**
  if $t > t_\zeta$ **then**
    Compute x_{t-1} using $\epsilon_{\theta^s}(x_t, t, y)$ and $\alpha(t), \sigma(t)$ {server: $T - t_\zeta$ steps}
  else
    Compute x_{t-1} using $\epsilon_{\theta^c}(x_t, t^c_{list}[\,t\,], y)$ and $\alpha(t^c_{list}[\,t\,]), \sigma(t^c_{list}[\,t\,])$ {client: t_ζ steps; t^c_{list} has t_ζ entries spanning $[1, M]$, 1-indexed, $t \in \{1, \ldots, t_\zeta\}$}
  end if
  $t \leftarrow t - 1$
end while

3.2 Collaborative Inference

After training, each client $c \in C$ can send a request to the server containing optional textual attribute labels y. The individual steps during the inference are specified in Alg. 2. The denoising loop runs t from T down to 1: the server performs the first $T - t_\zeta$ steps using ϵ^s_θ conditioned on the label y, after which the still noisy samples $\hat{x}^s_{t_\zeta}$ are handed to the client c, which performs the remaining t_ζ steps using its local model ϵ^c_θ. To account for the increased amount of noise in $\hat{x}^s_{t_\zeta}$ and hence allow for a higher noise reduction on the client node, the variance and noise scheduler are adapted on the client side. The client schedule need not start exactly at t_ζ: it begins at a chosen value $M \in [t_\zeta, T]$ and is linearly spaced over its t_ζ steps, so the client compensates for the elevated residual noise without changing the total number of timesteps. In our experiments, we set $M = \lfloor t_\zeta + \frac{t_\zeta}{T}(T - t_\zeta) \rfloor$, which extends the client interval slightly above t_ζ; the simpler choice $M = t_\zeta$ (no rescaling) is equally valid and recovers the unadjusted schedule. A sufficiently large cut point t_ζ ensures that possibly sensitive features are generated on the client side while the server performs the less privacy-critical initial denoising steps. If multiple clients request samples from the same label y, the server-side denoising process can be run once to generate an intermediate noise sample, and each client solely has to compute the remaining denoising steps.

4 Experiment

To assess our approach, we implement Alg. 1 and Alg. 2 and train the models on common benchmark datasets. We simulate a scenario in which $k = 5$ clients use a trusted server to train a DDPM or LDM collaboratively. Each client has access to an individual subset from the same domain. Thereby, we investigate the influence of collaborative training on the fidelity of generated images. Furthermore, we analyze the influence of the chosen cut point t_ζ on sample fidelity with respect to disclosed information.

4.1 Experimental Protocol

To ensure consistent conditions and avoid confounding factors, we maintain identical training and inference hyperparameters and seeds across different runs and settings, if not stated otherwise. We provide our source code for reproducibility using a GitHub repository[1].

[1] https://github.com/SimeonAllmendinger/collafuse

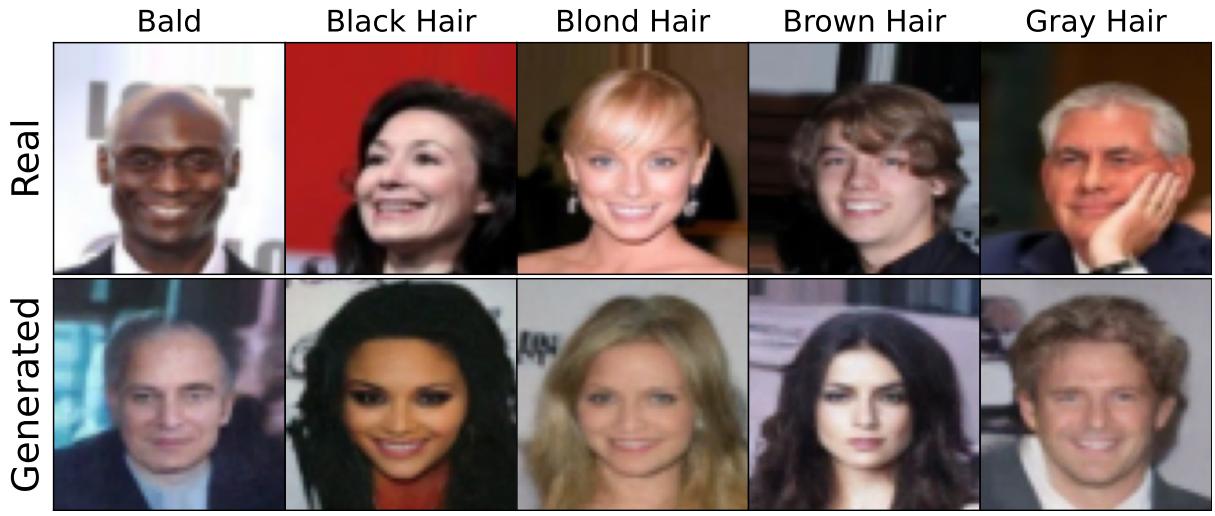

Fig. 2. Comparison between random samples from the training set (top row) and images generated with our collaborative diffusion models trained with cut point $t_\zeta = 100$ (bottom row). Images were not cherry-picked and generated starting with the same initial noise. The results demonstrate that collaboratively training diffusion models can achieve high image quality and attribute fidelity.

Model Architecture: We adopt the network architectures of traditional DDPMs (Ho et al. 2020), Imagen Models (Saharia, Chan, Saxena, et al. 2022), and LDMs (Rombach et al. 2022) which are based on the U-Net architecture (Ronneberger et al. 2015) to process 32×32, 64×64, 256×256, and 512×512 RGB images. As in the original implementation, the Imagen models are conditioned on text embeddings computed by T5-Base (Raffel et al. 2020). For the DDPMs, we use one-hot encoding. For the LDMs, we use pre-trained U-Nets as well as text- and autoencoders from "*CompVis/ldm-text2im-large-256*" (Rombach et al. 2022) and "*runwayml/stable-diffusion-v1-5*" (Rombach et al. 2022) from huggingface. Unlike Saharia, Chan, Saxena, et al. (2022), we do not apply any super-resolution model to increase the fidelity of the generated images in the Imagen model, as the focus of our work lies on the feasibility of the collaborative training and inference process. However, our collaborative diffusion setting also allows each client to add individual or shared super-resolution models (Saharia, Chan, Chang, et al. 2022) to their pipeline to upscale the generated images.

Datasets: We train and evaluate our collaborative diffusion models on common datasets: *CelebA*, *CIFAR-10*, *Animals-with-Attributes2 (Awa2)*. The CelebA dataset, collected by Z. Liu et al. (2015), consists of over 200,000 facial images of celebrities annotated with 40 binary attribute labels, including gender appearance, perceived age, hair color, and facial expressions. The CIFAR-10 dataset (Krizhevsky et al. 2009) includes 60,000 32x32 color images across 10 classes, such as airplanes, automobiles, and animals. The Awa2 dataset (Xian et al. 2019) contains 37,322 images of 50 animal classes with 85 attributes.

To evaluate `CollaFuse` under more realistic conditions, we extended our experimental setup beyond IID data partitions. While CIFAR-10 and Awa2 were uniformly split among clients, the CelebA dataset was distributed non-IID with respect to semantic attributes (Figure 3). Each client thus specialized in distinct attribute combinations (e.g., hair color, facial traits, or accessories), reflecting natural data heterogeneity often observed in federated or collaborative learning scenarios.

Training Parameters: Our training protocol employs a learning rate of 0.001, a batch size of 8, and $T = 1000$ timesteps. Each client holds 10,000 / 56,870-123,908 / 6,717 training images and 2,000 / 6,319-13,768 / 746 test

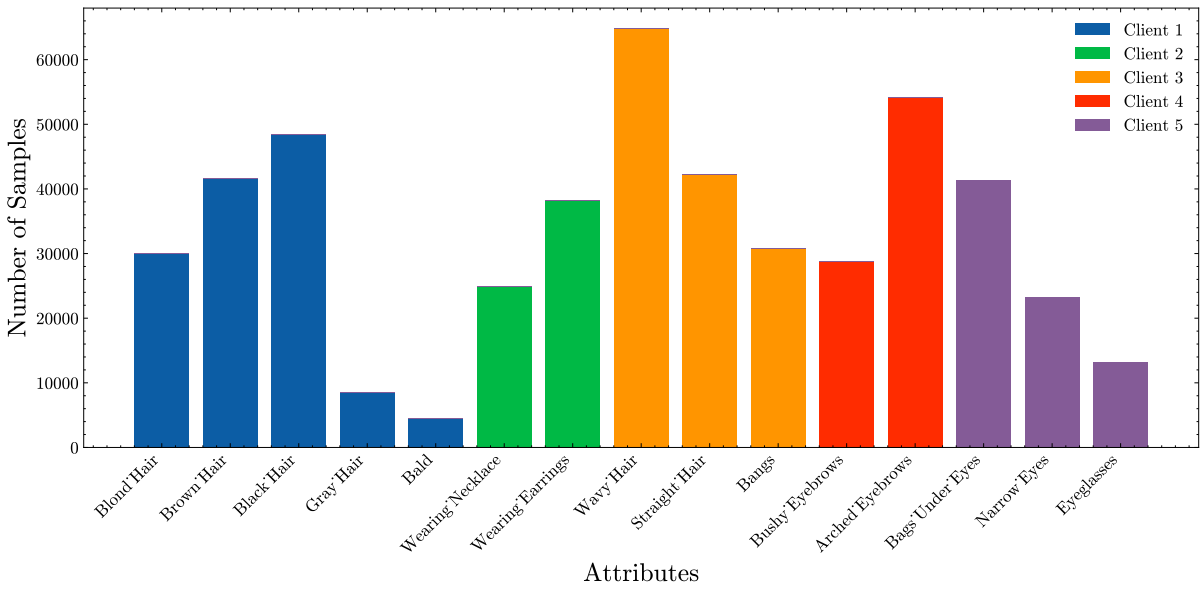

Fig. 3. While datasets such as CIFAR-10 and AWA2 were evenly partitioned among clients (IID), CelebA was deliberately distributed according to attribute labels to simulate heterogeneous, non-IID client data. Each color indicates one client's subset of dominant attributes, such as hair color, facial features, or accessories. This setup mimics realistic data heterogeneity across participants, where clients capture distinct subpopulations rather than uniform samples.

images (hold-out dataset) w.r.t the different datasets CIFAR-10 / CelebA / Awa2. For CelebA, the attribute-based non-IID split is overlapping rather than a disjoint partition: because each image carries multiple binary attributes, an image can be assigned to more than one client's attribute band, so the per-client counts sum to more than the dataset size. The experiments were conducted utilizing 11 NVIDIA A100-SXM4-80GB for computational processing.

Evaluation Metrics: To assess the quality of the generated images, we calculate the common *Fréchet Inception Distance* (FID) (Heusel et al. 2017) and the *Fréchet CLIP Distance* (FCD) between the test dataset and generated images from each client. We differentiate between images generated by clients from pure Gaussian noise ($t_\zeta = 1000$), and images generated based on the server image at the cut point. All metrics are computed on the implementations provided by Parmar et al. (2022) to ensure stable and comparable evaluation results. For both metrics, lower values indicate better approximation of the training distribution and improved image quality.

As we are interested in the performance of `CollaFuse`, we calculate the image fidelity across the set of clients to compare the performance for different values of t_ζ. Furthermore, we calculate the fidelity between the partially diffused images at the cut point t_ζ and the original images of the clients to assess the information disclosed to the server. We use this as a *proxy for the visual and distributional similarity* between the server-side intermediate outputs and real images, rather than as a privacy metric: a high FID/FCD indicates that server outputs are visually dissimilar from real data, but does not by itself bound membership, reconstruction, or attribute leakage. We complement it with the dedicated attribute-inference and inversion experiments in section 4.2, which assess leakage directly.

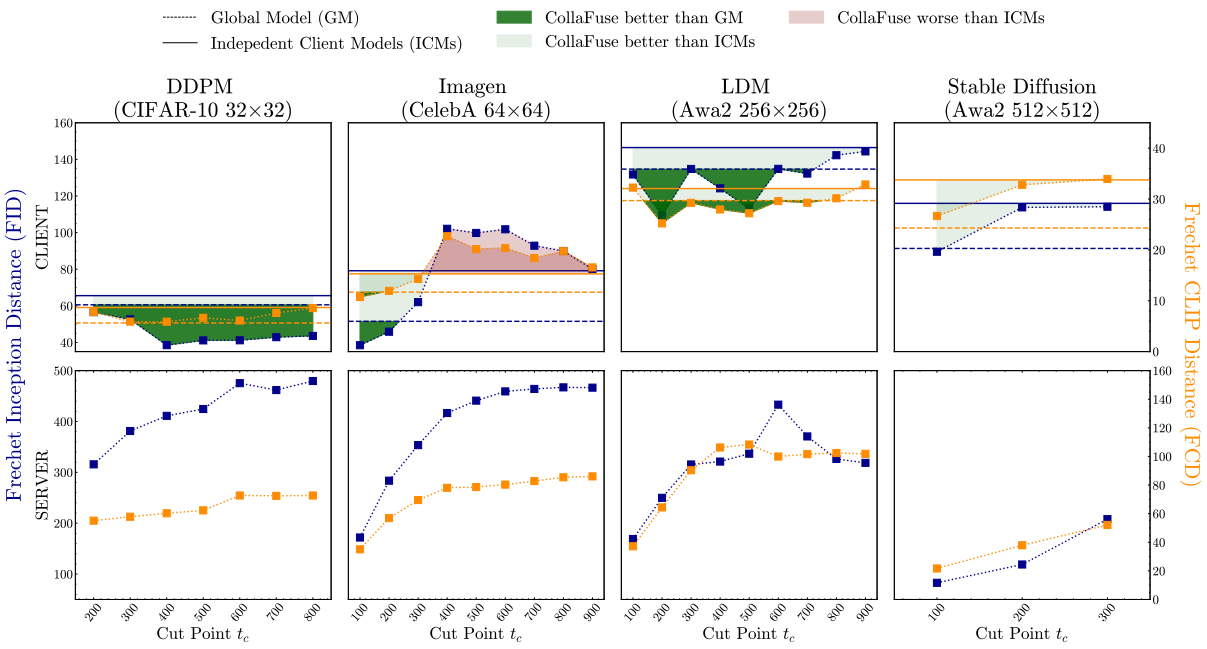

Fig. 4. Fidelity evaluation of clients and server using FID↓ and FCD↓: We assess 10,000 / 6,319-13,768 / 3,730 real x_0 and collaboratively generated images ($\hat{x}_0^c \circ \hat{x}_{t_\zeta}^s(\epsilon)$) from the CIFAR-10 / CelebA / Awa2 dataset across clients. Our findings show that cut points with $t_\zeta \leq 200$ outperform the baseline of independent client models (ICMs) ($t_\zeta = 1000$). In some of these settings—particularly at lower resolutions or smaller latent sizes—small cut points also surpass the global model (GM) ($t_\zeta = 0$), which is trained on the pooled data of all clients and is therefore not a privacy-feasible baseline in our setting. As expected, information disclosure almost consistently decreases as the cut point (t_ζ) increases across all models (DDPM, Imagen, LDM, Stable Diffusion).

4.2 Experimental Results

In our experiments, we analyze the performance of collaborative diffusion models by focusing on specific distributed categories among clients. Figure 2 displays exemplary generated images of our collaborative approach from the dataset CelebA, comparing them with their attributes and real images from the dataset. Our quantitative analysis includes a comparison with two baselines: global model ($t_\zeta = 0$) and independent client models ($t_\zeta = 1000$). The single global model (GM) on the server node is trained using the combined datasets of all clients, while the independent client models (ICM) are trained on client-specific distributed sub-datasets and separately operate on the client node. The FID and FCD scores in Figure 4 illustrate the feasibility of training diffusion models collaboratively. Each column shows the scores for a different architecture, dataset, and image resolution. The first row compares the performance for different cut-points with the GM (dashed horizontal line) and ICMs (solid horizontal line). The squares represent the FID and FCD scores for the respective cut points. Further, it is illustrated whether collaboration increases or decreases the performance. The second row shows the FID and FCD scores based on the images generated from the server, which are then sent to the clients. The figure shows that models with cut points $t_\zeta \leq 200$ surpass the performance of the independent client models in favor of collaborative image synthesis. In some settings, our collaborative experiments with smaller cut points also outperform the global model, which pools client data and is therefore not a privacy-feasible baseline. However,

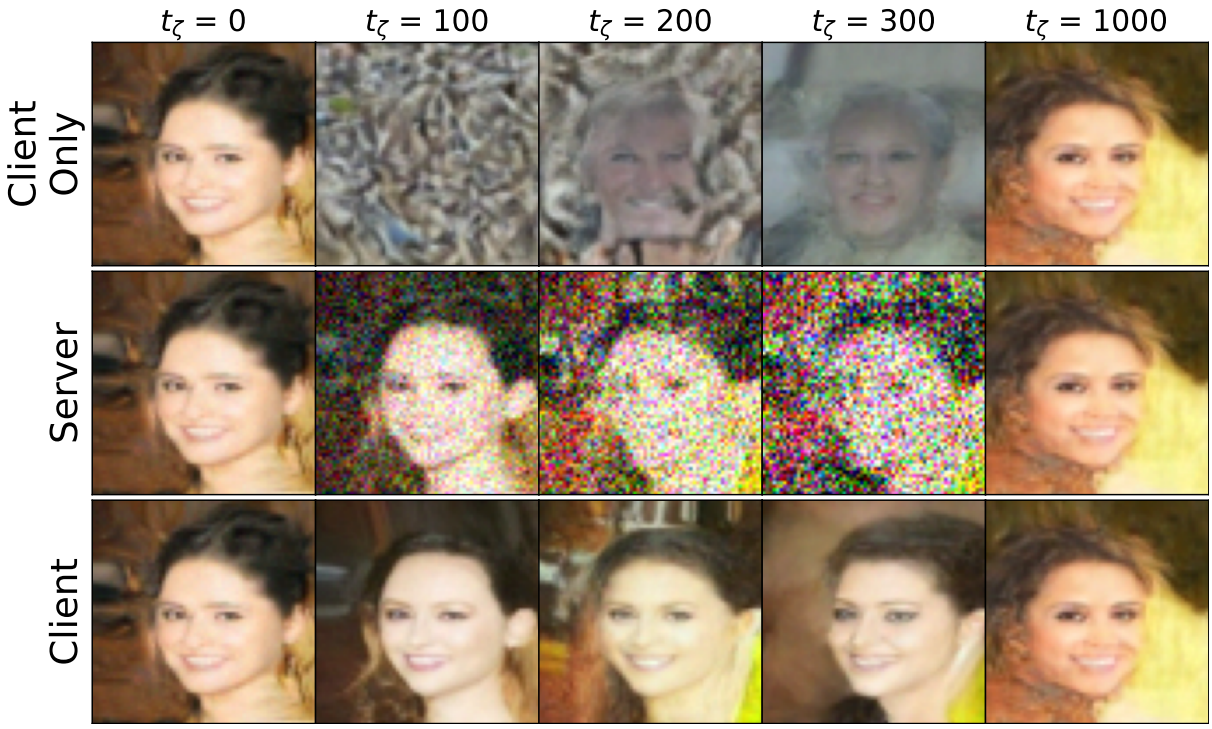

Fig. 5. Samples generated by CollaFuse trained on CelebA with different cut points t_ζ. The top row depicts images produced by the server, which are then sent to the client. The bottom row shows the samples after the final denoising performed by the client. For $t_\zeta = 0$, the server performs the full denoising process; for $t_\zeta = 1000$, each client trains a separate diffusion model without a server component.

cut points $t_\zeta > 300$ often weaken the fidelity of image generation, which converges with the performance of the independent client models for higher cut points.

In terms of **image quality**, we observe that the degradation of image fidelity and visual characteristics is evident in images generated by the server as the cut point t_ζ increases. Figure 5 displays this deterioration, which affects the images generated by the server at higher cut points. Conversely, the client's performance in generating images from noise is compromised when operating with low cut points. However, leveraging our collaborative approach, each client can produce meaningful images with improved fidelity, as shown in Figure 4, making use of denoised images provided by the server. Additionally, our results indicate that incorporating an adjustment of the variance and noise scheduler into the collaborative inference process significantly enhances the denoising capabilities on the client node. This is particularly effective given the higher levels of residual noise in the images received from the server, leading to an improvement in the overall quality of the images. Adopting the adjusted parameter M in Alg. 2, especially with an increased emphasis on managing variance and noise in the collaborative setting, proves to be beneficial in refining the quality of the final image output. To put these results into context, we perform ablation studies that compare CollaFuse to FedAvg-DDPM in terms of image quality, communication cost, and client-side computational cost during training and inference, showing that CollaFuse consistently improves KID across four datasets while reducing client-side FLOPs to t_ζ/T of the FedAvg-DDPM cost, with details provided in Appendix A.

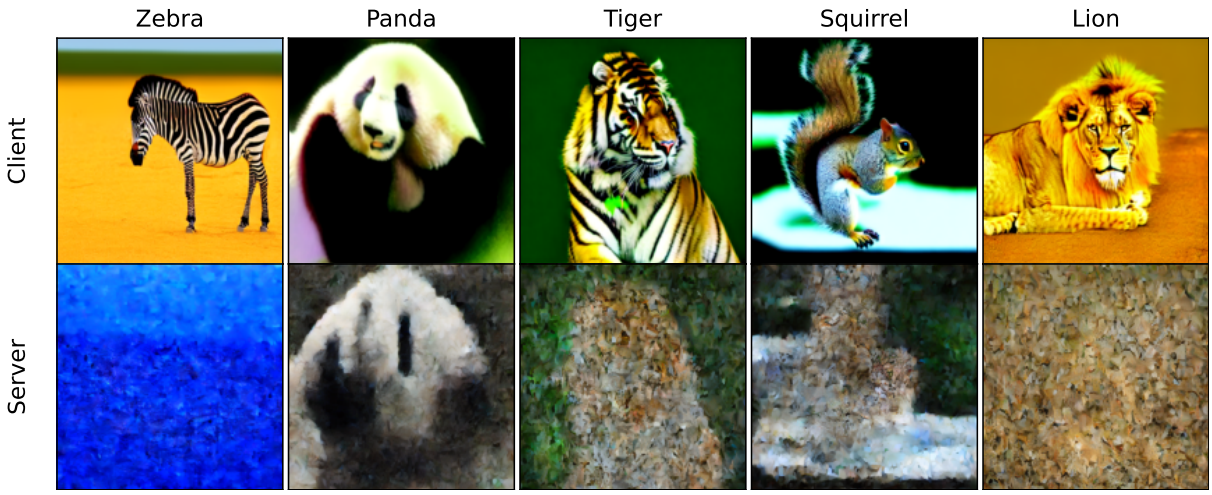

Fig. 6. Samples generated by CollaFuse trained on Awa2 for the cut point $t_\zeta = 400$. The top row depicts images after the final denoising performed by the client. The bottom row shows images produced by the server, which are then sent to the client.

In terms of **information disclosure** to the server, we analyze the similarity between real images and the images generated by the server–these are the images sent to clients for further denoising. We use the FID and FCD scores to assess how similar the generated images match the training distribution. The results show that as the cut point increases, the FID and FCD scores also rise, indicating a decrease in the similarity between the distribution of real images and that of the server-generated images. Figure 6 illustrates the server's ability to generate images and the extent of information disclosed for a cut point $t_\zeta = 400$.

We assume an honest-but-curious setting: each party follows the protocol but may inspect what it receives. The server sees x_{t^s} and ϵ^s per sample (Alg. 1, line 14), from which it can recover x_{t_ζ} but not x_0; a malicious client sees only the intermediate sample $\hat{x}^s_{t_\zeta}$ for its own label and its own data. Our attribute-inference adversary is a (non-pretrained) ViT-Base classifier trained on intermediate images; our inversion adversary is a simulated client reconstructing another client's data from cut-point features. We do not consider colluding, adaptive, or white-box adversaries, which we leave to future work. To further substantiate our analysis of information disclosure at intermediate cut points, we conducted a dedicated attribute inference experiment on CelebA. Using a ViT-Base classifier (*google/vit-base-patch16-224*; not pre-trained), we trained attribute predictors on intermediate images generated at different cut points t_ζ. Figure 7 summarizes the resulting F1-score differences for all 40 attributes relative to the baseline without a cut point ($t_\zeta = 0$). The results reveal a clear decline in attribute inference accuracy as the cut point increases, confirming that as more steps are assigned to the client the server-side representation is taken earlier in the reverse process, contains more noise, and leaks substantially less semantic and identity-related information. Under the specific classifiers and attacks tested, earlier (noisier) representations show reduced leakage. These are attack-based empirical observations under the stated threat model, not formal guarantees, and stronger adversaries may extract more.

Building on the attribute inference analysis, which focused on the leakage of semantic or sensitive attribute information, we next investigated whether intermediate representations could be exploited to reconstruct original data. This complementary evaluation assesses a more direct form of privacy risk, namely, the ability of an adversarial or malicious client to recover identifiable visual content from shared features.

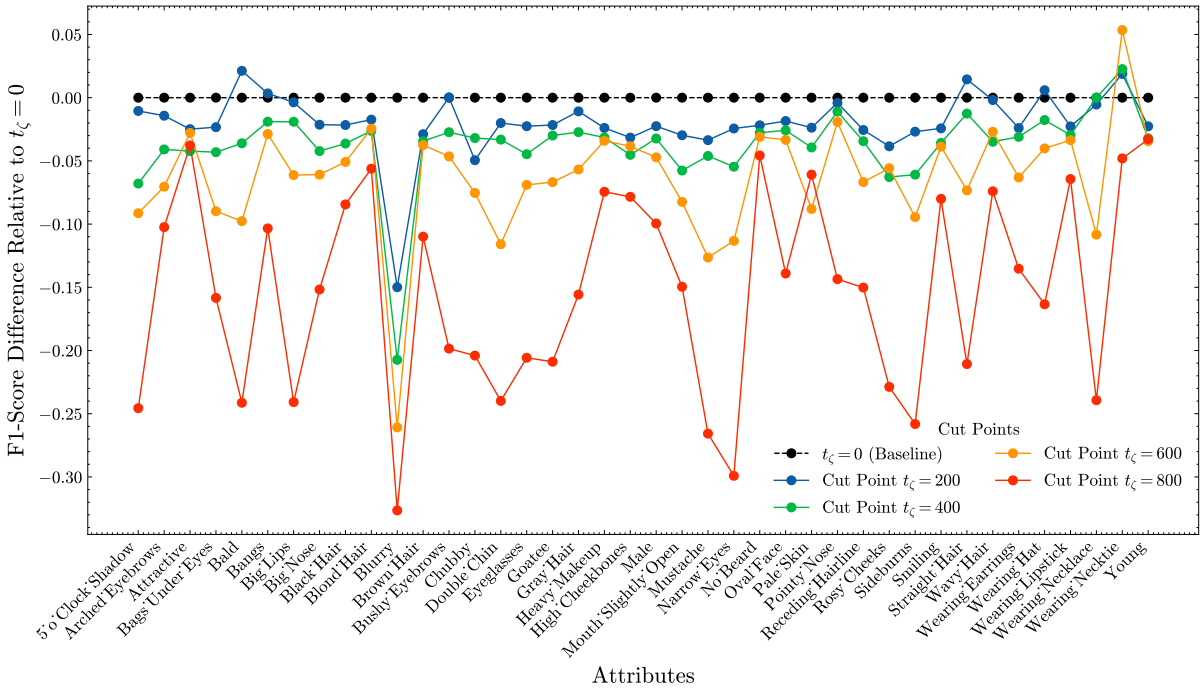

Fig. 7. Attribute inference F1-score differences for 40 CelebA attributes across varying cut points t_ζ. Each line represents the change in F1-score relative to the baseline without a cut point ($t_\zeta = 0$). Lower scores indicate reduced attribute predictability from intermediate representations. The results show a consistent decline in inference accuracy for earlier cut points, suggesting that higher noise levels at intermediate stages naturally mitigate information leakage. All baseline F1-scores at $t_\zeta = 0$ are above 65%.

To further assess reconstructibility and potential cross-client information leakage, we performed inversion attack experiments on the CIFAR-10 dataset. In this setting, a simulated malicious client attempted to reconstruct another client's private (training) data or infer its images from intermediate representations shared during collaborative generation. Using DDPMs conditioned on features extracted at various cut points t_ζ, we observed that reconstructed samples rapidly lost identifiable content as the cut point moved earlier in the generation process (Figure 8). For cut points $t_\zeta \geq 400$, the models exhibited a pronounced decline in their ability to reconstruct training data from other clients. Specifically, reconstruction quality remained higher for a model's own training data but deteriorated substantially for data originating from different clients, resulting in a sharp increase in the FCD metric. This indicates that, beyond this cut point, adversarial clients were increasingly constrained in reproducing the data distribution of other participants, reflecting a reduced potential for cross-client information leakage. Under the inversion attack evaluated here, these findings indicate that `CollaFuse` constrains cross-client reconstruction at higher cut points, consistent with the attribute-inference results. As with the attribute-inference analysis, this is empirical evidence against the specific attacker defined in our threat model rather than a formal guarantee.

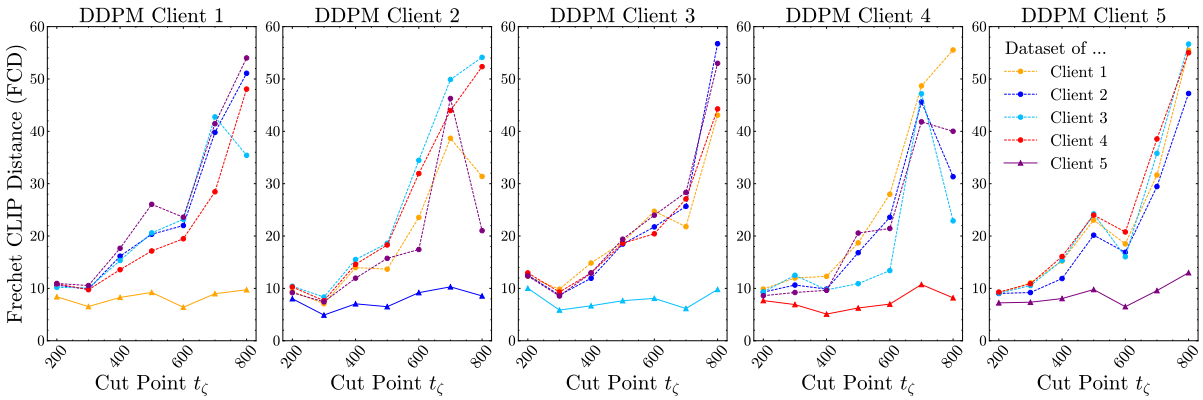

Fig. 8. Results of cross-client inversion attacks on CIFAR-10 across different cut points t_c. As the cut point increases (more denoising steps are performed on the client and the server hands over an earlier, noisier sample), reconstructed images exhibit a marked loss of client-specific semantics and visual detail. For $t_c \geq 400$, models show a decline in their ability to reconstruct other clients' training data, reflected in an increased FCD metric.

5 Discussion, Limitations, and Future Work

Our findings reveal that `CollaFuse` produces images of higher quality than those generated by independently trained diffusion models using sub-datasets. In several settings it also matches or surpasses globally trained diffusion models, which pool client data and are therefore not a privacy-feasible baseline. This highlights the efficacy of our method and demonstrates its capability to tackle the personalization versus generalization challenge frequently encountered in FL scenarios (Yao and B. Li 2024). Beyond image fidelity, our extended experiments provide a more comprehensive understanding of `CollaFuse` in terms of information disclosure. Under the attacks we evaluate, the attribute-inference analysis on CelebA shows that intermediate representations at higher cut points reveal substantially less semantic and identity-related information, consistent with higher noise levels in the diffusion process acting as a privacy buffer. Complementary inversion attack experiments on CIFAR-10 show the same trend: as the cut point increases, reconstructed samples rapidly lose visual fidelity and class-specific detail. Taken together, these results provide *attack-based empirical evidence* that, against the specific adversaries tested, the collaborative exchange in `CollaFuse` discloses limited information and constrains cross-client reconstruction. We emphasise that these are empirical observations under a defined threat model rather than formal privacy guarantees; stronger or adaptive adversaries may extract more, and we leave provable bounds to future work (see below). At the same time, our approach circumvents the need to share raw data or complete model updates, opting instead to share only the diffused images alongside server-side noise. Furthermore, by delegating computationally intensive tasks, our approach significantly reduces the computational load on individual clients, leveraging the potential strengths of growing foundation models on large servers (see Appendix A). Consequently, even where image fidelity decreases, clients may still favor a collaborative diffusion model for its ability to lighten their computational load, even in simple one-to-one setups. Thus, our strategy facilitates a balanced optimization of performance, information disclosure, and computational resources and can be tailored to individual preferences.

Limitations: Our current implementation of `CollaFuse` is based on DDPMs, which provide a theoretically grounded framework for analyzing collaborative training dynamics, cut-point sensitivity, and information disclosure. However, DDPMs are computationally intensive due to their stochastic denoising process. While this choice enables interpretability and a rigorous analysis of privacy aspects, it limits inference efficiency. Furthermore, our current approach assumes trustworthy clients and an honest server. However, collaborative

frameworks are known to be susceptible to backdoor attacks (Tajalli et al. 2023; Wang et al. 2020), in which an adversarial client may attempt to inject hidden functionalities into a shared model. Prior research has shown that diffusion models themselves are vulnerable to such attacks (Chou et al. 2023; Struppek, Hintersdorf, and Kersting 2023). While these scenarios are beyond the current scope, future work will extend our framework to systematically assess and mitigate adversarial threats while preserving image fidelity and efficiency.

Future Work: Future research will focus on extending `CollaFuse` to more advanced and efficient denoising paradigms such as DDIM (J. Song et al. 2021) and flow-based diffusion models. These variants can significantly accelerate inference and reduce resource requirements while maintaining the collaborative and privacy-preserving characteristics of our approach. In addition, we plan to scale `CollaFuse` to larger datasets, dynamic cut-point adaptation, and high-resolution domains to comprehensively assess its scalability and privacy–utility trade-offs under realistic conditions. Building on our privacy evaluations, future work will further explore formal privacy guarantees and attack-based analyses to establish stronger theoretical and empirical bounds on information leakage, while also analyzing the trade-offs among efficiency, privacy, and generation quality in more detail—also against potential benchmarks like FedAvg-DDPMs. Another promising direction is to investigate vulnerabilities such as backdoor attacks in collaborative diffusion and to develop countermeasures that preserve fidelity and efficiency. Additionally, integrating differentially private training algorithms (Dockhorn et al. 2023; Ghalebikesabi et al. 2023) and studying memorization effects in collaborative diffusion (Burg and Williams 2021; Carlini et al. 2023; Hintersdorf, Struppek, Kersting, et al. 2024) will further advance `CollaFuse` toward scalable, secure, and privacy-aware real-world applications. Finally, future research should place greater emphasis on algorithmic fairness to reduce the reinforcement of stereotypes and cultural biases when implementing CollaFuse (Struppek, Hintersdorf, Friedrich, et al. 2023), especially since privacy and fairness are both essential characteristics of real-world collaborative learning (Balbierer et al. 2024).

6 Conclusion

Our collaborative diffusion approach `CollaFuse` offers a novel solution to the challenges of diffusion-based generative models. By dividing the denoising process between a shared server and client models, we address performance, information disclosure, and computational concerns effectively. Our approach enables clients to outsource computationally intensive denoising steps to the server, balancing image quality without the necessity of sharing raw data. Through experiments, we demonstrate the effectiveness of collaborative training in enhancing image quality tailored to each client's domain while reducing the number of denoising steps on the client side. These findings highlight the potential of `CollaFuse` in advancing distributed machine learning research and development.

References

S. Augenstein, H. B. McMahan, D. Ramage, S. Ramaswamy, P. Kairouz, M. Chen, R. Mathews, and B. A. y Arcas. 2020. "Generative Models for Effective ML on Private, Decentralized Datasets." In: *International Conference on Learning Representations (ICLR)*.

B. Balbierer, L. Heinlein, D. Zipperling, and N. Kühl. 2024. "A Multivocal Literature Review on Privacy and Fairness in Federated Learning." In: *Wirtschaftsinformatik 2024 Proceedings*. https://aisel.aisnet.org/wi2024/14.

M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. 2018. "Demystifying MMD GANs." In: *International Conference on Learning Representations (ICLR)*.

T. Brooks et al.. 2024. *Video generation models as world simulators*. https://openai.com/research/video-generation-models-as-world-simulators. Accessed: 19-June-2024. (2024).

G. J. J. van den Burg and C. K. I. Williams. 2021. "On Memorization in Probabilistic Deep Generative Models." In: *Conference on Neural Information Processing Systems (NeurIPS)*, 27916–27928.

N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramèr, B. Balle, D. Ippolito, and E. Wallace. 2023. "Extracting Training Data from Diffusion Models." In: *USENIX Security Symposium (USENIX)*, 5253–5270.

S.-Y. Chou, P.-Y. Chen, and T.-Y. Ho. June 2023. "How to Backdoor Diffusion Models?" In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, (June 2023), 4015–4024. doi:10.1109/cvpr52729.2023.00391.

T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. 2018. "Deep learning for classical japanese literature." *arXiv preprint arXiv:1812.01718*.

G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik. 2017. "EMNIST: Extending MNIST to handwritten letters." In: *2017 international joint conference on neural networks (IJCNN)*. IEEE, 2921–2926.

P. Dhariwal and A. Q. Nichol. 2021. "Diffusion Models Beat GANs on Image Synthesis." In: *Advances in Neural Information Processing Systems (NeurIPS)*, 8780–8794.

T. Dockhorn, T. Cao, A. Vahdat, and K. Kreis. 2023. "Differentially Private Diffusion Models." *Transactions on Machine Learning Research (TMLR)*. https://openreview.net/forum?id=ZPpQk7FJXF.

J. Duan, J. Duan, X. Wan, and Y. Li. 2023. "Efficient Federated Learning Method for Cloud-Edge Network Communication." In: *2023 5th International Conference on Communications, Information System and Computer Engineering (CISCE)*, 118–121. doi:10.1109/CISCE58541.2023.10142819.

C. Fan and P. Liu. 2020. "Federated Generative Adversarial Learning." In: *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, 3–15.

X. Feng, C. Luo, J. Chen, Y. Huang, J. Zhang, W. Xu, J. Li, and V. C. M. Leung. 2023. "IoTSL: Toward Efficient Distributed Learning for Resource-Constrained Internet of Things." *IEEE Internet of Things Journal*, 10, 11, 9892–9905. doi:10.1109/JIOT.2023.3235765.

M. Fredrikson, S. Jha, and T. Ristenpart. 2015. "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures." In: *Conference on Computer and Communications Security (CCS)*, 1322–1333.

M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart. 2014. "Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing." In: *USENIX Security Symposium*, 17–32.

S. Ghalebikesabi et al.. 2023. "Differentially Private Diffusion Models Generate Useful Synthetic Images." *arXiv Preprint*. arXiv: 2302.13861.

M. de Goede, B. Cox, and J. Decouchant. 2024. "Training Diffusion Models with Federated Learning." *arXiv preprint*. arXiv: 2406.12575.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. 2020. "Generative adversarial networks." *Communications of the ACM*, 63, 11, 139–144.

O. Gupta and R. Raskar. 2018. "Distributed learning of deep neural network over multiple agents." *Journal of Network and Computer Applications*, 116, 1–8.

C. Hardy, E. Le Merrer, and B. Sericola. 2019. "MD-GAN: Multi-Discriminator Generative Adversarial Networks for Distributed Datasets." In: *International Parallel and Distributed Processing Symposium (IPDPS)*, 866–877.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. 2017. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium." In: *Advances in Neural Information Processing Systems (NeurIPS)*, 6626–6637.

D. Hintersdorf, L. Struppek, M. Brack, F. Friedrich, P. Schramowski, and K. Kersting. July 2024. "Does CLIP Know My Face?" *Journal of Artificial Intelligence Research*, 80, (July 2024), 1033–1062. doi:10.1613/jair.1.15461.

D. Hintersdorf, L. Struppek, K. Kersting, A. Dziedzic, and F. Boenisch. 2024. "Finding NeMo: localizing neurons responsible for memorization in diffusion models." In: *Proceedings of the 38th International Conference on Neural Information Processing Systems* (NIPS '24) Article 2800. Curran Associates Inc., Vancouver, BC, Canada, 43 pages. ISBN: 9798331314385.

J. Ho, A. N. Jain, and P. Abbeel. 2020. "Denoising Diffusion Probabilistic Models." In: *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*. https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. 2022. "LoRA: Low-Rank Adaptation of Large Language Models." In: *International Conference on Learning Representations (ICLR)*.

X. Huang, P. Li, H. Du, J. Kang, D. Niyato, D. I. Kim, and Y. Wu. 2024. "Federated Learning-Empowered AI-Generated Content in Wireless Networks." *IEEE Network*, 38, 5, 304–313. doi:10.1109/MNET.2024.3353377.

M. Joshi, A. Pal, and M. Sankarasubbu. 2022. "Federated Learning for Healthcare Domain - Pipeline, Applications and Challenges." *ACM Transactions on Computing for Healthcare*, 3, 4.

F. V. S. Jothiraj and A. Mashhadi. May 2024. "Phoenix: A Federated Generative Diffusion Model." In: *Companion Proceedings of the ACM Web Conference 2024* (WWW '24 Companion). ACM, New York, NY, USA, (May 2024), 1568–1577. doi:10.1145/3589335.3651935.

A. Kazerouni, E. K. Aghdam, M. Heidari, R. Azad, M. Fayyaz, I. Hacihaliloglu, and D. Merhof. 2023. "Diffusion models in medical imaging: A comprehensive survey." *Medical Image Analysis*, 88, 102846.

D. P. Kingma and M. Welling. 2014. "Auto-Encoding Variational Bayes." In: *International Conference on Learning Representations (ICLR)*.

P. Kortoçi, Y. Liang, P. Zhou, L.-H. Lee, A. Mehrabi, P. Hui, S. Tarkoma, and J. Crowcroft. 2022. "Federated split GANs." In: *ACM Workshop on Data Privacy and Federated Learning Technologies for Mobile Edge Network*.

A. Krizhevsky, V. Nair, and G. Hinton. 2009. "CIFAR-10 (Canadian Institute for Advanced Research)." http://www.cs.toronto.edu/~kriz/cifar.html.

B. Lai, J. He, J. Kang, G. Li, M. Xu, T. zhang, and S. Xie. June 2024. "On-demand Quantization for Green Federated Generative Diffusion in Mobile Edge Networks." In: *ICC 2024 - IEEE International Conference on Communications*. IEEE, (June 2024), 2883–2888. doi:10.1109/icc51166.2024.10622695.

Y. LeCun. 1998. "The MNIST database of handwritten digits." *http://yann.lecun.com/exdb/mnist/*.

W. Li, J. Chen, Z. Wang, Z. Shen, C. Ma, and X. Cui. 2022. "IFL-GAN: Improved Federated Learning Generative Adversarial Network With Maximum Mean Discrepancy Model Aggregation." *IEEE Transactions on Neural Networks and Learning Systems*, 1–14.

C. Little, M. Elliot, and R. Allmendinger. 2023. "Federated learning for generating synthetic data: a scoping review." *International Journal of Population Data Science*, 8, 1.

Z. Liu, P. Luo, X. Wang, and X. Tang. 2015. "Deep Learning Face Attributes in the Wild." In: *International Conference on Computer Vision (ICCV)*.

B. Ma, X. Yin, J. Tan, Y. Chen, H. Huang, H. Wang, W. Xue, and X. Ban. Mar. 2024. "FedST: Federated Style Transfer Learning for Non-IID Image Segmentation." *Proceedings of the AAAI Conference on Artificial Intelligence*, 38, 5, (Mar. 2024), 4053–4061. doi:10.1609/aaai.v38i5.28199.

G. Mariani, I. Tallini, E. Postolache, M. Mancusi, L. Cosmo, and E. Rodolà. 2024. "Multi-Source Diffusion Models for Simultaneous Music Generation and Separation." In: *International Conference on Learning Representations (ICLR)*.

B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas. 2017. "Communication-Efficient Learning of Deep Networks from Decentralized Data." In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 1273–1282.

M. Mendieta, G. Sun, and C. Chen. 2025. "Navigating Heterogeneity and Privacy in One-Shot Federated Learning with Diffusion Models." In: *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2601–2610. doi:10.1109/WACV61041.2025.00258.

S. Nazir and M. Kaleem. 2023. "Federated Learning for Medical Image Analysis with Deep Neural Networks." *Diagnostics*, 13, 9.

A. Q. Nichol and P. Dhariwal. 2021. "Improved Denoising Diffusion Probabilistic Models." In: *International Conference on Machine Learning (ICML)*, 8162–8171.

S. Ohta and T. Nishio. 2023. "Λ-Split: A Privacy-Preserving Split Computing Framework for Cloud-Powered Generative AI." *arXiv preprint*. arXiv: 2310.14651.

G. Parmar, R. Zhang, and J.-Y. Zhu. 2022. "On Aliased Resizing and Surprising Subtleties in GAN Evaluation." In: *Conference on Computer Vision and Pattern Recognition (CVPR)*, 11400–11410.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. 2020. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer." *Journal of Machine Learning Research*, 21, 140, 1–67.

R. Ramachandranpillai, M. F. Sikder, D. Bergström, and F. Heintz. Apr. 2024. "Bt-GAN: Generating Fair Synthetic Healthdata via Bias-transforming Generative Adversarial Networks." *Journal of Artificial Intelligence Research*, 79, (Apr. 2024), 1313–1341. doi:10.1613/jair.1.15317.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. 2022. "High-Resolution Image Synthesis With Latent Diffusion Models." In: *Conference on Computer Vision and Pattern Recognition (CVPR)*, 10684–10695.

O. Ronneberger, P. Fischer, and T. Brox. 2015. "U-Net: Convolutional Networks for Biomedical Image Segmentation." In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Vol. 9351, 234–241.

C. Saharia, W. Chan, H. Chang, C. A. Lee, J. Ho, T. Salimans, D. J. Fleet, and M. Norouzi. 2022. "Palette: Image-to-Image Diffusion Models." In: *Special Interest Group on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 15:1–15:10.

C. Saharia, W. Chan, S. Saxena, et al.. 2022. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." In: *Conference on Neural Information Processing Systems (NeurIPS)*.

R. Schwermer, E.-A. Bicer, P. Schirmer, R. Mayer, and H.-A. Jacobsen. 2023. "Federated Computing in Electric Vehicles to Predict Coolant Temperature." In: *International Middleware Conference: Industrial Track*, 8–14.

R. Schwermer, J. Buchberger, R. Mayer, and H.-A. Jacobsen. 2022. "Federated office plug-load identification for building management systems." In: *ACM International Conference on Future Energy Systems*, 114–126.

Y. Shen, A. Sowmya, Y. Luo, X. Liang, D. Shen, and J. Ke. 2023. "A Federated Learning System for Histopathology Image Analysis With an Orchestral Stain-Normalization GAN." *IEEE Transactions on Medical Imaging*, 42, 7, 1969–1981.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov. 2017. "Membership Inference Attacks Against Machine Learning Models." In: *Symposium on Security and Privacy (S&P)*, 3–18.

U. Singer et al.. 2023. "Make-A-Video: Text-to-Video Generation without Text-Video Data." In: *International Conference on Learning Representations (ICLR)*.

J. Song, C. Meng, and S. Ermon. 2021. "Denoising Diffusion Implicit Models." In: *International Conference on Learning Representations*. https://openreview.net/forum?id=St1giarCHLP.

Y. Song, H. Liu, S. Zhao, H. Jin, J. Yu, Y. Liu, R. Zhai, and L. Wang. Oct. 2024. "Fedadkd: heterogeneous federated learning via adaptive knowledge distillation." *Pattern Analysis and Applications*, 27, 4, (Oct. 2024). doi:10.1007/s10044-024-01350-4.

Y. Song and S. Ermon. 2020. "Improved Techniques for Training Score-Based Generative Models." In: *Advances in Neural Information Processing Systems (NeurIPS)*.

L. Struppek, D. Hintersdorf, F. Friedrich, M. Brack, P. Schramowski, and K. Kersting. Dec. 2023. "Exploiting Cultural Biases via Homoglyphs in Text-to-Image Synthesis." *Journal of Artificial Intelligence Research*, 78, (Dec. 2023), 1017–1068. doi:10.1613/jair.1.15388.

L. Struppek, D. Hintersdorf, and K. Kersting. 2023. "Rickrolling the Artist: Injecting Backdoors into Text Encoders for Text-to-Image Synthesis." In: *International Conference on Computer Vision (ICCV)*.

S. Sun, G. Goldgof, A. Butte, and A. Alaa. 2023. "Aligning Synthetic Medical Images with Clinical Knowledge using Human Feedback." In: *Thirty-seventh Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=qlnlamFQEa.

B. Tajalli, O. Ersoy, and S. Picek. 2023. "On Feasibility of Server-side Backdoor Attacks on Split Learning." In: *IEEE Security and Privacy Workshops (SPW)*, 84–93.

C. Thapa, P. C. Mahawaga Arachchige, S. Camtepe, and L. Sun. 2022. "SplitFed: When Federated Learning Meets Split Learning." *AAAI Conference on Artificial Intelligence (AAAI)*, 36, 8, 8485–8493.

N. R. Veeraragavan and J. F. Nygård. 2023. "Securing Federated GANs: Enabling Synthetic Data Generation for Health Registry Consortiums." In: *International Conference on Availability, Reliability and Security (ARES)*, 89:1–89:9.

H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J. Sohn, K. Lee, and D. S. Papailiopoulos. 2020. "Attack of the Tails: Yes, You Really Can Backdoor Federated Learning." In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. 2019. "Zero-Shot Learning—A Comprehensive Evaluation of the Good, the Bad and the Ugly." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41, 9, 2251–2265.

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

D. Yao and B. Li. Mar. 2024. "PerFedRLNAS: One-for-All Personalized Federated Neural Architecture Search." *Proceedings of the AAAI Conference on Artificial Intelligence*, 38, 15, (Mar. 2024), 16398–16406. doi:10.1609/aaai.v38i15.29576.

B. Yin, Z. Chen, and M. Tao. 2023. "Predictive GAN-Powered Multi-Objective Optimization for Hybrid Federated Split Learning." *IEEE Transactions on Communications*, 71, 8, 4544–4560.

M. Zajac, K. Deja, A. Kuzina, J. M. Tomczak, T. Trzcinski, F. Shkurti, and P. Milos. June 2023. "Exploring Continual Learning of Diffusion Models." In: *Proceedings of the 4th Workshop on Continual Learning in Computer Vision (CLVision), CVPR Workshops*. CVPR 2023 Workshop paper. (June 2023).

Z. Zhao, F. Yang, and G. Liang. Dec. 2023. "Federated Learning Based on Diffusion Model to Cope with Non-IID Data." In: *Pattern Recognition and Computer Vision*. Springer Nature Singapore, (Dec. 2023), 220–231. ISBN: 9789819985463. doi:10.1007/978-981-99-8546-3_18.

L. Zhu, Z. Liu, and S. Han. 2019. "Deep Leakage from Gradients." In: *Advances in Neural Information Processing Systems (NeurIPS)*, 14747–14756.

A Computational and Communication Cost

We compare the per-client cost of CollaFuse against a federated learning (FL) baseline for diffusion models. A diffusion model is a noise predictor $\epsilon_\theta : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ trained with the denoising objective

$$\mathcal{L}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[1,T],\, x_0\sim p_{\text{data}},\, \epsilon\sim\mathcal{N}(0,I)}\left[\, \|\epsilon_\theta(x_t,t) - \epsilon\|^2 \,\right], \qquad x_t = \alpha_t x_0 + \sigma_t \epsilon, \tag{6}$$

and sampled by integrating the reverse-time SDE.

A.1 Notation and Assumptions

Let Φ_{fwd} denote the FLOPs of one U-Net forward pass and $\Phi_{\text{bwd}} \approx 2\Phi_{\text{fwd}}$ those of a backward pass, so one training step costs $3\Phi_{\text{fwd}}$. Let T be the number of denoising timesteps, $t_\zeta \in [0,T]$ the cut point (server: $[t_\zeta, T]$; client: $[1, t_\zeta]$), k the number of clients, N the samples per client, E the local epochs per round, R the number of rounds, $|\theta|$ the parameter count (identical across all models), and d the image (or latent) dimensionality. The two ranges share the endpoint t_ζ, matching Alg. 1 (line 7), where the client samples $t^c \sim \mathcal{U}[1, t_\zeta]$ and the server $t^s \sim \mathcal{U}[t_\zeta, T]$; this single overlapping timestep is immaterial to the cost ratios below.

We adopt the following assumptions. **(α_1)** *Shared architecture*: Φ_{fwd} is identical across all models. **(α_2)** *Uniform timestep sampling*: timesteps are drawn uniformly from the relevant interval. **(α_3)** *Comparable gradient coverage*: the expected number of training steps per timestep needed to reach a target loss is comparable across schemes.

A.2 Training Cost

Each FL client trains the full network over all timesteps $[1,T]$. With a per-client step budget of *REN* minibatch updates, this gives

$$\Phi^{\text{FL}}_{\text{train}} = 3REN\,\Phi_{\text{fwd}}. \tag{7}$$

A CollaFuse client updates only θ^c and samples its training timesteps from $[1, t_\zeta]$ rather than $[1,T]$. We stress that this restriction alone does *not* reduce training cost: the training loop (Alg. 1) still performs *REN* minibatch updates, each a full client-network forward and backward pass, so with an unchanged step budget $\Phi^{\text{CF}}_{\text{train}} = 3REN\,\Phi_{\text{fwd}} = \Phi^{\text{FL}}_{\text{train}}$. A saving arises only if the client step budget is scaled in proportion to the share of the trajectory it must learn. Under (α_3), reaching a target loss requires a comparable number of updates *per timestep* across schemes; since the client now covers t_ζ of the T timesteps, a step budget of (t_ζ/T) *REN* suffices, giving

$$\Phi^{\text{CF}}_{\text{train}} = 3\left(\tfrac{t_\zeta}{T} REN\right)\Phi_{\text{fwd}} = 3REN\,\Phi_{\text{fwd}} \cdot \frac{t_\zeta}{T}. \tag{8}$$

Proposition 1 (Training cost, conditional). *Suppose the client's per-client step budget is scaled to (t_ζ/T) REN (e.g. by reducing rounds, local epochs, or samples accordingly). Then under (α_1)–(α_3), $\Phi^{CF}_{train}/\Phi^{FL}_{train} = t_\zeta/T$. Without such scaling, the timestep restriction leaves the number of optimizer steps and the per-step FLOPs unchanged, and the ratio is* 1.

This is a design statement about budget allocation, not an identity that follows from the timestep partition itself, and it is therefore weaker than the inference reduction below, which holds exactly from step counts alone (proposition 2). For $t_\zeta = 100$ and $T = 1000$, allocating the client 10% of the per-timestep training budget reduces client-side training FLOPs by a factor of 10; the remaining trajectory is learned by the shared server and amortized across clients.

A.3 Inference Cost

Generating one sample requires T sequential network evaluations under FL, $\Phi_{\text{inf}}^{\text{FL}} = T\,\Phi_{\text{fwd}}$. In CollaFuse the client evaluates only the final t_ζ steps while the server handles the rest:

$$\Phi_{\text{inf}}^{\text{CF, client}} = t_\zeta\,\Phi_{\text{fwd}}, \qquad \Phi_{\text{inf}}^{\text{CF, server}} = (T - t_\zeta)\,\Phi_{\text{fwd}}. \tag{9}$$

Proposition 2 (Inference cost). *Under (α_1), $\Phi_{inf}^{CF,\,client}/\Phi_{inf}^{FL} = t_\zeta/T$ exactly. If m clients share a conditioning label y, the server trajectory depends only on (x_T, y) and can be reused, reducing the amortised server cost to $(T-t_\zeta)\Phi_{fwd}/m$ while the client cost is unchanged.*

Proof. Immediate from the step counts in (9): the client runs t_ζ steps versus T under FL. Reuse of the server trajectory across clients sharing a label y requires fixing both the initial noise x_T and, for stochastic DDPM sampling, the server-side random draws. If clients require *diverse* samples, the server path cannot be reused without sacrificing sample diversity, and the amortized saving does not apply. □

We stress the distinction between the two reductions: the inference ratio (proposition 2) is an exact identity that follows purely from the step counts, whereas the training ratio (proposition 1) is contingent on (α_3).

A.4 Training Communication Cost

We define a *round* as the outer synchronization loop shared by both schemes: within each round, every client performs E local training epochs over its N samples, followed by a single synchronization step with the server.

Under FedAvg, each client communicates exactly once per round, uploading its local model update and downloading the aggregated model at a cost of $2|\theta|$ scalars. CollaFuse has a fundamentally different communication structure. Because the timestep partition assigns disjoint denoising ranges to client and server, each party computes its loss independently; the client minimizes its denoising objective over $[1, t_\zeta]$ without any server involvement. The server's training data is instead supplied by the client: at each training step the client constructs a noisy image x_{t^s} at a randomly sampled server timestep $t^s \sim \mathcal{U}[t_\zeta, T]$ and passes both x_{t^s} and the corresponding noise ϵ^s to the server (Alg. 1, lines 7–14), providing $2d$ scalars of training data per step—d for the noisy image and d for the noise target. No gradient signal flows from server to client, so local epochs $E > 1$ are feasible, in contrast to layer-wise split learning, where the server must participate in every client forward pass and $E = 1$ is effectively enforced. Over one round, the client sends EN such pairs, giving

$$C_{\text{round}}^{\text{CF}} = 2dEN. \tag{10}$$

Proposition 3 (Communication regime). CollaFuse *communicates less per round than FedAvg iff $|\theta| > ENd$.*

This is the *large-model, modest-data* regime. Modern image-diffusion U-Nets are large (e.g., the Stable Diffusion v1.5 U-Net has $\sim 10^9$ parameters) so $|\theta|$ typically lies in the 10^9 range. The communication balance, therefore, hinges on the local dataset size. For a modest $N = 10^4$ with $d = 3 \cdot 32 \cdot 32$ and $E = 1$, $ENd \approx 3 \times 10^7$ sits well below $|\theta|$ and CollaFuse communicates less per round. As N grows, the margin narrows: for our CelebA clients ($N \approx 5.7$–12.4×10^4, $d = 3 \cdot 64 \cdot 64$) the term ENd reaches $\sim 10^9$, comparable to a v1.5-scale U-Net, placing the setting near the regime boundary. Beyond it, very large local datasets or smaller models, the inequality reverses: the compute advantages (Props. 1–2) persist, but communication becomes the dominant per-round cost. The didactic experiments in appendix A.6 deliberately operate in this reversed regime ($|\theta| \approx 2 \times 10^5$) to stress-test the analysis at the opposite end of the scale spectrum. Inference additionally transmits $\hat{x}_{t_\zeta}^s$ (d scalars per sample) server-to-client.

This per-round scalar count is a simplified regime analysis. It omits timestep indices t_s, label/text embeddings y, quantization and precision, protocol overhead, the server-to-client inference transfer, and the *total* communication

required to reach a target quality rather than per-round volume. A full end-to-end communication-to-quality comparison is left to future measurement.

A.5 Client Comparison

Table A.1. Per-client cost (FLOPs; scalars for communication). The cut point t_ζ controls the client burden: small t_ζ offloads to the server, $t_\zeta = T$ recovers the local-only baseline. The training ratio holds under (α_3); the inference and communication entries are exact.

	Training	**Inference/sample**	**Comm./round**
FL	$3REN\,\Phi_{\text{fwd}}$	$T\,\Phi_{\text{fwd}}$	$2\lvert\theta\rvert$
`CollaFuse`	$\frac{t_\zeta}{T}\,3REN\,\Phi_{\text{fwd}}$	$t_\zeta\,\Phi_{\text{fwd}}$	$2dEN$
Ratio (CF/FL)	t_ζ/T	t_ζ/T	$dEN/\lvert\theta\rvert$

A.6 Quality Comparison and Analytic Cost Instantiation

We instantiate the analytic cost ratios of Props. 1–3 and provide a small-scale sample-quality comparison against a FedAvg-DDPM baseline across four 28×28 grayscale datasets: MNIST (LeCun 1998), FashionMNIST (Xiao et al. 2017), KMNIST (Clanuwat et al. 2018), and EMNIST-Letters (Cohen et al. 2017). The latter three increase visual and semantic diversity (clothing items, hiragana characters, and 26 letter classes, respectively), with EMNIST-Letters yielding the richest heterogeneity. To mirror the non-IID conditions of the main experiments, each dataset is sorted by label and partitioned into $k = 5$ contiguous bands, so every client specializes in a distinct subset of classes (similar to the main experiments). The FLOP ratios are computed from the formulas (not independently measured); the measured component is sample quality (KID).

Both methods use an identical didactic ≈200k-parameter U-Net (α_1), with $T = 500$ timesteps, cut point $t_\zeta = 100$, $R = 100$ rounds, and $E = 5$ local epochs per round. Unlike the deployment-scale U-Nets discussed in appendix A.4 ($|\theta| \sim 10^9$), this model is intentionally small ($|\theta| \approx 2 \times 10^5$) to keep the ablation tractable; as a consequence, it sits in the communication regime where proposition 3 predicts `CollaFuse` to be *disadvantaged*, which we confirm below. We vary the per-client dataset size $N \in \{256, 1024, 2048\}$ to assess scalability. FedAvg trains the full network over $[1, T]$ and averages weights each round; `CollaFuse` trains client models over $[1, t_\zeta]$ and a shared server over $(t_\zeta, T]$ with no weight exchange. Sample quality is measured by the Kernel Inception Distance (KID ↓) (Bińkowski et al. 2018), which quantifies the discrepancy between the distributions of generated and held-out real images and is well suited to unpaired generative evaluation. We note this comparison is not a controlled isolation of the timestep-split mechanism alone: `CollaFuse` differs from FedAvg in personalization, the presence of a shared server model, and the absence of weight exchange, any of which may contribute to the KID difference. We therefore present it as a preliminary ablation rather than a definitive comparison.

Cost ratios. The per-client FLOP ratios equal the theoretical $t_\zeta/T = 0.20$ exactly for inference (proposition 2); for training (proposition 1) the same 0.20 holds under the gradient-coverage assumption (α_3). Both ratios are independent of dataset and N. For the communication regime (proposition 3), the inequality reverses in this deliberately small-model setting: $|\theta| \approx 2 \times 10^5$ while $2dEN = 2 \cdot 784 \cdot 5 \cdot N \in \{2.0, 8.0, 16.1\} \times 10^6$ for $N \in \{256, 1024, 2048\}$—all substantially larger than $2|\theta| \approx 4.1 \times 10^5$. Consequently, FedAvg communicates 5–39× fewer scalars per round than `CollaFuse` here, confirming the predicted regime boundary at the small-model end of the spectrum complementary to the large-model regime of appendix A.4.

Table A.2. Empirical comparison across four non-IID datasets and three per-client data sizes ($T = 500$, $t_\zeta = 100$, $k = 5$, $R = 100$, $E = 5$). KID (×100) values are mean ± standard deviation across $k = 5$ clients; bold denotes the better mean per row. Client-side training and inference FLOP ratios equal $t_\zeta/T = 0.20$ (training under (α_3), inference exactly), dataset- and N-independent, and are omitted. Per-round communication: FedAvg $2|\theta| \approx 4.1 \times 10^5$ scalars (constant in N); CollaFuse $2dEN \in \{2.0, 8.0, 16.1\} \times 10^6$ for $N \in \{256, 1024, 2048\}$. KID standard deviations exceeding the mean (e.g. EMNIST-Letters, $N = 1024$) reflect finite-sample estimator variance; the unbiased KID estimator can produce slightly negative per-client values when generated images are very close to the real data.

		KID (↓)	
Dataset	N	FedAvg	CollaFuse
MNIST	256	3.51 ± 1.34	**1.81 ± 1.58**
	1024	2.48 ± 0.86	**0.95 ± 0.75**
	2048	1.12 ± 0.86	**0.54 ± 0.66**
FashionMNIST	256	6.86 ± 2.22	**4.58 ± 1.54**
	1024	4.96 ± 2.14	**2.45 ± 1.09**
	2048	4.24 ± 1.96	**2.73 ± 1.35**
KMNIST	256	8.51 ± 2.35	**3.72 ± 0.59**
	1024	4.93 ± 4.00	**2.01 ± 1.65**
	2048	2.13 ± 1.12	**1.09 ± 0.84**
EMNIST-Letters	256	4.43 ± 0.92	**0.31 ± 0.28**
	1024	1.31 ± 0.91	**0.18 ± 0.41**
	2048	0.87 ± 0.56	**0.17 ± 0.47**

Sample quality. Table A.2 shows that CollaFuse attains *lower KID* than FedAvg in every dataset/size cell, indicating that—despite operating on only 20% of the denoising trajectory locally—CollaFuse clients generate images whose distribution better matches the true data distribution. The KID gap is largest and most clearly outside the estimator noise on EMNIST-Letters at $N = 256$ (4.43 ± 0.92 vs. 0.31 ± 0.28), suggesting that the richer label diversity benefits most from the shared server representation; at larger N the EMNIST gap remains in the same direction but its KID standard deviation grows relative to the mean, so we read it as consistent rather than individually decisive. Taken together, these KID results indicate that CollaFuse preserves—and on the distributional KID metric improves—per-client generation quality relative to the FL baseline, consistent with our findings in the main experiments.

Received 08 November 2025; accepted 18 June 2026