# Training Diffusion Models with Federated Learning

## A communication-efficient model for cross-silo federated image generation

**Matthijs de Goede**

**Supervisors: Bart Cox, Jérémie Decouchant**

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Matthijs de Goede
Final project course: CSE3000 Research Project
Thesis committee: Bart Cox, Jérémie Decouchant and Qing Wang

## Abstract

The training of diffusion-based models for image generation is predominantly controlled by a select few Big Tech companies, raising concerns about privacy, copyright, and data authority due to the lack of transparency regarding training data. Hence, we propose a federated diffusion model scheme that enables the independent and collaborative training of diffusion models without exposing local data. Our approach adapts the Federated Averaging (FedAvg) algorithm to train a Denoising Diffusion Model (DDPM). Through a novel utilization of the underlying UNet backbone, we achieve a significant reduction of up to 74% in the number of parameters exchanged during training, compared to the naive FedAvg approach, whilst simultaneously maintaining image quality comparable to the centralized setting, as evaluated by the FID score. Our implementation is publicly available.[1]


## 1 Introduction

Recently, there has been a surge in the popularity of diffusion-based image generation models like Stable Diffusion [1], Imagen [2], and DALL-E [3], which have been praised for their ability to generate synthetic images of exceptional quality and realism. Effective training of these generative models, which typically have hundreds of millions of parameters, requires significant computing power, storage capacities, and a vast amount of training data [4]. As a result, most state-of-the-art models are produced by only a handful of Big Tech corporations that have the means to train and maintain them, leading to further consolidation of power within Big Tech [4].

Furthermore, the lack of transparency surrounding the origin of their training data raises data authority, privacy, and copyright concerns [5]. It is often difficult to determine ownership of data obtained from public sources and to ensure informed consent for its use in training machine learning models [6]. The inclusion of such data in training processes is problematic as the resulting models may produce outputs that closely resemble copyrighted or sensitive inputs.

To address these issues, we strongly advocate a paradigm shift to a more decentralized approach, where data providers actively participate in training processes, remain in control over their data and consciously share only the strictly required data to produce joint models. This would enable smaller entities and open source communities to participate in the collaborative training of image generation models without compromising their privacy and data authority, thereby decreasing the data and power concentration within Big Tech. A technique that suits this idea is Federated Learning.

**Federated Learning (FL)** [7] is a distributed optimization technique that allows multiple clients to collaboratively train a model by leveraging local data. During each training round, a subset of the clients is asked to perform model updates with local data. The local model updates are sent to a central federator server, which performs a global model update based on the aggregated local updates. The updated model is then broadcast to all clients. FL allows for a diverse range of data among clients to be harnessed to build robust models without directly sharing raw data with others, thereby ensuring greater privacy and smaller communication overheads than collaborative methods where raw data is exchanged.

Most of the FL applications today focus on classification and regression tasks. For instance, banks use collaboratively trained models to detect fraudulent transactions [8], whereas healthcare providers jointly classify sensor data to enhance hospital treatments [9]. Federated Learning has also proven to be effective in training large language models across many devices for next-word prediction [10].

In the domain of image generation, the use of Federated Learning is still an active research area. Statistical heterogeneity across client datasets and large communication overheads are key challenges in FL [11] that must be overcome to make federated image generation successful. Existing works such as [12, 13] describe federated techniques based on Generative Adversarial Networks (GANs) [14]. However, to the best of our knowledge, no federated algorithms have yet been proposed for diffusion models.

**Diffusion models** are a type of probabilistic generative models that use noise to gradually destruct training images through multiple forward steps and then learn the reverse denoising process with a neural network to generate new images of the target distribution, given any input of random noise [15]. Diffusion models are state-of-the-art for image generation as they are more stable in convergence and produce images with higher quality than GANs. However, this comes at the cost of being significantly slower [16].

This paper aims to bring FL and diffusion models together. More precisely, we address the following research question:

*How can diffusion models for image generation be trained using federated learning?*

To answer this question, we design a Federated Diffusion Model, FEDDIFF, based on a Denoising Diffusion Probabilistic Model (DDPM) [17] that is trained using the Federated Averaging (FedAvg) algorithm [7]. Additionally, we introduce three novel communication efficient training methods, USPLIT, ULATDEC, and UDEC, that take advantage of the structure of the underlying UNet [18] architecture to reduce the number of communicated parameters during training, whilst maintaining comparable image quality as measured by the FID score [19]. In a nutshell, USPLIT splits parameter updates among clients every round, whereas ULATDEC and UDEC limit the federated training of parameters to specific parts of the network. To compare their effectiveness, we evaluate the performance of FEDDIFF in combination with the different training methods. Finally, we study FEDDIFF under different data distributions and client settings to assess its robustness to statistical heterogeneity.

As a summary, we make the following **contributions**:

- We propose a novel algorithm to train diffusion models in a federated way.

[1] https://shorturl.at/aeqFS

- We describe and compare three novel communication-efficient training methods that take advantage of the model architecture to reduce the number of communicated parameters during training. USPLIT decreases the communication overhead by 25%, ULATDEC by 41% , and UDEC by 74%.

- We compare our models by evaluating the image quality of the output images that they generate under different data distributions and client settings. Our results show comparable image quality to the centralized setting in federated settings with up to ten clients and IID data.

This paper is structured as follows. Section 2 provides background information on federated learning and diffusion models, whereas Section 3 sheds light on related research. Section 4 explains our communication-efficient methods for federated diffusion, which Section 5 tests and compares. Section 6 concludes and provides future work suggestions. Finally, Section 7 elaborates on the ethical aspects of this research.

## 2 Background

In this section, we provide the necessary technical background on different types of diffusion models, with a focus on the DDPM. Furthermore, we provide a formalization of FL and its challenges with statistical heterogeneity.

**Types of Diffusion Models.** Among diffusion models, we distinguish between three predominant formulations. First, Denoising Diffusion Probabilistic Models (DDPMs) [17, 20] estimate a probability distribution over image data using a diffusion process over discrete timesteps, with both forward and reverse processes represented as Markov chains. Second, Score-based Generative Models (SGMs) [21, 22] learn the Stein Score [23], which represents the gradient of the log-density function of the image data. During sampling, noisy inputs pass discrete timesteps in the reverse process at which they are pushed in the direction in which the data density, and thus sample likelihood grows the most. Third, Stochastic Differential Equations (Score SDEs) [24] are the continuous-time generalization of both SGMs and DDPMs that estimate the score function at any time using differential equations.

We choose to focus on the DDPM formulation, mainly because of its simplicity and popularity. The loss-based objective function is easier to optimize than the score-based objectives that SGMs and SDEs use. Once the transition kernels are learned, no numerical methods are required to generate samples, unlike with SDEs. The DDPM is also the most explored and widespread option out of the three [15].

**Denoising Diffusion Models (DDPM).** The DDPM introduced by [17] models a probability distribution $p_\theta(x_0) := \int p_\theta(x_{0:T})dx_{1:T}$, over the pixel space through noisy latents $x_1, ..., x_T$. Given training images $x_0$ from a noiseless target distribution $q(x_0)$, the latents $x_1, ..., x_T$ are obtained following a Markovian forward process $q(x_{1:T})$ that gradually adds Gaussian noise according to a variance schedule $\beta_1, ..., \beta_T$, as given by equations 1 and 2.

$$q(x_{1:T}) := \prod_{t=1}^{T} q(x_t|x_{t-1}) \tag{1}$$

$$q(x_t|x_{t-1}) := \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{2}$$

Provided that the variance schedule is chosen so that $\bar{\alpha}_T = \prod_{s=1}^{T}(1-\beta_s) \to 0$, the distribution of $x_T$ is well approximated by the standard Gaussian (random noise) distribution $p(x_T) \approx \mathcal{N}(x_T; 0, I)$ [15]. In the reverse process, the goal is to create a noiseless sample starting with a sample of random noise. When the $\beta_t$ are sufficiently small, the reverse process has the same functional form as the forward process. Therefore, the reverse process can be defined by a Markov chain $p_\theta(x_{0:T})$ with learned Gaussian transitions parameterized by $\theta$, as given by equations 3 and 4.

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

$$p_\theta(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{4}$$

In [17], the variances of the denoising kernels are fixed to a single value: $\Sigma_\theta(x_t, t) = \sigma_t^2 I$, where $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$. However, they can also be learned during training [25]. Instead of approximating $\mu_\theta(x_t, t)$ directly, it is re-parameterized as a function of $\epsilon_\theta(x_t, t)$ to achieve better sampling quality [17]. $\epsilon_\theta(x_t, t)$ approximates the noise $\epsilon_t$ that is to be subtracted from samples $x_t$ at timestep $t$ during the reverse process:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{1-\beta_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)) \tag{5}$$

A special property of the forward process is that:

$$q(x_t|x_0) := \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t x_0}, (1-\bar{\alpha}_t)I) \tag{6}$$

Using this, any noisy latent $x_t$ can be sampled via a single step given the original image $x_0$ and fixed variances $\beta_t$:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t \quad \text{where} \quad \epsilon_t \sim \mathcal{N}(0, I) \tag{7}$$

The training objective can be formulated as minimizing the distance between the real noise $\epsilon_t$ and the noise estimation $\epsilon_\theta(x_t, t)$ by the model for each of the timesteps $t$:

$$\mathcal{L}_{\text{simple}}(\theta) := \mathbb{E}_{t\sim[1,T]}\mathbb{E}_{x_0\sim p(x_0)}\mathbb{E}_{\epsilon_t\sim N(0,I)}\|\epsilon_t - \epsilon_\theta(x_t, t)\|_2^2 \tag{8}$$

Here, $\mathcal{L}_{simple}$ is a simplified objective function derived from the variational lower bound on the negative log-likelihood for parameter $\theta$ ($\mathcal{L}_{vlb}$) [17]. We can learn $\theta$ by using a neural network trained on minimizing $\mathcal{L}_{\text{simple}}$ using Stochastic Gradient Descent (SGD), as shown in Algorithm 1.

**Algorithm 1** DDPM Training Algorithm

**repeat**
  $x_0 \sim q(x_0)$
  $t \sim Uniform(\{1, ..., T\})$
  $\epsilon_t \sim \mathcal{N}(0, I)$
  Take a gradient descent step on
    $\nabla_\theta \|\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon_t, t)\|_2^2$
**until** converged

Once trained, a DDPM can generate images via Algorithm 2.

**Algorithm 2** DDPM Sampling Algorithm

$x_T \sim \mathcal{N}(0, I)$
**for** $t = T$ **down to** 1 **do**
  $z \sim \mathcal{N}(0, I)$ if $t > 1$, else $z = 0$
  $x_{t-1} = \mu_\theta(x_t, t) + \sigma_t z$
**end for**
**return** $x_0$

Figure 1 shows the intuition behind the DDPM model.

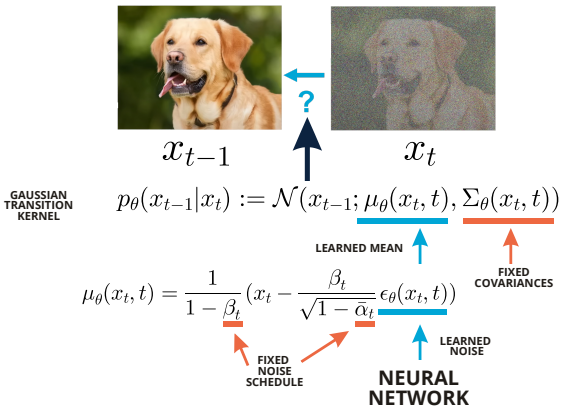


Figure 1: Graphical representation of the intuition behind the DDPM. The reverse denoising process uses Gaussian transition kernels with fixed covariances $\Sigma_\theta(x_t, t)$ and means $\mu_\theta(x_t, t)$ that are learned using a neural network predicting the noise $\epsilon_\theta(x_t, t)$ to subtract from samples $x_t$ at each timestep $t$.

**Formalization of Federated Learning.** A typical federated learning problem can be formulated as a distributed optimization problem involving $K$ clients with the following objective function to minimize [7]:

$$\min_{\theta \in \mathbb{R}^d} f(\theta), \quad \text{where} \quad f(\theta) := \frac{1}{K} \sum_{k=1}^{K} w_k f_k(\theta) \tag{9}$$

For a deep learning problem, $f_k(\theta)$ typically represents the loss incurred over a local client dataset $D_k \subset D$ under global model parameter vector $\theta$. The impact $w_k$ that a client $k$ has on the global objective is often weighted by the relative size of its dataset so that $w_k = \frac{|D_k|}{|D|}$.

**Statistical Heterogeneity in Federated Learning.** If the client datasets $D_k$ are formed by distributing the training examples over the $K$ clients uniformly at random, we say that the data is Independent and Identically Distributed (IID). In this case, we have that $\mathbb{E}_{D_k}[f_k(\theta)] = f(\theta)$ for all clients [7]. Cases where this does not hold are referred to as statistically heterogeneous or non-IID. In such cases, there is no guarantee that the $f_k$ estimate $f$ well. Dealing with statistical heterogeneity is one of the main challenges within FL [11, 26].

Considering the federated diffusion scenario, we focus on two causes for statistical heterogeneity. First, there can be significant differences in the number of training images each client contributed, referred to as **quantity skew**. To address this, client model updates can be weighted based on their respective dataset sizes [27]. Second, in the context of labeled datasets, image label distributions may vary among clients, which is known as **label distribution skew**. Handling label distribution skew can be challenging because each client tends to adjust its local model toward its most dominant labels, resulting in different update directions that need to be combined [27].

## 3 Related Work

We have not been able to identify directly related work on the combination of diffusion models and FL. However, a federated algorithm to train image segmentation models with a similar architecture as diffusion models has been proposed by [28]. Moreover, remarkable explorations have been made regarding alternative solutions for federated image generation based on GANs [12, 13]. Additionally, numerous papers focused on enhancing communication efficiency within the context of FL [29–31]. Last, it is worth mentioning Latent Diffusion Models (LDMs) [1], which perform the diffusion process in a low dimensional latent space, resulting in fewer parameters to optimize and exchange.

**Federated UNet.** The transition kernels for the reverse process of diffusion models are usually learned using architectures that build upon the UNet [18] convolutional network [1, 16, 17, 25]. As the UNet model was initially developed for image segmentation, it is no surprise that the first federated solution centers around this task. Namely, [28] introduces a federated UNet model to segment satellite images based on land use. Aggregation of the local model updates at the federator is performed using FedAvg [7]. The model is shown to perform well on label-skewed datasets. However, the used datasets contain few images, which is typical for image segmentation problems but differs from the image generation scenario. The authors further claim spectacular compression rates for both the number of parameters as well as the memory taken by these parameters, although no further details are provided.

**Federated Image Generation.** Generative Adversarial Networks (GANs) [14] used to dominate the field of image generation before diffusion models surpassed them in terms of image fidelity and training stability [16]. GANs differ from diffusion models in terms of their architectural approach. Diffusion models utilize a single network to make noise predictions at each timestep of the denoising process, whereas

GANs employ two networks: a generator that directly generates output images from noise, and a discriminator that classifies the produced images as real or fake to steer the generator.

GANs have a rich research history that also includes the cross-silo [26] federated setting. Specifically, [13] proposed a federated GAN framework and tested different synchronization strategies with up to six clients to determine whether training either the generator or discriminator collaboratively whilst training the other locally would yield comparable results to training both components in a federated manner, which was found not to be the case. Additionally, the study revealed that federated training of GANs becomes less effective when the data distribution is more skewed and that this effect becomes more pronounced as the number of clients increases. We pose a similar hypothesis for federated diffusion.

Alternatively, [12] proposes a communication-efficient method where the discriminator and generator are trained by averaging over the local parameter values only every $K$ rounds. They show that the model's performance is robust to increasing the synchronization interval K, in a setting with five clients. Additionally, they provide a formal proof on the convergence of the algorithm in non-IID scenarios.

**Improving Communication Efficiency.** Various works have looked into compression and quantization methods to reduce message size in FL [29–31]. With stochastic $k$-level quantization, a limited number of $\log_2 k$ bits is used to represent each of the coordinates within a gradient vector. Each coordinate is rounded to one of the $k$ evenly spread levels between the minimum and maximum value of the corresponding coordinate. Variable length encodings for each of the coordinates can subsequently be applied to further reduce the number of bits transmitted to the federator [30].

Alternative methods include gradient sparsification, where only a subset of the gradients is sent to the federator based on absolute values, thresholds, or random bitmasks, and low-rank decomposition, where a model update is represented as the product of two low-rank matrices, out of which only one is trained and sent to the federator, whilst the other is initialized randomly every round [29].

More recently, [32] introduced correlated quantization, which uses shared randomness to introduce correlation between the local quantizers at each client, improving error bounds and speeding up convergence. The main intuition behind correlated quantization is that if the first client rounds up its value, the second client should round down its value to reduce the mean squared error.

The research on compression and quantization methods is mainly based on general statistical methods that could also be applied to diffusion gradients. However, none of the methods seems to take advantage of the underlying model architecture, so that we consider them orthogonal to our work.

**Latent Diffusion Models.** A recent breakthrough in diffusion research is the Latent Diffusion Model (LDM) [1], where the diffusion process takes place in a latent space of reduced dimensionality rather than the high dimensional RGB picture space. It was found that most of the bits from input images relate to perceptual rather than semantic or conceptual composition so that the images could aggressively be compressed without losing information about the latter. A major benefit of this approach is the reduced number of parameters to be optimized in the UNet [18] to approximate the denoising process. This is especially fruitful in a federated setting where the weights have to be sent back and forth between clients and the federator. A downside of this approach is that it requires a separately trained encoder and decoder to convert between the image and latent space.

# 4 Communication Efficient Federated Diffusion

In this section, we explain our federated diffusion algorithm FEDDIFF as well as the underlying UNet architecture and our communication efficient training methods, USPLIT, UDEC, and ULATDEC, which take advantage of this architecture.

**Federated Diffusion.** In our federated diffusion scenario, we consider a cross-silo setting [26] with a small set of $K$ clients equipped with reasonable computing power and relatively large datasets $D_k \in D$. We use the Federated Averaging (FedAvg) algorithm [7] to optimize the objective from Equation 9, as it has proven to be capable of training a wide variety of deep neural networks using relatively few rounds of communication between the federator and the clients.

Initially, we randomly initialize a global model with parameter vector $\theta_0$. We introduce $R$ training rounds in which all clients partake. They receive the latest model parameters $\theta_{r-1}$ from the federator at the start of each round $r$ and perform SGD minimizing $\mathcal{L}_{\text{simple}}$ over their local dataset $D_k$ to produce an updated parameter vector $\theta_r^k$, such as in Algorithm 1. Specifically, we use mini-batch SGD with batch size $B$, and fixed learning rate $\eta$. Parameter $E$ regulates the number of local epochs that every client performs over its dataset every round. At the end of every round, the clients send back $\theta_r^k$ to the federator, which takes a weighted sum over the client vectors using the relative dataset size $\frac{|D_k|}{|D|}$ to produce an updated global model with parameters $\theta_r$. Algorithm 3 details FEDDIFF's pseudocode.

**Algorithm 3** Federated Diffusion (FEDDIFF)

**Input**: Number of clients $K$, number of communication rounds $R$, number of local epochs $E$, local mini-batch size $B$, local datasets $D^k$, learning rate $\eta$, number of diffusion timesteps T and variance schedule $\beta_1, ..., \beta_T$.
**Output**: Global model parameters $\theta_R$

**Federator executes**:
initialize $\theta_0$
$|D| \leftarrow \sum_{k=1}^{K} |D^k|$
**for** $r = 1$ to $R$ **do**
  **for** $k = 1$ to $K$ **do**
    $\theta_r^k \leftarrow$ CLIENTUPDATE$(k, \theta_{r-1})$
  **end for**
  $\theta_r \leftarrow \frac{1}{|D|} \sum_{k=1}^{K} \theta_r^k \cdot |D^k|$
**end for**

**Client executes**:
**function** CLIENTUPDATE$(k, \theta_{r-1})$:

$\theta_r^k \leftarrow \theta_{r-1}$
$\mathcal{B} \leftarrow$ (split $D^k$ into batches of size $B$)
**for** $e = 1$ to $E$ **do**
**for** $b \in \mathcal{B}$ **do**
$\theta_r^k \leftarrow \theta_r^k - \eta \cdot \nabla_{\theta_r^k}$CALCULATELOSS$(b; \theta_r^k)$
**end for**
**end for**
**return** $\theta_r^k$
**end function**

**function** CALCULATELOSS$(b; \theta_r^k)$:
**for** $i \in b$ **do**
$t \sim Uniform(\{1, .., T\})$
$\epsilon_t \sim \mathcal{N}(0, I)$
$\bar{\alpha}_t = \prod_{s=1}^{t}(1 - \beta_t)$
$\mathcal{L}_i = \|\epsilon_t - \epsilon_{\theta_r^k}(\sqrt{\bar{\alpha}_t} i + \sqrt{1 - \bar{\alpha}_t}\epsilon_t, t)\|_2^2$
**end for**
**return** $\frac{1}{|b|}\sum_{i \in b} \mathcal{L}_i$
**end function**

**UNet Architecture.** For every client, we use an identical UNet [18] convolutional neural network to approximate the function $\epsilon_\theta(x_t, t)$. The name "UNet" is derived from the network's U-shaped architecture, which consists of an encoder and a decoder path with what is referred to as a latent bridge or bottleneck in the middle. First, the encoder path gradually downsamples the noisy input images to capture an increasing number of higher-level but lower-resolution feature maps. The bottleneck in the middle can then be leveraged to perform feature selection, after which the decoder path performs upsampling to generate pixel-level predictions of the noise $\epsilon_t$. Skip connections inspired by [33] are employed to bridge the gap between the encoder and decoder, allowing the network to combine both low-level and high-level features effectively.

In our version, the Wide ResNet Blocks [33] used by [17] are replaced by more state-of-the-art ConvNeXt Blocks [34]. Another difference is that we apply three rather than four levels of downsampling because we aim at generating small 28x28 images. Our bottleneck preserves spatial dimensionality and feature map count to allow a smooth gradient flow between the encoder and decoder and straightforward concatenation via the skip connections in the layers above. Parameter sharing over time is accommodated by leveraging transformer sinusoidal position embeddings [35] for the diffusion timesteps $t$, as in [36]. A graphical representation of our UNet model, showing the feature map dimensions and counts resulting from the operations in the encoder, bottleneck, and decoder can be found in Figure 2.

**Communication Efficient Training Methods.** By default FEDDIFF uses what we refer to as the FULL training method, which consists of the federator sending the full parameter vector $\theta$ to each of the $K$ clients and receiving the updated parameter vectors $\theta^k$ from each of the clients during each of the $R$ communication rounds. Let $\theta_{\text{enc}}$, $\theta_{\text{bot}}$, $\theta_{\text{dec}}$ be the parameter vectors associated with the UNets encoder, bottleneck and decoder respectively so that $\theta = \theta_{\text{enc}} \frown \theta_{\text{bot}} \frown \theta_{\text{dec}}$, where the $\frown$ operator denotes vector concatenation. The total communication overhead of FULL is now $\mathcal{O}(R \cdot K \cdot 2|\theta|)$.

We propose two alternative types of training techniques that exploit the structure of the UNet to reduce the total communication overhead incurred during the training process.

**USPLIT** decreases the communication overhead by splitting parameter updates complementarily amongst the clients. The federator initiates each communication round again by sending the full parameter vector $\theta$ to each of the clients so that these can initialize their local model identically. However, each client is assigned a specific subset of the parameters, which can include $\theta_{\text{enc}}$, $\theta_{\text{bot}}$ and/or $\theta_{\text{dec}}$, to report the updates for that round. The global model is then updated using an adapted version of FEDDIFF that only considers the updates from the responsible clients for each network part.

In more detail, tasks are assigned as follows: Every round, we divide the set of clients into random pairs. In each pair, one client reports about the encoder and the other about the decoder. The task of reporting about the bottleneck is randomly assigned to one of the two. If the number of clients is odd, the last client is assigned either the encoder or decoder task randomly, in addition to the bottleneck task.

This task assignment method mimics selecting a random fraction $C = 0.5$ of the clients every round to perform model updates, like in [7]. However, this is now done for each of the network parts independently. By assigning new tasks every round, the federator still gathers information regarding each of the network parts for each of the clients over time, whilst reducing the communication overhead of the client updates by a factor of two. As the communication overhead introduced by the federator remains the same, this results in an overall overhead in $\mathcal{O}(R \cdot K \cdot \frac{3}{2}|\theta|)$.

Alternatively, **UDEC** and **ULATDEC** limit the federated training of the model to a subset of the parameters, and leave the training of the other parameters up to the clients themselves. This results in every client having a composed model with both globally trained as well as locally trained parameters, much like in Transfer Learning [37].

The intuition behind both methods is that the denoising capacity of the UNet can mainly be attributed to the decoder, which creates the noise estimations based on the features extracted and selected by the encoder and bottleneck respectively. Hence, UDEC collaboratively trains (and thus exchanges) only the decoder parameters. As a result, clients have the freedom to utilize their locally trained encoder and bottleneck to extract and select features. This might result in mismatches between the locally selected features and the features expected as inputs to the decoder. ULATDEC aims to mitigate this issue by training the bottleneck collaboratively too, so that the feature selection is more unified. As the bottleneck in our UNet does not perform explicit feature selection by reducing the number of feature maps, we expect little difference in model performance between both methods. UDEC and ULATDEC have a communication overhead of $\mathcal{O}(R \cdot K \cdot 2|\theta_{\text{dec}}|)$ and $\mathcal{O}(R \cdot K \cdot 2|\theta_{\text{dec}} \frown \theta_{\text{bot}}|)$ respectively.

## 5 Experimental Setup and Results

In this section, we first describe our experimental setup and evaluation metrics. Then we describe the different experiments that we carried out to quantitatively evaluate our meth-

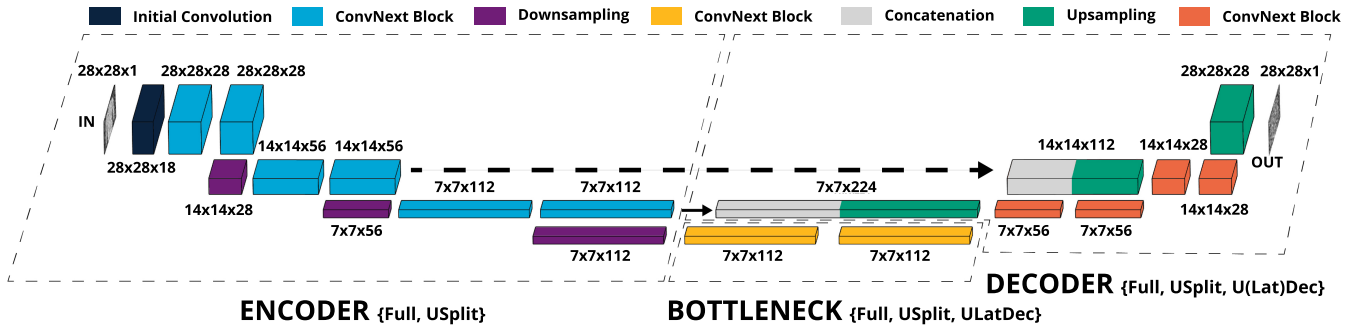


Figure 2: UNet depiction showing the widths, heights, and counts for the feature maps resulting from the different operations in the encoder, bottleneck, and decoder. For each network part, the training methods that consider it for federated training are indicated within the brackets.

ods in different federated settings and discuss their results.

**Experimental Details.** All models were implemented using the PyTorch framework. We used the Fashion-MNIST dataset [38], which consists of 60,000 training and 10,000 test images of 10 different fashion items in grayscale, each having 28x28 pixels. The diffusion parameters from [17] were adopted, specifically $T = 1000$ and the linear diffusion schedule ranging from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. Our model of choice was the UNet, as discussed in Section 4, which contained a total of 2,996,315 parameters. For the SGD optimizer, we used local batch size $B = 128$ and learning rate $\eta = 10^{-4}$. To damp out gradient oscillations, we employed the Adam optimizer [39]. All experiments were conducted on a single NVIDIA GeForce RTX 3090 GPU with CUDA 11.7. We performed 5 runs per experiment and reported averages.

**Evaluation Metrics.** To evaluate the communication efficiency of our models, we reported the cumulative number of communicated parameters between the federator and all clients during model training ($N$). To measure image quality, we used the widespread Fréchet Inception Distance (FID) [19], which measures the distance between a target distribution and a distribution of generated samples based on mean vectors and covariance matrices extracted by a pre-trained Inception V3 model [40]. The lower the FID, the better the image quality. Usually, 50.000 images per distribution are used to extract the required statistics, but given the slow diffusion sampling and the fact that our global test set only contained 10,000 images, we decided to use 5,000 images instead. We measured the FIDs on client level, given that the federator only had access to partial models with ULATDEC and UDEC.

**Establishing a Centralized Baseline.** We first considered the centralized setting where $K = 1$ and trained models with $R = 30$. We visually estimated the quality of the output images and found this to be sufficient after 10 rounds of training. Hence, we set the corresponding mean FID of 72 as the image artifact threshold, below which quality was deemed acceptable. We further established that there was little improvement from round 15 onwards. Hence, we set the corresponding mean FID of 43 as the centralized baseline and fixed $R = 15$ for the federated setting to compare with.

**Testing the Federated Setting.** Next, we conducted experiments in the FULL federated setting, testing different numbers of clients $K \in \{2, 5, 10\}$ on IID data using $R = 15$ and $E = 1$. Figure 3 demonstrates that the FID scores quickly surpassed the artifact threshold as the number of clients increased. To achieve better FID scores without increasing the number of communication rounds, we explored different numbers of local epochs $E \in \{2, 3, 5, 8\}$ per communication round. As shown in Figure 3, increasing $E$ significantly improved the FID scores. The higher the number of clients $K$, the more local epochs $E$ were required to bring the FID scores under the artifact threshold. However, the training time linearly increased with $E$. To strike a balance between training time and output quality, we opted for $E = 5$, which yielded FID scores that were comparable with the centralized baseline, whilst maintaining reasonable maximum training times at around 30 minutes per model.

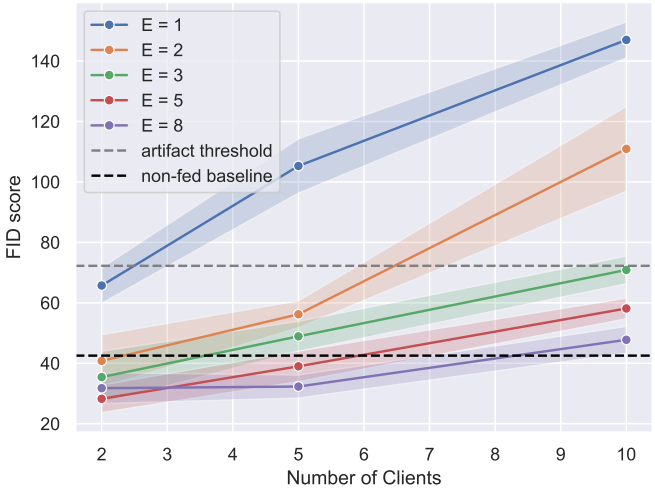


Figure 3: Mean FID scores with error bounds for different number of clients $K$ and local epochs $E$ with $R = 15$ in the FULL federated setting on IID data.

**Comparison of the Training Methods.** With the number of epochs $E = 5$ and global communication rounds $R = 15$ fixed, we compared the FULL federated training with USPLIT, ULATDEC and UDEC in terms of the cumulative number of communicated parameters $N$ and the resulting FIDs for different number of clients $K \in \{2, 5, 10\}$ with IID data.

Figure 4 shows the linear development of $N$ over the train-

ing rounds for each of the methods with $K = 5$, whereas Table 1 shows $N$ for each of the settings. On average, USPLIT achieved a 25% reduction over FULL, where ULATDEC and UDEC achieved a 41% and 74% reduction respectively. These are in correspondence with the Big-$\mathcal{O}$ bounds for communication overhead established in Section 4.

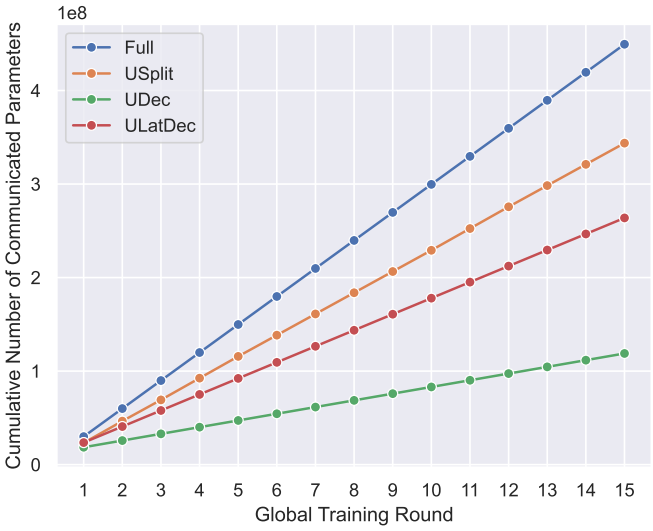


Figure 4: Cumulative number of communicated parameters $(\cdot 10^8)$ during training for the different training methods with $K = 5$.

Table 1 shows comparable FID scores for USPLIT and FULL in the IID setting, where UDEC and ULATDEC have higher FID scores. There is little difference between the latter two, which is in line with our hypothesis that training the latent bridge in a federated manner would not significantly improve the image quality for our version of the UNet. In future work, we plan to explore different bottleneck configurations to investigate their effect on both training methods.

Another noteworthy observation concerns the higher standard deviations for UDEC and ULATDEC, in comparison to FULL and USPLIT. These can be attributed to performance variations across local client models resulting from partial federated training, as elucidated in Table 2. For instance, the FID scores of Client 3 are twice as high as those of Client 1, indicating that Client 1 was strikingly more successful in training the encoder and bottleneck locally than Client 3, even though their training data was IID.

Lastly, we can see that for $K \in \{2, 5\}$, the mean FID scores are below the image artifact threshold for each of the methods. Together with the actual outputs shown in Figure 5, this proves that even with a 74% reduction in $N$, images with quality comparable to the centralized baseline can be generated in a federated setting with IID data. FULL and USPLIT are also able to deal with $K = 10$, although the FID scores are significantly worse than for $K = 2$ and $K = 5$. UDEC and ULATDEC fail to produce images of sufficient quality with $K = 10$.

In general, the FID scores tend to rise as the number of clients increases, suggesting the need to increase either $R$ or $E$ in scenarios involving a larger number of clients. In future work, we therefore plan to plot the FID scores over different higher round numbers, which will require more time than currently available (to give an indication: completing the experiments that gave rise to Table 1 took an entire week).

Table 1: FID scores and number of communicated parameters $N$ for different training methods, numbers of clients $K$ and data distributions, using $R = 15$ and $E = 5$. The baseline uses $E = 1$. The * denotes that the FID scores have been averaged over all local client models. Scores that exceed the artifact threshold of 72 within one standard deviation are marked in orange.

| **Method** | **K** | **N** $(\cdot 10^6)$ | **FID** | | |
|---|---|---|---|---|---|
| | | | **IID** | **l-skew** | **q-skew** |
| BASELINE | 1 | 0 | $43 \pm 1$ | n/a | n/a |
| FULL | 2 | 179.78 | $39 \pm 2$ | $33 \pm 1$ | $33 \pm 3$ |
| | 5 | 449.45 | $39 \pm 4$ | $43 \pm 4$ | $23 \pm 5$ |
| | 10 | 898.89 | $61 \pm 2$ | $64 \pm 3$ | $76 \pm 11$ |
| USPLIT | 2 | 134.83 | $37 \pm 3$ | $38 \pm 4$ | $55 \pm 4$ |
| | 5 | 343.73 | $41 \pm 5$ | $61 \pm 5$ | $39 \pm 9$ |
| | 10 | 674.17 | $62 \pm 3$ | $70 \pm 8$ | $87 \pm 19$ |
| ULATDEC* | 2 | 105.50 | $45 \pm 13$ | $49 \pm 4$ | $54 \pm 24$ |
| | 5 | 263.75 | $53 \pm 15$ | $72 \pm 30$ | $122 \pm 138$ |
| | 10 | 527.51 | $70 \pm 14$ | $101 \pm 83$ | $137 \pm 125$ |
| UDEC* | 2 | 47.54 | $49 \pm 16$ | $49 \pm 5$ | $78 \pm 48$ |
| | 5 | 118.85 | $51 \pm 15$ | $75 \pm 31$ | $139 \pm 135$ |
| | 10 | 237.69 | $72 \pm 20$ | $98 \pm 67$ | $147 \pm 119$ |

Table 2: Averaged FID scores for the local client models resulting from UDEC and ULATDEC training on IID data with $K = 5$.

| **Local Model** | UDEC | ULATDEC |
|---|---|---|
| Client 0 | 44 | 44 |
| Client 1 | 35 | 36 |
| Client 2 | 46 | 55 |
| Client 3 | 71 | 68 |
| Client 4 | 58 | 60 |

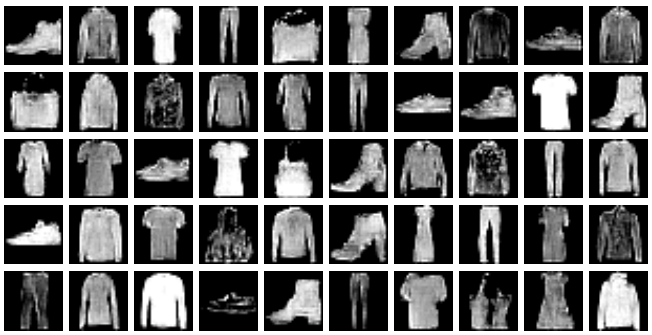

Figure 5: Fashion-MNIST samples generated with the baseline model (first row) and FEDDIFF models trained using the FULL (second row), USPLIT (third row), ULATDEC (fourth row), and UDEC (fifth row) methods with $K = 5$, $R = 15$ and $E = 5$.

**Testing with non-IID data.** To evaluate the robustness of the training methods with respect to statistical heterogeneity, we simulated label distribution skew (l-skew) and quantity skew (q-skew) in our data, using a Dirichlet distribution [41,42]. To

mimic l-skew, we sampled $p_j \sim Dir_K(\beta)$ for every label $j$ and allocated a $p_{j,k}$ proportion of the instances to each client $k$. To mimic q-skew, we sampled $q \sim Dir_K(\beta)$ and allocated a $q_k$ proportion of the total training dataset to each client $k$. Parameter $\beta$ is the concentration parameter. When $\beta \to \infty$, the result is an IID distribution. The closer $\beta$ is to 0, the more skewed the distribution. We fixed $\beta = 0.5$ as in [41]. Figure 6 shows and example of a l-skewed data partition when $K = 5$, where every client has a few major classes with many samples, as well as minor classes with relatively few samples.

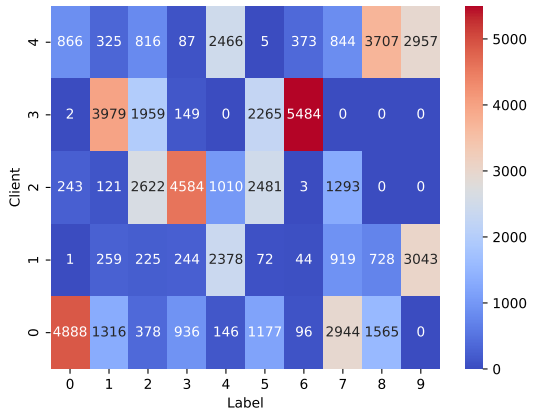


Figure 6: An example of l-skew on the Fashion-MNIST dataset for $K = 5$ using $\beta = 0.5$. Each cell yields the number of images of a certain label assigned to a certain client.

We ran five experiments with l-skew and q-skew for all training methods with $K \in \{2, 5, 10\}$ and reported the averaged FID scores in Table 1. The combination of a large number of clients $K = 10$ together with q-skew or l-skew appeared problematic for most training methods (except FULL), which is in line with our hypothesis based on GAN results from Section 3. Where FULL and USPLIT were able to cope with q-skew in combination with fewer clients, UDEC and ULATDEC failed to cope with it at all. Interestingly, FULL performed extremely well on q-skewed data with $K = 5$, outperforming the IID scenario by far without an explainable reason. FULL appeared robust against l-skew, which is in line with findings by [28] and resulted in similar FID scores as in the IID setting. However, all other methods seem to be affected by l-skew starting from $K = 5$, leading to notable drops in image quality compared to the IID setting.

**Testing with other Datasets.** The choice for the Fashion-MNIST dataset allowed for fast training and evaluation. However, training diffusion models using low-resolution grayscale images forms a drastic simplification of real-world diffusion training tasks. Hence, we were interested in experimenting with higher dimension colored images too. We chose the CelebA dataset [43], which contains over 200k images of celebrities for this purpose. We resized the images to 64x64 and to facilitate the creation of different data distributions, we created 16 different classes among the images based on the combination of sex (male, female), age (young, old) and hair color (black, brown, blond, gray). As some images were not annotated properly, we ended up with a usable dataset comprising of 162,770 training images and 19,962 test images. Using FEDDIFF, we trained a 14,892,477 parameter model over an IID dataset with $K = 5$, $R = 30$, $E = 5$ and $B = 64$, which took over 37 hours. We were able to determine a FID score of 53 after a 5 hour sampling process. The federated model demonstrated its ability to generate realistic faces, as depicted in Figure 7. Regrettably, due to the extensive time required for training and evaluation, we were unable to conduct further experiments.

Figure 7: CelebA samples generated with a FEDDIFF model trained using the FULL method on IID data with $K = 5$.

## 6 Conclusions and Future Work

We have demonstrated that diffusion models can be trained using federated learning by utilizing an adapted Federated Averaging (FedAvg) algorithm to train a UNet-based Denoising Diffusion Probabilistic Model (DPPM). Moreover, we have shown that the images generated by our federated model exhibit comparable quality to those generated by their non-federated counterparts, as evaluated by the FID score. We have also shown our method's robustness to label and quantity-skewed data distributions.

Furthermore, we discovered that complementarily splitting the parameter updates for the encoder, decoder, and bottleneck parts of the UNet among clients every round can enhance communication efficiency during training. This approach led to a 25% reduction in the number of exchanged parameters whilst maintaining image quality comparable to the naive approach, where all parameters are exchanged between the federator and clients every round. However, this method demonstrated limited resilience against label and quantity skew in a federated setting with few clients.

Additionally, we found that training the encoder and bottleneck locally resulted in a significant reduction in communication by up to 74% compared to the naive approach. However, this approach exhibited variations in image quality among the local client models and was only effective when applied to a limited number of clients in conjunction with IID data.

Lastly, we identify several directions for future work. First, our work is limited to the DDPM formulation so that federated solutions for SDEs and SGMs are still to be explored. Second, we believe robustness against non-IID data distributions could be improved by experimenting with alternative aggregation methods beyond FedAvg. Third, one could establish theoretical bounds for the convergence of our proposed methods. Finally, our methods could be combined with Latent Diffusion Models (LDMs) to work with more challenging and higher-resolution datasets.

## 7 Responsible Research

Our research adheres to responsible and ethical practices, prioritizing reproducibility and transparency. We made our research fully reproducible by providing the complete source code, parameters, and techniques used in our experiments. This allows others to replicate, validate, and build upon our work. Additionally, we maintained transparency by saving all intermediate models generated during the research process. Lastly, we followed ethical guidelines to ensure that no sensitive or private data was used during our research.

The significant impact of generative AI on society highlights the need to anticipate and address potential ethical implications. The accessibility of pre-trained diffusion models capable of generating highly realistic outputs has facilitated the creation of deceptive and malicious content resembling real images. Our research contributes to enabling the general public to not only use but also train these models, which raises concerns about amplifying these risks. For example, clients could deliberately manipulate their local dataset to promote the creation of malicious content by a global model. It is important to conduct further research into such byzantine behaviors and introduce regulations before deploying large-scale federated generative models in practice.

However, we believe that the benefits of federated training for diffusion models, such as improved privacy, data authority, and reduced dependence on Big Tech companies will eventually outweigh the aforementioned risks. That is, when the generation and spread of disinformation through federated learning can effectively be detected and legally penalized.

## References


[1] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 2022, pp. 10 674–10 685. [Online]. Available: https://doi.org/10.1109/CVPR52688.2022.01042

[2] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi, "Photorealistic text-to-image diffusion models with deep language understanding," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 36 479–36 494. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/ec795aeadae0b7d230fa35cbaf04c041-Paper-Conference.pdf

[3] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," *ArXiv*, vol. abs/2204.06125, 2022.

[4] A. Kak and S. M. West, "Ai now 2023 landscape: Confronting tech power," AI Now Institute, Report, April 11 2023. [Online]. Available: https://ainowinstitute.org/2023-landscape

[5] G. Franceschelli and M. Musolesi, "Copyright in generative deep learning," *Data Policy*, vol. 4, p. e17, 2022.

[6] A. J. Andreotta, N. Kirkham, and M. Rizzi, "Ai, big data, and the future of consent," *AI & SOCIETY*, vol. 37, no. 4, pp. 1715–1728, Dec 2022. [Online]. Available: https://doi.org/10.1007/s00146-021-01262-5

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[8] D. Myalil, M. Rajan, M. Apte, and S. Lodha, "Robust collaborative fraudulent transaction detection using federated learning," 2021, Conference paper, p. 373 – 378, cited by: 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85125868738&doi=10.1109%2fICMLA52953.2021.00064&partnerID=40&md5=cf7c3ef35a5c5e083159a7e02adf6ade

[9] L. Sun and J. Wu, "A scalable and transferable federated learning system for classifying healthcare sensor data," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 2, pp. 866–877, 2023.

[10] A. Hard, K. Rao, R. Mathews, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *ArXiv*, vol. abs/1811.03604, 2018.

[11] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[12] M. Rasouli, T. Sun, and R. Rajagopal, "Fedgan: Federated generative adversarial networks for distributed data," *ArXiv*, vol. abs/2006.07228, 2020.

[13] C. Fan and P. Liu, "Federated generative adversarial learning," in *Pattern Recognition and Computer Vision*, Y. Peng, Q. Liu, H. Lu, Z. Sun, C. Liu, X. Chen, H. Zha, and J. Yang, Eds. Cham: Springer International Publishing, 2020, pp. 3–15.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, p. 139 – 144, 2020, cited by: 1524; All Open Access, Bronze Open Access, Green Open Access. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85094930828&doi=10.1145%2f3422622&partnerID=40&md5=8a19ba3da390316c8f9c43b8a6c18ef1

[15] F.-A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, "Diffusion models in vision: A survey," *IEEE Transac-*

*tions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2023.

[16] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 8780–8794. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf

[17] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[18] O. Ronneberger, P.Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, vol. 9351. Springer, 2015, pp. 234–241, (available on arXiv:1505.04597 [cs.CV]). [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a

[19] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf

[20] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 2256–2265. [Online]. Available: https://proceedings.mlr.press/v37/sohl-dickstein15.html

[21] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf

[22] Y. song and S. Ermon, "Improved techniques for training score-based generative models," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 12 438–12 448. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/92c3b916311a5517d9290576e3ea37ad-Paper.pdf

[23] Q. Liu, J. Lee, and M. Jordan, "A kernelized stein discrepancy for goodness-of-fit tests," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 276–284. [Online]. Available: https://proceedings.mlr.press/v48/liub16.html

[24] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. [Online]. Available: https://openreview.net/forum?id=PxTIG12RRHS

[25] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 8162–8171. [Online]. Available: https://proceedings.mlr.press/v139/nichol21a.html

[26] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021. [Online]. Available: http://dx.doi.org/10.1561/2200000083

[27] X. Ma, J. Zhu, Z. Lin, S. Chen, and Y. Qin, "A state-of-the-art survey on solving non-iid data in federated learning," *Future Generation Computer Systems*, vol. 135, pp. 244–258, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X22001686

[28] R. Kanagavelu, K. Dua, P. Garai, N. Thomas, S. Elias, S. Elias, Q. Wei, L. Yong, and G. S. M. Rick, "Fedukd: Federated unet model with knowledge distillation for land use classification from satellite and street views," *Electronics*, vol. 12, no. 4, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/4/896

[29] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016. [Online]. Available: https://arxiv.org/abs/1610.05492

[30] A. T. Suresh, F. X. Yu, H. B. McMahan, and S. Kumar, "Distributed mean estimation with limited communication," in *International Conference on Machine Learning*, 2017. [Online]. Available: https://arxiv.org/abs/1611.00429

[31] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/6c340f25839e6acdc73414517203f5f0-Paper.pdf

[32] A. T. Suresh, Z. Sun, J. Ro, and F. X. Yu, "Correlated quantization for distributed mean estimation and optimization," in *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 2022, pp. 20 856–20 876. [Online]. Available: https://proceedings.mlr.press/v162/suresh22a.html

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '16. IEEE, Jun. 2016, pp. 770–778. [Online]. Available: http://ieeexplore.ieee.org/document/7780459

[34] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 2022, pp. 11 966–11 976. [Online]. Available: https://doi.org/10.1109/CVPR52688.2022.01167

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[36] N. Rogge and K. Rasul, "The annotated diffusion model," Jun 2022. [Online]. Available: https://huggingface.co/blog/annotated-diffusion

[37] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, p. 9, 2016.

[38] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *ArXiv*, vol. abs/1708.07747, 2017.

[39] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015.

[40] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 2818–2826. [Online]. Available: https://doi.org/10.1109/CVPR.2016.308

[41] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 7252–7261. [Online]. Available: https://proceedings.mlr.press/v97/yurochkin19a.html

[42] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-iid data silos: An experimental study," *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pp. 965–978, 2021.

[43] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 3730–3738.