SPIRE: Conditional Personalization for Federated Diffusion Generative Models

Kaan Ozkara
Amazon Annapurna Labs
kaanozka@amazon.com

Ruida Zhou
Amazon AGI
zruida@amazon.com

Suhas Diggavi
UCLA
suhas@ee.ucla.edu

Abstract

Two defining characteristics of federated learning (FL) client data are distributional heterogeneity and small local sample sizes. These properties necessitate data efficient, and client specific adaptation rather than a one-size-fits-all model. Recent advances in diffusion models have revolutionized generative AI. However, their scale is too large for straightforward fine-tuning; making personalization difficult. To enable personalized diffusion generative models, we propose Shared-backbone Personal Identity Representation Embeddings (SPIRE), a framework that casts per-client diffusion based generation as conditional generation in FL. SPIRE factorizes the network into (i) a high-capacity global backbone that learns a population-level score function and (ii) lightweight, learnable client embeddings that encode local data statistics. This separation enables parameter-efficient fine-tuning that touches < 0.01 of weights. We provide the first theoretical bridge between conditional diffusion training and maximum-likelihood estimation in Gaussian-mixture models. For a two-component mixture we prove that gradient descent on the DDPM with respect to mixing weights loss recovers the optimal mixing weights and enjoys dimension-free error bounds. Our analysis also hints at how client embeddings act as biases that steer a shared score network toward personalized distributions. Empirically, SPIRE matches or surpasses strong baselines during collaborative pre-training, and vastly outperforms them when adapting to unseen/new clients—reducing Kernel Inception Distance while updating only hundreds of parameters. SPIRE further mitigates catastrophic forgetting and remains robust across fine-tuning learning-rate and epoch choices.

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300. Copyright 2026 by the author(s).

1 INTRODUCTION

Modern diffusion models have achieved striking success in image, audio and video synthesis with their ever-growing scale. But in a personalized federated generative task the central challenge is statistical: client data are scarce, imbalanced, and non-IID. A single global generator often fails to capture the client-specific distributions induced by distributional characteristics. These discrepancies are most pronounced for *new clients*, who arrive after the global model has been trained and possess only a handful of local samples, when a straightforward fine-tuning may be catastrophic for the performance. A key question is, how can we enable a large diffusion model with the flexibility to personalize quickly—even when users' data are scarce and heterogeneous?

Our key insight. Many real-world data sets share a *high-dimensional structure* that is common across clients (e.g., the geometry of natural images) plus a *low-dimensional* client-specific component (e.g., class priors, color statistics or individual identities). We formalize this intuition through a simple heterogeneity model in which shared parameters ϕ captures the complex structure, whereas each client controls lightweight parameters γ_j. For diffusion models we operate the split by (i) training a *shared backbone* that learns a population score function and (ii) injecting a *conditioning embedding* that acts as a signature of client statistics. Personalization thus reduces to jointly learning ϕ from all users and individually learning γ_j—a few hundred parameters instead of millions—making it feasible to adapt on-device. Our key contributions can be summarized as follows.

- We introduce SPIRE, a framework that treats per-client diffusion modeling as a conditional generation problem in FL. SPIRE decouples global and local learning through an embedding-based conditioning mechanism, enabling parameter-efficient fine-tuning for new clients, who have not participated in the federated pretraining.
- We provide the *first theoretical analysis* connecting conditional diffusion training to maximum-likelihood estimation in Gaussian-mixture models (GMMs). In the two-component case we show that gradient descent on the DDPM objective with respect to mixing weights (which corresponds to low dimensional client specific parameters γ_j) finds the global maximum and derive dimension-free error bounds (Theorem 4.2). The GMM also provides a theoretical insight into conditioning mechanism in our architecture, including bias to steer personalization.
- We develop a practical FL algorithm that trains backbone and embeddings jointly, requires *no additional communication* during personalization, and is robust to catastrophic forgetting. Our experiments on MNIST, CIFAR-10 and CelebA Liu et al. (2015) demonstrate that SPIRE matches or exceeds strong baselines during collaborative pre-training and *vastly outperforms* them when adapting to new clients while updating fewer than 0.01% of the parameters.

Below we relate to closest works, but we provide more detailed related work in Appendix A.

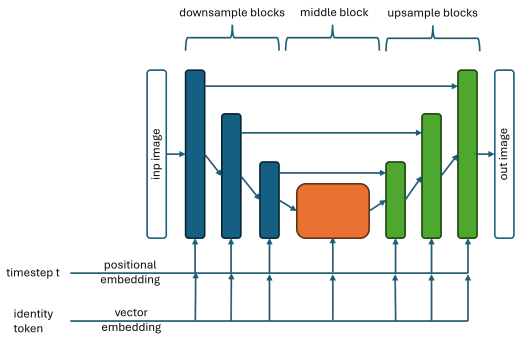
Personalized FL. There has been very limited work on personalized federated diffusion models, despite the vast amount of work for personalized FL, particularly for supervised learning scenarios. The previous personalized FL works have focused on adjusting the optimization criteria (Dinh et al., 2020; Hanzely and Richtárik, 2020; Ozkara et al., 2023), meta learning based approaches (Fallah et al., 2020; Acar et al., 2021; Khodak et al., 2019), clustering based approaches Mansour et al. (2020), knowledge distillation based approaches (Ozkara et al., 2021; Lin et al., 2020) and so on. Conceptually, even though exclusively supervised learning is considered, the most related to our work is shared representation approach in (Collins et al., 2021), where the authors vertically partition neural networks as global feature extractors and local heads. In contrast, our approach is a horizontal partitioning, where high dimensional backbone is trained with light weight conditioning information based on client's data statistics. In terms of the task, (Ozkara et al., 2025) also looks at personalized diffusion models, through a lens of statistical hierarchical Bayesian framework. Their method causes a regularization in the loss function, whereas, ours is based on a conditional architecture. Furthermore their method requires two separate models which is less suitable for new clients. At a high level, FedEM (Marfoq et al., 2021) also uses mixture assumptions but maintains multiple full models per component, which is prohibitive for large diffusion models. pFedFDA (**?**) estimates class-conditional feature statistics explicitly, whereas SPIRE learns client embeddings implicitly. **?** focuses on DP-based diffusion training, and FedBiP (**?**) personalizes a pretrained LDM for one-shot classification—both differ from our multi-round conditional training.

Conditional diffusion models. Conditioning is one of the key properties to add additional information while training diffusion models Dhariwal and Nichol (2021). The most common way of conditioning is to embed the information to a vector space and use in intermediate layers as bias (or modulate the activations). Our work explains this use in a principled way and connects it to a theoretical view (see Section 4), for the first time, as far as we are aware. We utilize the embeddings to estimate a function that extracts the client data statistics. Furthermore, our work utilizes conditioning mechanism as a way to do PEFT, we are not aware of any other works that consider conditioning for FL and new clients.

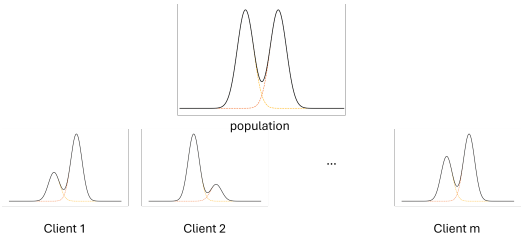
Diffusion models for GMMs. The *two-component symmetric Gaussian-mixture model (GMM)* has long served as a testbed for analyzing EM-algorithm convergence (see, e.g., (Daskalakis et al., 2017; Kwon and Caramanis, 2020)). (Shah et al., 2023) shows that diffusion models (DDPM loss) can be learned through gradient descent to estimate GMM score function when mixing weights, covariances are known. (Chen et al., 2024), further shows diffusion models can be used to efficiently learn general GMMs, through piecewise polynomial function approximations. In Section 4, our method extends the former; in a distributed and heterogeneous setup, we use gradient descent to learn the mixing weights of the GMMs and prove dimension-free bounds.

2 PROBLEM SETUP: PERSONALIZED FEDERATED DIFFUSION GENERATIVE MODELS

In this section, we first introduce a heterogeneity model of clients' data that is partitioned into two: a complex common structure and a lower dimensional individual structure that is specific to clients. A concrete example is a mixture model where high dimensional cluster centers are common for every client but mixing weights are individual (e.g. see Figure1b). Later, we will explain (conditional) diffusion models for personalization and the overall federated learning



(a) Overall diffusion model architecture.



(b) Model of heterogeneity for GMMs.

Figure 1: (Left) Each client holds an identity token which is mapped to a vector embedding that is trained throughout with the backbone. During the inference, a client can use its identity token to do a personalized generation. (Right) In GMM heterogeneity model, all clients share the same means but the mixing weights are individual.

setup that we consider. Lastly, we will explain how the formulation is especially useful for new clients.

2.1 Problem setup and background

We consider a conditional personalization approach for learning individual diffusion models tailored to local data from clients. Let us denote $q^{(j)}(x)$ as the data distribution of a particular client j. We assume $q^{(j)}(x)$ is implicitly conditioned on two types of variables, i.e. $q^{(j)}(x) = q^{(j)}(x|\phi, \gamma_j)$ where $\phi \in \Phi$ is a high dimensional shared parameter, and $\gamma_j \in \Gamma$ is a lower dimensional parameter that is specific to a client j. Note that through the Bayes rule, the pdf implies a mapping from data distribution of a client and the shared parameters to γ_j, there is a (random/data dependent) mapping such that $\gamma_j = g(x, \phi), x \sim q^{(j)}$. To motivate, we give a theoretical and an empirical example.

Example 2.1. *Consider data generated by a two component GMM with mean, mixing weights, and covariance parameters* $(\pm\mu, \{w_j\}_{j=1}^m, \Sigma)$*, each client shares the high dimensional mean parameter* μ *but individually possesses different mixing weights* w_j *of the components. In this example* ϕ, γ_j *corresponds to* μ, w_j *respectively.*

Example 2.2. *Consider a non-parametric score function that is approximated through a neural network with parameters* $\boldsymbol{\theta}_j$*, for a client* j*. Potentially,* $\boldsymbol{\theta}_j$ *can be partitioned into high dimensional* $\boldsymbol{\theta}_{bb}$ *that is learned collaboratively and low dimensional* $\boldsymbol{e}_j$ *that is learned locally. The goal is to use* $\boldsymbol{\theta}_{bb}, \boldsymbol{e}_j$ *as proxies for* ϕ, γ_j *respectively. Ideally,* $\boldsymbol{\theta}_{bb}$ *should describe a general structure in generation task, and* $\boldsymbol{e}_j$ *should add personalization according to input data statistics.*

The goal of the personalized diffusion models is to learn $q^{(j)}(x)$ that is the individual conditional data distribution of a client given ϕ, γ_j. In federated learning, every client holds a small dataset $\{X_i^{(j)}\}_{i=1}^n \sim q^{(j)}$ (assuming same n for simplicity), yet the globally data is from m participating clients, where m is assumed to be larger than n. This abundance enables collaborative learning of the complex ϕ, while each client estimates its low-dimensional γ_j locally. New clients, absent from pre-training, can fine-tune only γ_j with few samples, achieving sample-efficient personalization.

Diffusion Models. Conventionally, diffusion models are composed of several parts. First, a *forward process* that mathematically describes the gradual transformation from a true sample to noise. A corresponding *backward process* is necessary to formalize how the score function can be used to denoise a noisy sample to obtain a data point from the true distribution. Lastly, since the true score function is unknown one needs to formulate an optimization problem (i.e. *score matching*) to estimate it to be used in the backward process. Due to space limitations we define the original diffusion model in Appendix B. Here we introduce the personalized diffusion models. In the personalized FL setup, we would like to generate samples from clients' individual probability distributions. We first start with the forward process. The forward diffusion process is defined as

$$\mathrm{d}X_t^{(j)} = -\delta_t X_t^{(j)}\,\mathrm{d}t + \sqrt{2\delta_t}\,\mathrm{d}W_t, \quad X_0^{(j)} \sim q_0^{(j)}(x).$$

here $q_0^{(j)}(x)$ denotes the true underlying distribution that generates client js data, similarly $q_t^{(j)}(x)$ will be used to denote a latent distribution, δ_t is a drift coefficient, commonly assumed a constant e.g. 1. W_t is the standard Brownian motion. $X_t^{(j)} \sim q_t^{(j)}$ is a latent variable. Equivalently, it can be shown (Shah et al.,

2023) for some $\beta_t : \lim_{t\to\infty} \beta_t = 1, \beta_0 = 0$,

$$X_t^{(j)} = \sqrt{1-\beta_t^2} X_0^{(j)} + \beta_t Z_t, \tag{1}$$

where $X_0^{(j)} \sim q_0^{(j)}$, $Z_t \sim \mathcal{N}(0, \mathrm{I})$. Correspondingly, the backward SDE process is defined to formalize recovering a data sample from true distribution, given a corrupted sample and score function,

$$\mathrm{d}\overleftarrow{X}_t^{(j)} = \left[\delta_{T-t}\overleftarrow{X}_t^{(j)} + 2\delta_{T-t}\nabla \log q_{T-t}^{(j)}(\overleftarrow{X}_t^{(j)})\right]\mathrm{d}t + \sqrt{2\delta_{T-t}}\,\mathrm{d}W_t, \quad \overleftarrow{X}_0^{(j)} \sim q_T^{(j)}(\cdot). \tag{2}$$

In the backward process, we use $\overleftarrow{X}_0^{(j)}$ to denote a sample from pure noise $q_T^{(J)}$. $\nabla \log q_{T-t}^{(J)}(\overleftarrow{X}_t^{(j)})$ is the true score function at time $T-t$ (time t in terms of forward process). The true score function is unknown, hence we need to estimate it from the data using a fixed pure noise distribution. To that end, one constructs the score matching optimization criterion,

$$\min_\theta \int_\tau^T \mathbb{E}_{X_t^{(j)} \sim q_t^{(j)}} \left[\left\| s_{\theta_j}(x_t, t) - \nabla \log q_t^{(j)}(X_0^{(j)}) \right\|^2 \right] \mathrm{d}t,$$

where θ_j is a neural network that is assumed to parameterize the score function. Equivalent to score matching is the DDPM view, which yields Chen et al. (2023),

$$\min_\theta \int_\tau^T \mathbb{E}_{X_0^{(j)} \sim q_0^{(j)}(\cdot), Z_t} \left[\left\| s_{\theta_j}(X_t^{(j)}, t) + \frac{Z_t}{\beta_t} \right\|^2 \right] \mathrm{d}t.$$

3 SPIRE: PERSONALIZED FL USING CONDITIONAL DIFFUSION MODELS

In our work, we consider using client's data (through learning its embedding) as the conditioning information to the model. This results in an horizontal split of the model. During pre-training, $\boldsymbol{\theta}_{bb}, \{\boldsymbol{e}_j\}_j$ are trained jointly, hence, the model implicitly learns to estimate the function $\boldsymbol{e}_j = g^{(j)}(x^{(j)}, \boldsymbol{\theta}_{bb})$. One can see that finding the embedding in the diffusion model training corresponds to extracting data statistics, in other words, $g^{(j)}(x, \theta_{bb}) = \arg\min_{e_j} L_{ddpm}(q^{(j)}, \theta_j)$, where L_{ddpm} is the score matching loss in (3) and $x^{(j)} \sim q^{(j)}(x)$.

Form of $\boldsymbol{\theta}_j$. Our heterogeneity modeling yields separability of $\boldsymbol{\theta}_j$ into two parts. A more complicated (higher dimensional) backbone part $\boldsymbol{\theta}_{bb}$ that is shared among estimated scores of different clients, and a lower dimensional embedding of clients' data statistics, $\boldsymbol{e}_j$, that is individual to each client. We can write $\boldsymbol{\theta}_j = [\boldsymbol{\theta}_{bb}, \boldsymbol{e}_j]$. During training, overall personalized federated DDPM objective consists of finding m different models (with shared backbone and individual identity embedding in our case), $\min_{\{\theta_j\}_{j=1}^m} \frac{1}{m}\sum_{j=1}^m \int_\tau^T \mathbb{E}_{X_0 \sim q_0^{(j)}(\cdot), Z_t}[\|s_{\theta_j}(X_t^{(j)}, t) + \frac{Z_t}{\beta_t}\|^2]\,\mathrm{d}t$. Empirical version of the DDPM loss can be defined through averaging over local dataset.

$$\min_{\{\theta_j\}_{j=1}^m} \frac{1}{m}\sum_{j=1}^m L_{ddpm}^{(j)}(\{X_i^{(j)}\}_{i=1}^n, \theta_j) := \frac{1}{n}\sum_{j=1}^n \int_\tau^T \mathbb{E}_{Z_t}\left[\left\| s_{\theta_j}((X_i^{(j)})_t, t) + \frac{Z_t}{\beta_t}\right\|^2\right] \mathrm{d}t. \tag{3}$$

In practice instead of taking expectation over timesteps, we sample randomly. A client, then, can plug in the estimated score (s_{θ_j}) into the backward process (2) to generate personalized samples.

3.1 Federated training setup.

During training, each client maintains a personalized embedding (through a identity token, e.g., client id number) which acts as a key for the backbone model. The embeddings are kept in client and trained locally, and backbone model is shared and trained by essentially FedAvg. Algorithm1 summarizes the key steps.

Personalization to new clients. Personalization for new clients is one of the key challenges in federated learning. It is especially, pronounced for generative models due to scale of the models. A new client joining the federated ecosystem, later than pre-training stage, can utilize the pre-trained $\boldsymbol{\theta}_{bb}$ and train only its own embedding parameters. This defines a natural way of parameter efficient fine-tuning (PEFT) for personalization, where we are estimating $q^{(new)}$ through $(\boldsymbol{\theta}_{bb}, \boldsymbol{e}_{new})$. See Algorithm 2 in Appendix B, for new client fine-tuning details.

Algorithm 1 Line 1 initializes the shared backbone parameters $\boldsymbol{\theta}_{\mathrm{bb}}$ and the per-client embeddings $\{\mathbf{e}_j\}$. Whenever the synchronization test $\tau \mid k$ succeeds, each client receives the latest backbone (line 5). During each local step the client samples a mini-batch and timesteps, and corrupts the data with the forward process (line 6), and performs a DDPM gradient update on the full parameter vector (line 7). At the next sync point, client sends the updated backbone slice to the server (line 8), and keeps $\mathbf{e}_{j,k+1}$ locally. The server averages the received slices (line 10) to produce $\boldsymbol{\theta}_{\mathrm{bb},k+1}$, which is broadcast at in the next round.

Algorithm 1: Conditional collaborative personalized pre-training for `SPIRE`

Input: Number of iterations K, learning rate (η), batch size b, number of local iterations τ

Initialize backbone model $\boldsymbol{\theta}_{bb}$, and client embeddings $\{\boldsymbol{e}_j\}_{j=1}^m$, set $\boldsymbol{\theta}_j = [\boldsymbol{\theta}_0, \boldsymbol{e}_i]$.
for $k = 1$ **to** K **do**
  for $i = 1$ **to** m **do**
    if τ divides k **then**: Receive $\boldsymbol{\theta}_{bb,k}$ from server and set $\boldsymbol{\theta}_{j,k} = [\boldsymbol{\theta}_{bb,k}, \boldsymbol{e}_{j,k}]$
    Sample $(\{X_i\}_{i=1}^b, \{t_i\}_{i=1}^b)$, use (1) to obtain noisy samples $\{X_{t_i}\}_{i=1}^b$
    Take gradient step on local DDPM loss $\boldsymbol{\theta}_{j,k+1} = \boldsymbol{\theta}_{j,k} - \nabla_{\theta_j} L_{ddpm}^{(j)}(\{X_{t_i}\}_{i=1}^b)$
    if τ divides $k+1$ **then:** extract updated $\theta_{j,bb,k+1}$ from $\theta_{j,k+1}$ and send to server
  end for
  On server aggregate $\theta_{bb,k} = \frac{1}{m}\sum_{j=1}^m \theta_{j,bb,k}$, broadcast to clients
end for

4 THEORETICAL JUSTIFICATION: CONNECTION TO LEARNING MIXING WEIGHTS OF GMMS

Recent work has revisited the same model with *fixed* mixing weights to probe the theoretical foundations of diffusion models and score matching Shah et al. (2023). In our setting we change the emphasis to a distributed setup. The mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ is shared across all clients, while each client j retains its own mixing weight $w_j \in (0,1)$. This maps directly onto our heterogeneity framework, where a global parameter (the shared mean) co-exists with lightweight, client-specific parameters (the scalar weights). Conceptually, it reduces to a one-layer version of our personalized architecture. Detailed proofs of the results in this section is deferred to Appendix B.

4.1 Overview

We focus on a symmetric GMM with two equal components *in the population level* and unknown mixing weights *in client level* as in Figure 1b. In the personalization setup our model consists of shared means $\pm\mu \in \mathbb{R}^d$, and individual mixing weights of the components $\{(w_j, 1-w_j)\}_{j=1}^m$. In particular, the true data distribution for a particular client j is,

$$q^{(j)} = w_j \mathcal{N}(\mu, \mathrm{I}) + (1-w_j)\mathcal{N}(-\mu, \mathrm{I}) \tag{4}$$

where $q^{(j)}$ is the GMM distribution, I is the identity matrix of size $d \times d$, $\mu \in \mathbb{R}^d$ is the true mean parameter, $w_j \in \mathbb{R}$ is the mixing weight for the positive component for client j. This particular mixture distribution is studied in literature, mostly when mixing weights are known and equal, i.e. 1/2 (Shah et al., 2023). Below, we consider the setup for a particular client and omit the subscript in w_j.

Noisy samples. Utilizing the same OU forward process as in Shah et al. (2023) and above in (8), it can be shown (see Appendix B) for $X_t \sim q_t$ we have, $X_t = \exp\{(-t)\}X_0 + \sqrt{1-\exp\{(-2t)\}}Z_t$, with $X_0 \sim q_0$ and $Z_t \sim \mathcal{N}(0, \mathrm{I})$. In the analysis, we will fix a particular timestep t. Accordingly, we will focus on the equivalent DDPM objective:

$$\min_{s_t} \mathbb{E}_{X_0, Z_t}\left[\left\| s_t(X_t) + \frac{Z_t}{\sqrt{1-e^{-2t}}}\right\|^2\right]. \tag{5}$$

Recall, in our case of two component mixture we have, $q_0 = w\mathcal{N}(\mu, \mathrm{I}) + (1-w)\mathcal{N}(-\mu, \mathrm{I})$. Accordingly, q_t can be easily obtained to be (see Appendix B) $q_t = w\mathcal{N}(\mu_t, \mathrm{I}) + (1-w)\mathcal{N}(-\mu_t, \mathrm{I})$, where $\mu_t := \mu\exp\{(-t)\}$. The score function has a closed form solution as $\nabla_x \log q_t(x) = r_1(x)\mu_t + r_2(x)(-\mu_t) - x$ where $r_1(x) = \frac{w\exp\{(-\|x-\mu_t\|^2/2)\}}{w\exp\{(-\|x-\mu_t\|^2/2)\}+(1-w)\exp\{(-\|x+\mu_t\|^2/2)\}}$, and $r_2(x) = 1 - r_1(x)$ are conditional probability of being assigned to a cluster, i.e. 'beliefs'. Accordingly we can obtain the closed form of the score function as outlined in Lemma 4.1 (see Appendix B for the proof).

Lemma 4.1. *The score function at diffusion time step t for q_t is defined as follows ,*

$$\nabla_x \log q_t(x) = \tanh\left(\mu_t^\top x - \frac{1}{2}\log(w/(1-w))\right)\mu_t - x.$$

Connection to conditional architecture. The expression in Lemma 4.1 can be viewed as a *single-layer neural network with a residual (identity) connection.* The shared mean vector $\boldsymbol{\mu}_t$ serves as the layer's weight vector; the log-odds term $\frac{1}{2}\log\left(\frac{w}{1-w}\right)$ is an explicit bias that shifts the pre-activation $\boldsymbol{\mu}_t^\top x$. After applying the non-linearity $\tanh(\cdot)$, the network outputs a vector aligned with $\boldsymbol{\mu}_t$, which is then combined with the skip path $-x$. This construction is exactly the one-layer special case of the conditional architecture introduced in the previous section: the conditioning signal is carried solely by the client-specific mixing weight w, while the weights $\boldsymbol{\mu}_t$ remain global. Hence the lemma illustrates, in its simplest form, how general conditional diffusion models encode per-client information through mixing-weight-induced biases. Training a conditional model can therefore be seen as *finding the appropriate mixing weights* that encode each condition, leaving the heavy shared parameters untouched.

Consequently, we consider 4.1 as the network architecture ($s_t(X_t)$ parameterized by μ_t, w) to estimate the

score function. In the personalized setup, we are interested in a fine-tuning stage where the means are known and each client learns the individual mixing weight. As a result the problem of interest can be rewritten as,

$$\min_{w} \mathbb{E}_{X_0 \sim q^{(j)}(x), Z_t} \Big[\Big\| \tanh\left(\mu_t^\top X_t - \frac{1}{2}\log((1-w)/w)\right)\mu_t - X_t + \frac{Z_t}{\sqrt{1-e^{-2t}}} \Big\|^2 \Big]. \quad (6)$$

Our goal is to examine the MSE at the convergence point; to that end, we relate the first order convergence condition with EM updates Kwon and Caramanis (2020), which are provided in Appendix B.

4.2 Theoretical results

Here, first, we discuss the resulting bounds in the two component GMM problem, for learning the mixing weights. Then we look at score estimation error in a two step procedure where the mean is estimated globally, and then each client learns the mixing weight locally.

Theorem 4.2. *Fixing a μ_t, MSE of estimating w by a gradient method on the DDPM loss results in,*

$$\mathbb{E}_{X_0}(w-\hat{w})^2 \leq \frac{w(1-w)}{n} + \frac{d}{4\|\mu_t\|^2 n}$$

Proof Outline. We start by taking the derivative of (6) with respect to w at a particular timepoint t. Let $u_t = \mu_t^\top X_t - \frac{1}{2}\log(w/(1-w)), f(X_t) = \tanh(u) - X_t + \frac{Z_t}{\sqrt{1-e^{-2t}}}$, by the chain rule, $\frac{\mathrm{d}L}{\mathrm{d}w} = \frac{1}{w(1-w)}\mathbb{E}\left[\tanh'(u_t)(\|\mu_t\|^2\tanh(u_t) + \mu_t^\top X_t)\right]$, where L is the shorthand notation for local DDPM loss. Hence, at the critical point we have the following condition being satisfied,$\mathbb{E}[\tanh(u_t)] = -\frac{\mu_t^\top \mathbb{E}[X_t]}{\|\mu_t\|^2}$. Now looking at the EM algorithm and in particular M-step, we see that $w^+ = \mathbb{E}[r_1(X_t)] = \mathbb{E}[\frac{1}{2}(1+\tanh(u_t))]$, hence, plugging the convergence point of score loss into EM algorithm, $w^+ = \frac{1}{2}(1 - \frac{\mu_t^\top \mathbb{E}[X_t]}{\|\mu_t\|^2})$, in other words EM algorithm is also converged, equivalently, $\mathbb{E}[X_t] = \mu_t(2w-1)$ which is exactly the first order moment matching condition. In other words, convergence of score estimation loss corresponds EM algorithm convergence that satisfies the first order moment equality, since the means and covariances are known this corresponds to the global maximum of the likelihood. At the convergence point we can measure the mean squared error of estimating the mixing weight from samples. In particular, let us define $\hat{w} = \frac{1}{2}(1 - \frac{\mu_t^\top \hat{X}_t}{\|\mu_t\|^2})$ as the sample mixing weight, where $\hat{X}_t$ is the sample mean at time t. Then the L_2 error can be found to be $\mathbb{E}(w-\hat{w})^2 \leq \frac{w(1-w)}{n} + \frac{d}{4\|\mu_t\|^2 n}$.

Remark 1. The bound on mixing-weight error is dimension-free because both its numerator and denominator scale with d, unlike mean estimation, whose difficulty grows in high dimensions. Thus clients with few samples can reliably learn their own weights. The bound has two components: (i) sampling term—the variance of cluster assignments; it shrinks when one mixture component dominates, (ii) separation term—inversely proportional to $\|\mu_t\|^2$; larger mean separation improves accuracy, while small separation can be offset by more data. Crucially, the bound depends only on the norm $\|\mu_t\|$—not on how accurately the mean is estimated—because correct assignment, not mean precision, drives weight estimation. Overall generation quality will, however, reflect errors in both $\hat{\mu}$ and$\hat{w}$. With the mixing-weight bound in hand, we next assess score error: we measure the mean-squared gap between the true score and that produced by the converged estimators, where the global backbone learns $\hat{\mu}_t$ via DDPM gradient descent Shah et al. (2023) and each client refines its own $\hat{w}$ using our personalized procedure.

Theorem 4.3. *Let $L_{est}(\hat{\mu}, \hat{w})$ be the score matching loss between the true score function and estimated score function where the $\hat{\mu}_t$ is estimated globally using the procedure in Shah et al. (2023), and mixing weights are estimated individually by doing gradient steps in the personalized DDPM loss. Assuming B is a constant such that $\|\mu_t\|^2 \leq B$. Then, the loss has the following error,*

$$L_{est}(\hat{\mu}, \hat{w}) = \mathcal{O}\Big(\frac{d^6 B^8}{mn(1-e^{-2t})^2}\Big) + \mathcal{O}\Big(\frac{1}{n}\Big) \quad (7)$$

Proof Outline. We start by utilizing Lipschitz continuity of tanh and log functions, and some algebraic manipulations yield, $L_{est}(\hat{\mu}, \hat{w}) \leq 4B(d + (4w(1-w)+1)B)\mathbb{E}\|\mu - \hat{\mu}\|^2 + \frac{1}{2w_{min}^2(1-w_{min})^2}\frac{w(1-w)}{n}$. Converting the high probability bound in Shah et al. (2023) to an expectation bound we have, $\mathbb{E}\|\mu_t - \hat{\mu}_t\|^2 \leq C\frac{d^5 B^6}{mn\beta_t^2}$, for constant $C > 0$. Combining with earlier bound concludes the proof outline.

Remark 2. Theorem 4.3 shows that score-estimation error is governed by errors in both the mean and the mixing weight. Except when the client count reaches $\Theta(d^6 B^8)$, the mean-estimation term dominates. Because mixing-weight error is dimension-free and small even with limited data, a client can accurately personalize once a good global mean is pretrained. This shows the benefit of our split.

5 EXPERIMENTS

Baselines. As diffusion models require very large U-Net models to be trained, personalized FL methods

with two different models, i.e. separate individual and global models, are not feasible. Hence, we compare our method to single model methods. One such baseline is the simple *FedAvg+fine-tuning* for personalization, though it is a simple approach it is accepted as one of the competitive personalization methods. Another baseline is *meta-learning* approaches, specifically Fallah et al. (2020), where the global model is minimizing a meta-learning based loss that allows for better personalization. Lastly, inspired by Collins et al. (2021), we introduce a *shared representation* approach, as a competing solution to compare with, where downsample and upsample blocks are trained globally and middle block is trained individually. The goal of this such partition is to enable global learning of feature extraction (downsample blocks) and feature generation (upsample blocks) similar to global feature extraction in Collins et al. (2021); however, in classifier models Collins et al. (2021) trains individual classifier heads which is not directly applicable to U-Net models. Inspired by that, we enable individual learning of middle block to introduce personalization in the latent space. We provide more details on implementation details of baselines in Appendix C.

Datasets. We randomly generate a synthetic dataset (using the true underlying score function) for synthetic experiments to estimate score function (mean and mixing weight) of GMM. For real datasets, we use MNIST, CIFAR-10/100, and CelebA (celebrity faces) with increasing complexity. The images have 28x28, 32x32, and 64x64 resolutions respectively.

For pretraining on MNIST, there are 30 clients and each client has access to 400 samples from a randomly selected class; for fine-tuning on new clients, there are 30 clients each client has access to 400 samples from 3 different classes which simulates another layer of challenge, namely the new clients have samples from a different mixture dataset compared to the pretraining clients. On CIFAR-10, there are 18 clients each with 2750 samples from a randomly selected class. The new clients have samples from two randomly selected classes, but the amount of data at each client drops to 275 to simulate a data scarcity setting. On CelebA dataset there are 24 clients each with data coming from 5 different individuals, the total amount of data per client is only around 25. This setting is very challenging and acts as a stress test for our method. The new clients have data from different individuals compared to pre-training data.

Models. We use a conditional U-Net architecture where client id's, that are converted to vector embeddings, are used as the conditioning information. We use 14 layer, 6M parameter architecture for MNIST experiments, and 18 layer 13M parameter for CIFAR-10, CelebA experiments. More details can be found in Appendix. We use AdamW Loshchilov and Hutter (2019) optimizer and provide more details on hyperparameters in Appendix C.

5.1 Numerical Results

Collaborative pre-training. The main image quality metric we use is Kernel Inception Distance (KID) Bińkowski et al. (2018), as FID Heusel et al. (2017) is not usable for smaller datasets due to biasedness. Table 1 reports KID after the collaborative pre-training phase. On MNIST and CelebA our `SPIRE` achieves the lowest KID, improving over baselines. Particularly, the gap is significant on CelebA dataset. On CIFAR-10 `SPIRE` matches *FedAvg+FT* while the *Shared-Rep.* variant edges out both methods by a small margin. We attribute the CIFAR-10 result to the fact that all clients observe only a single class during pretraining—global feature extractors alone already generalize reasonably well in that setting. Across all three data sets the meta-learning baseline struggles, confirming the practical difficulty of applying first-order MAML–style algorithms to large diffusion models. We attribute this to the fact that MAML's low-order Taylor approximation of client objectives is insufficient to capture the complexity of heterogeneous DDPM losses, particularly in early training where general features are still being learned.

Generalization to new clients. The advantage of `SPIRE` becomes far more pronounced once entirely new clients—holding data from unseen class mixtures or unseen identities—join the federation (Table 1,b). After less than 10 epochs of local fine-tuning `SPIRE` yields large gains over the next best method Shared representation. Neither *FedAvg+FT* nor meta-learning can close the gap; in fact, both are outperformed by `SPIRE` on every data set. These results support our central claim: *embedding the client identifier as a learnable conditioning signal equips a* single *diffusion model with enough capacity to adapt to novel client distributions.*

Qualitative analysis. Figures 2 and 5 (in Appendix C) visualize samples drawn from models trained on CelebA and MNIST, respectively, for a particular client. For MNIST, the client has access to images labeled as '5','6', and '9'. For the most of the images `SPIRE` is able to generate 5,6,9 images with occasional hallucination from other classes. MAML approach fails to generate quality images. Shared representation approach often misses the structure of the numbers, and FedAvg+FT does not generate diverse images implying an overfitting to the data. For CelebA , all methods struggle with the coloring of images due to the challenge in pretraining task. Images generated by `SPIRE` exhibit noticeably sharper facial

Table 1: Pre-training performance for participating clients, and fine-tuning performance for newly joined client performance KID↓ values of respective methods on respective datasets.

Method	Pre-training			New client fine-tuning		
	MNIST	CIFAR-10	CelebA	MNIST	CIFAR-10	CelebA
FedAvg+fine-tuning	0.015	0.034	0.051	0.028	0.101	0.075
SPIRE	**0.010**	0.034	**0.046**	**0.012**	**0.038**	**0.061**
Shared Representation	**0.010**	**0.031**	0.056	0.017	0.048	0.094
Meta learning	0.061	0.073	0.081	0.079	0.093	0.120

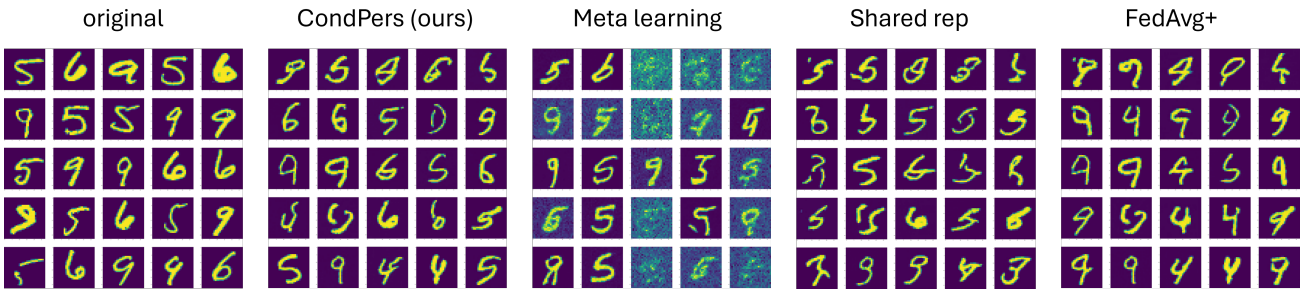


Figure 2: Grid of original MNIST images and generated images by competing methods for particular client, who has sampled data from classes '5,6,9; using the same random seed. Our method results in less hallucinations, more diversity while preserving structure.

details. Competing methods over-fit, producing more washed-out or saturated outputs. The faces generated by our method are more diverse and have better visual features e.g. shadowing.

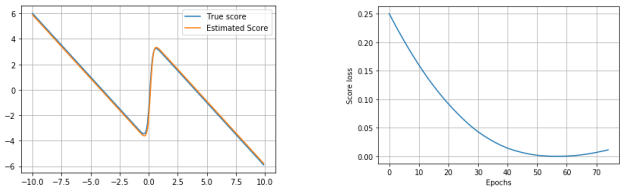


Figure 3: Estimated score function (left) and score loss convergence (right) for the synthetic experiments. The true mixing weight and mean are 0.7 and 4 respectively, the estimated are 0.72 and 4.11.

Synthetic experiments for GMMs. For synthetic experiments, we are verifying correctness of our found score function form in 4.1. Given the objective in (6), we use Adam algorithm to update both the mean and mixing weights. The results in Figure 3 indicate convergence of training and success of estimating the score through the DDPM objective. See Appendix for experimental details.

Conditional personalization is robust to catastrophic forgetting. A major advantage of our method, in new client fine-tuning, compared to competing methods is its robustness to overfitting, i.e. catastrophic forgetting. Since our method relies on

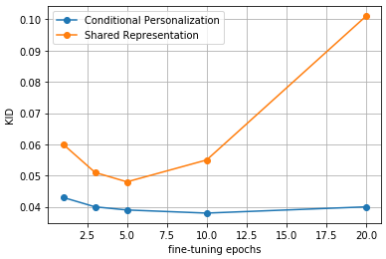


Figure 4: The number of local epochs need to be carefully fine-tuned for new clients using shared representation approach due to overfitting and underfitting risk, on the other hand conditional personalization performs better and is more robust with respect to number of local epochs (CIFAR-10).

updating only a small part of the model, previously learned information through the backbone has no risk of getting lost during the fine-tuning stage. This prevents an oversensitivity towards number of fine-tuning epochs or learning rate; which are mainly responsible for a potential forgetting. To display this notion, we compare to shared representation approach and illustrate the resulting KID values for new clients under different finetuning epochs Figure 4.

Overall. In the experiments we showed the efficacy of our proposed way of personalization. SPIRE is especially superior to other methods for new client fine-tuning which is a very crucial problem in FL and, in large scale ML. Due to the natural PEFT implied by

our method, it is robust to choices such as learning rate and number of iterations. It does not carry risks such as catastrophic forgetting.

6 CONCLUSION

We introduced SPIRE, the first framework that treats personalized diffusion in federated learning as a *conditional generation* problem. By decomposing the model into a global backbone and lightweight client embeddings, SPIRE enables parameter-efficient, communication-free adaptation on devices that hold only a handful of samples. Our theory establishes a link between gradient descent on the DDPM objective and Gaussian-mixture models, yielding dimension-free error bounds for the client-specific parameters. Empirically, SPIRE consistently matches or surpasses strong baselines during collaborative pre-training and delivers superior performance for new-client personalization, resulting quantitatively and qualitatively better generation. Future directions include (i) extending the embedding-based conditioning to multi-modal and text-to-image diffusion, (ii) exploring hierarchical embeddings that capture groups of clients and (iii) integrating SPIRE into large pre-trained diffusion models such as StableDiffusion Rombach et al. (2022).

Acknowledgments

The research was supported in part by the DEVCOM Army Research Laboratory under award # W911NF1720196, and by the National Science Foundation under award #2502536. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the funding agencies.

References

Durmus Alp Emre Acar, Yue Zhao, Ruizhao Zhu, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Debiasing model updates for improving personalized federated training. In *International Conference on Machine Learning*, pages 21–31. PMLR, 2021.

Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1lUOzWCW.

Huili Chen, Jie Ding, Eric William Tramel, Shuang Wu, Anit Kumar Sahu, Salman Avestimehr, and Tao Zhang. Self-aware personalized federated learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=EqJ5_hZSqgy.

Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=zyLVMgsZ0U_.

Sitan Chen, Vasilis Kontonis, and Kulin Shah. Learning general gaussian mixtures with efficient score matching, 2024. URL https://arxiv.org/abs/2404.18893.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In Marina Meila and Tong Zhang, editors, *International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 2089–2099. PMLR, 2021.

Constantinos Daskalakis, Christos Tzamos, and Manolis Zampetakis. Ten steps of em suffice for mixtures of two gaussians. In *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pages 704–710, 2017.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=AAWuCvzaVt.

Canh T. Dinh, Nguyen H. Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. In *Advances in Neural Information Processing Systems*, 2020.

Simon Shaolei Du, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. Few-shot learning via learning the representation, provably. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=pW2Q2xLwIMD.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. In *Advances in Neural Information Processing Systems*, 2020.

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. In *Advances in Neural Information Processing Systems*, 2020.

Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Mikhail Khodak, Maria-Florina F Balcan, and Ameet S Talwalkar. Adaptive gradient-based meta-learning methods. In *Advances in Neural Information Processing Systems*, 2019.

Nikita Yurevich Kotelevskii, Maxime Vono, Alain Durmus, and Eric Moulines. Fedpop: A bayesian approach for personalised federated learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=KETwimTQexH.

Jeongyeol Kwon and Constantine Caramanis. The em algorithm gives sample-optimality for learning mixtures of well-separated gaussians, 2020. URL https://arxiv.org/abs/2002.00329.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems 2020, MLSys*, 2020.

Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. In *Advances in Neural Information Processing Systems*, 2020.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34, 2021.

Kaan Ozkara, Navjot Singh, Deepesh Data, and Suhas Diggavi. Quped: Quantized personalization via distillation with applications to federated learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Kaan Ozkara, Antonious Girgis, Deepesh Data, and Suhas Diggavi. A statistical framework for personalized federated learning and estimation: Theory, algorithms, and privacy. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FUiDMCr_W4o.

Kaan Ozkara, Bruce Huang, Ruida Zhou, and Suhas Diggavi. ADEPT: Hierarchical bayes approach to personalized federated unsupervised learning. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025. URL https://openreview.net/forum?id=z9RDCfRZHw.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. URL https://arxiv.org/abs/2112.10752.

Kulin Shah, Sitan Chen, and Adam Klivans. Learning mixtures of gaussians using the DDPM objective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=aig7sgdRfI.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S. Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, pages 266–282. Springer, 2020.

Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized collaborative learning of personalized models over networks. In *Artificial Intelligence and Statistics*, pages 509–517. PMLR, 2017.

Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. Fully decentralized joint learning of personalized models and collaboration graphs. In *International Conference on Artificial Intelligence and Statistics*, pages 864–874. PMLR, 2020.

Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M. Alvarez. Personalized federated learning with first order model optimization. In *International Conference on Learning Representations*, 2021.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [NA]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Appendix

A Related Work, Limitations and Broader Impact

Related Works. Beyond the studies highlighted in Section 1, the literature on Personalized Federated Learning (FL) has evolved along several complementary tracks. Meta-learning approaches aim to learn initialization points that quickly adapt to each client's data (Fallah et al., 2020; Acar et al., 2021; Khodak et al., 2019). Regularization-based methods encourage personalization by adding client-specific penalties to the global objective (Deng et al., 2020; Mansour et al., 2020; Hanzely and Richtárik, 2020). Clustered FL partitions clients into groups with similar data distributions (Zhang et al., 2021; Mansour et al., 2020; Ghosh et al., 2020; Marfoq et al., 2021), while knowledge-distillation techniques transfer information between local and global models (Li et al., 2020; Ozkara et al., 2021). Multi-task formulations treat each client as a related task to optimize jointly (Dinh et al., 2020; Smith et al., 2017; Vanhaesebrouck et al., 2017; Zantedeschi et al., 2020), and common-representation methods share feature extractors across clients (Du et al., 2021; Tian et al., 2020; Collins et al., 2021). A hierarchical Bayesian viewpoint has recently inspired new supervised FL algorithms with adaptation capabilities (Ozkara et al., 2023; Chen et al., 2022; Kotelevskii et al., 2022).

Privacy Discussion. During pre-training, `SPIRE` has the same privacy surface as standard FedAvg: only the shared backbone is communicated, so any differential privacy (DP) or secure aggregation mechanism from the FL literature applies directly to backbone updates. During new-client fine-tuning, each client updates only its local embedding without server communication, incurring zero additional privacy risk. An interesting future direction is whether `SPIRE` architectural decoupling—where client-specific information resides in local embeddings rather than shared parameters—may offer additional robustness to DP noise compared to methods that encode all personalization in the shared model.

B Proofs and other details for theoretical results

B.1 Diffusion model setup

We first start with the forward process. Ornstein-Uhlenbeck (OU) forward diffusion process describes the gradual transformation from a true data sample to noise through a SDE, it is defined as

$$\mathrm{d}X_t = -\delta_t X_t \,\mathrm{d}t + \sqrt{2\delta_t}\,\mathrm{d}W_t, \quad X_0 \sim q_0(x), \tag{8}$$

here $q_0(x)$ denotes the true underlying distribution, similarly $q_t(x)$ will be used to denote a latent distribution, δ_t is a drift coefficient, commonly assumed a constant e.g. 1. W_t is the standard Brownian motion. $X_t \sim q_t$ is a latent variable. Equivalently, it can be shown for some $\beta_t : \lim_{t\to\infty} \beta_t = 1, \beta_0 = 0$,

$$X_t = \sqrt{1-\beta_t^2} X_0 + \beta_t Z_t, \text{with } X_0 \sim q_0 \text{ and } Z_t \sim \mathcal{N}(0, \mathrm{I}).$$

Corresponding to the forward process, is the backward SDE process, which is defined to formalize recovering a data sample from true distribution, given a corrupted sample and score function,

$$\mathrm{d}\overleftarrow{X}_t = \left[\delta_{T-t}\overleftarrow{X}_t + 2\delta_{T-t}\nabla \log q_{T-t}(\overleftarrow{X}_t)\right]\mathrm{d}t + \sqrt{2\delta_{T-t}}\,\mathrm{d}W_t, \quad \overleftarrow{X}_0 \sim q_T(\cdot).$$

In the backward process, we use $\overleftarrow{X}_0$ to denote a sample from pure noise q_T. $\nabla \log q_{T-t}(\overleftarrow{X}_t)$ is the true score function at time $T-t$ (or at time t in terms of the forward process). Of course the true score function is unknown, hence we need to estimate it from the data using a fixed pure noise distribution. To that end, one constructs the score matching optimization criterion,

$$\min_\theta \int_\tau^T \mathbb{E}_{X_t \sim q_t(\cdot)} \left[\left\| s_\theta(x_t, t) - \nabla \log q_t(X_0) \right\|^2\right] \mathrm{d}t, \tag{9}$$

where θ is a neural network that is assumed to parameterize the score function. Equivalent to score matching is the DDPM view, which yields,

$$\min_\theta \int_\tau^T \mathbb{E}_{X_0 \sim q_0(\cdot), Z_t} \left[\left\| s_\theta(X_t, t) + \frac{Z_t}{\beta_t} \right\|^2\right] \mathrm{d}t. \tag{10}$$

EM updates. EM algorithm is the most common approach to find parameters of GMMs. E-step is consists of finding the soft assignments or so called responsibilities, and M-step corresponds to updating the parameters. As can be seen in Kwon and Caramanis (2020):

$$\text{(E-step)}: \quad r_i(X) = \frac{w_i \exp\big(-\|X-\mu_i\|^2/2\big)}{\sum_{j=1}^k w_j \exp(-\|X-\mu_j\|^2/2)}, \text{ where subscripts denote cluster assignments}$$

$$\text{in our case:} \quad r_1(X) = \frac{w \exp\big(-\|X-\mu\|^2/2\big)}{w \exp(-\|X-\mu\|^2/2) + (1-w)\exp(-\|X+\mu\|^2/2)}, r_2(X) = 1 - r_1(X)$$

$$\text{(M-step)}: \quad w^+ = \mathbb{E}_X[r_1(X)], \quad \mu^+ = \mathbb{E}_X[r_1(X)X]/\mathbb{E}_X[r_1(X)] \text{ where } + \text{ denotes the next iteration.}$$

B.2 Proofs in Section 4

Lemma B.1 (Solution of the OU forward SDE)**.** *Let $(W_t)_{t\geq 0}$ be a standard d-dimensional Brownian motion independent of the initialization $X_0 \sim q_0$. Consider the Ornstein–Uhlenbeck (OU) forward (noising) process*

$$dX_t \;=\; -X_t\,dt \;+\; \sqrt{2}\,dW_t, \qquad X_0 \sim q_0. \tag{11}$$

Then for every $t \geq 0$ the marginal distribution of X_t is

$$X_t \;=\; \mathrm{e}^{-t}\,X_0 \;+\; \sqrt{1-\mathrm{e}^{-2t}}\;Z_t, \qquad Z_t \sim \mathcal{N}\big(0,\mathrm{I}\big), \tag{12}$$

with Z_t independent of X_0.

Proof. Multiply (11) by the integrating factor e^t:

$$\mathrm{e}^t\,dX_t + \mathrm{e}^t\,X_t\,dt \;=\; \sqrt{2}\,\mathrm{e}^t\,dW_t \;\Longrightarrow\; d\big(\mathrm{e}^t X_t\big) = \sqrt{2}\,\mathrm{e}^t\,dW_t.$$

Integrating from 0 to t and dividing by e^t yields

$$X_t \;=\; \mathrm{e}^{-t}X_0 \;+\; \sqrt{2}\,\mathrm{e}^{-t}\int_0^t \mathrm{e}^s\,dW_s.$$

Because the Itô integral is Gaussian with mean 0, its covariance is

$$\mathrm{Cov}\Big(\sqrt{2}\,\mathrm{e}^{-t}\int_0^t \mathrm{e}^s\,dW_s\Big) = 2\,\mathrm{e}^{-2t}\int_0^t \mathrm{e}^{2s}\,ds = 1-\mathrm{e}^{-2t}.$$

Thus the stochastic integral can be written as $\sqrt{1-\mathrm{e}^{-2t}}\;Z_t$ with $Z_t \sim \mathcal{N}(0,\mathrm{I})$, independent of X_0. Substituting back gives (12), completing the proof. □

Lemma B.2 (Two–component mixture is preserved by the OU forward process)**.** *Let the initial distribution be the symmetric two–component Gaussian mixture*

$$q_0(x) = w\,\mathcal{N}\big(x;\mu,\mathbf{I}\big) \;+\; (1-w)\,\mathcal{N}\big(x;-\mu,\mathbf{I}\big), \qquad 0<w<1,\;\; \mu\in\mathbb{R}^d.$$

Consider the Ornstein–Uhlenbeck (OU) forward SDE

$$X_t \;=\; e^{-t}X_0 \;+\; \sqrt{1-e^{-2t}}\;Z_t, \qquad \textit{with } Z_t \sim \mathcal{N}(0,\mathbf{I}) \textit{ independent of } X_0.$$

Then, for every $t \geq 0$, the law of X_t remains a two–component mixture

$$q_t(x) = w\,\mathcal{N}\big(x;\mu_t,\mathbf{I}\big) \;+\; (1-w)\,\mathcal{N}\big(x;-\mu_t,\mathbf{I}\big), \qquad \textit{where } \mu_t \;:=\; e^{-t}\mu.$$

Proof. Condition on the component of the mixture that generated X_0.

Component "$+\mu$". If $X_0 \sim \mathcal{N}(\mu, \mathbf{I})$, then

$$X_t \;=\; e^{-t}X_0 + \sqrt{1-e^{-2t}}\,Z_t \;\sim\; \mathcal{N}\big(e^{-t}\mu,\; e^{-2t}\mathbf{I} + (1-e^{-2t})\mathbf{I}\big) \;=\; \mathcal{N}\big(e^{-t}\mu, \mathbf{I}\big).$$

Component "$-\mu$". Analogously, if $X_0 \sim \mathcal{N}(-\mu, \mathbf{I})$, we obtain

$$X_t \;\sim\; \mathcal{N}\big(-e^{-t}\mu, \mathbf{I}\big).$$

Mixture. Because the two cases occur with probabilities w and $1-w$, the unconditional distribution of X_t is the weighted sum of the two Gaussians derived above, namely

$$q_t(x) = w\,\mathcal{N}\big(x; e^{-t}\mu, \mathbf{I}\big) + (1-w)\,\mathcal{N}\big(x; -e^{-t}\mu, \mathbf{I}\big),$$

which coincides with the claimed form after defining $\mu_t = e^{-t}\mu$. □

Proof of Lemma 4.1. Recall that q_t is the symmetric two–component mixture

$$q_t(x) = w\,\mathcal{N}\big(x; \mu_t, \mathrm{I}\big) + (1-w)\,\mathcal{N}\big(x; -\mu_t, \mathrm{I}\big), \qquad 0 < w < 1, \quad \mu_t = e^{-t}\mu.$$

Because $q_t(x) = \sum_{k=1}^{2} \pi_k p_k(x)$ with $p_1(x) = \mathcal{N}(x; \mu_t, \mathrm{I}),\;\; p_2(x) = \mathcal{N}(x; -\mu_t, \mathrm{I})$ and weights $\pi_1 = w,\; \pi_2 = 1-w$, the score can be written as

$$\nabla_x \log q_t(x) = \frac{\displaystyle\sum_{k=1}^{2} \pi_k\, p_k(x)\, \nabla_x \log p_k(x)}{\displaystyle\sum_{k=1}^{2} \pi_k p_k(x)}.$$

For a Gaussian with unit covariance, $\nabla_x \log \mathcal{N}(x; m, \mathrm{I}) = -(x-m)$. Hence

$$\nabla_x \log q_t(x) = -\Big[r_1(x)\big(x-\mu_t\big) + r_2(x)\big(x+\mu_t\big)\Big],$$

where

$$r_1(x) \;=\; \frac{w\,p_1(x)}{w\,p_1(x) + (1-w)p_2(x)}, \qquad r_2(x) = 1 - r_1(x)$$

are the posterior ("belief") probabilities of the two components. A short algebraic simplification gives

$$\nabla_x \log q_t(x) \;=\; \big(2r_1(x) - 1\big)\,\mu_t \;-\; x. \tag{13}$$

Writing the Gaussian densities explicitly:

$$p_1(x) = \frac{1}{(2\pi)^{d/2}} \exp\Big[-\tfrac{1}{2}\|x-\mu_t\|^2\Big], \quad p_2(x) = \frac{1}{(2\pi)^{d/2}} \exp\Big[-\tfrac{1}{2}\|x+\mu_t\|^2\Big].$$

Then

$$\frac{r_1(x)}{1-r_1(x)} = \frac{w\,p_1(x)}{(1-w)\,p_2(x)} = \frac{w}{1-w}\,\exp\Big[-\tfrac{1}{2}(\|x-\mu_t\|^2 - \|x+\mu_t\|^2)\Big].$$

Because $\|x-\mu_t\|^2 - \|x+\mu_t\|^2 = -4\,\mu_t^\top x$, we obtain

$$\frac{r_1(x)}{1-r_1(x)} = \exp\Big[2\mu_t^\top x \;+\; \log\frac{w}{1-w}\Big].$$

Setting $z(x) := 2\mu_t^\top x \;+\; \log\frac{w}{1-w}$ and recalling that $r_1 = \sigma(z)$ with the logistic function $\sigma(z) = 1/(1+e^{-z})$, we have $2r_1 - 1 = \tanh\big(\frac{1}{2}z\big)$. Therefore

$$2r_1(x) - 1 \;=\; \tanh\Big(\mu_t^\top x + \tfrac{1}{2}\log\frac{w}{1-w}\Big) \;=\; \tanh\Big(\mu_t^\top x - \tfrac{1}{2}\log\frac{1-w}{w}\Big).$$

Substituting the expression for $2r_1(x) - 1$ back into (13) yields

$$\nabla_x \log q_t(x) = \tanh\Big(\mu_t^\top x - \tfrac{1}{2}\log\frac{w}{1-w}\Big)\mu_t \;-\; x,$$

which is exactly the claimed form in Lemma 4.1. □

B.3 Algorithm for new client fine-tuning

Algorithm 2: Conditional personalized fine-tuning for new clients

Input: Number of iterations K, learning rate (η)

1: **Download** backbone model, initialize the embedding $e_{j,0}$
2: Set $\theta_{j,0} = [\theta_{bb}, e_{j,0}]$
3: **for** $k = 1$ **to** K **do**
4: Sample a batch of samples $\{X_i\}_{i=1}^b$ and diffusion timesteps $\{t_i\}_{i=1}^b$
5: Use forward process (1) to obtain a batch of noisy samples $\{X_{t_i}\}_{i=1}^b$
6: Take gradient step for embedding $\boldsymbol{e}_{j,k+1} = \boldsymbol{e}_{j,k} - \nabla_{\boldsymbol{e}_j} L_{ddpm}^{(j)}(\{X_{t_i}\}_{i=1}^b)$
7: **end for**

Algorithm 2 A new client downloads the current backbone (line 1), forms $\boldsymbol{\theta}_{j,0}$ by appending a freshly initialized embedding (line 2), and for K iterations (line 3) repeats: sampling data and timesteps (lines 4–5) and updating *only* its embedding via the DDPM loss (line 6). No further communication is required after the initial download, so personalization remains entirely on-device.

B.4 Proofs of Theorems 4.2, 4.3

Proof of Theorem 4.2. We start by taking the derivative of (6) with respect to w at a particular timepoint t. Let $u_t = \mu_t^\top X_t - \frac{1}{2}\log(w/(1-w))$, $f(X_t) = \tanh(u_t) - X_t + \frac{Z_t}{\sqrt{1-e^{-2t}}}$, by the chain rule,

$$
\begin{aligned}
\frac{\mathrm{d}L}{\mathrm{d}w} &= \frac{\mathrm{d}L}{\mathrm{d}f(X_t)} \cdot \frac{\mathrm{d}f(X_t)}{\mathrm{d}\tanh(u_t)} \cdot \frac{\mathrm{d}\tanh(u_t)}{\mathrm{d}u_t} \cdot \frac{\mathrm{d}u_t}{\mathrm{d}(\frac{1}{2}\log((1-w)/w))} \cdot \frac{\mathrm{d}(\frac{1}{2}\log((1-w)/w))}{w} \\
&= \mathbb{E}\left[2\mu_t^\top(\tanh(u_t)\mu_t - X_t + Z_t/(\sqrt{1-e^{-2t}})\tanh'(u_t)\frac{-1}{2w(1-w)}\right] \\
&= \frac{-1}{w(1-w)}\mathbb{E}\left[\mu_t^\top(\tanh(u_t)\mu_t - X_t + Z_t/\sqrt{1-e^{-2t}}) \cdot \tanh'(u_t)\right] \\
&\frac{-1}{w(1-w)}\mathbb{E}\left[\mu_t^\top(\tanh(u_t)\mu_t - X_t + Z_t/\sqrt{1-e^{-2t}}) \cdot \tanh'(u_t)\right] \\
&\frac{-1}{w(1-w)}\mathbb{E}\left[\|\mu_t\|^2\tanh(u_t)\tanh'(u_t) - \mu_t^\top X_t\tanh'(u_t) + \mu_t^\top Z_t/\sqrt{1-e^{-2t}}\tanh'(u_t)\right] \\
&\overset{(a)}{=} \frac{-1}{w(1-w)}\mathbb{E}\left[\|\mu_t\|^2\tanh(u_t)\tanh'(u_t) - \mu_t^\top X_t\tanh'(u_t) + \|\mu_t\|^2\tanh''(u_t)\right] \\
&\overset{(b)}{=} \frac{-1}{w(1-w)}\mathbb{E}\left[-\|\mu_t\|^2\tanh(u_t)\tanh'(u_t) - \mu_t^\top X_t\tanh'(u_t)\right] \\
&= \frac{1}{w(1-w)}\mathbb{E}\left[\tanh'(u_t)(\|\mu_t\|^2\tanh(u_t) + \mu_t^\top X_t)\right]
\end{aligned}
$$

Hence, at the critical point we have the following condition being satisfied,

$$
\mathbb{E}[\tanh(u_t)] = -\frac{\mu_t^\top \mathbb{E}[X_t]}{\|\mu_t\|^2}. \tag{14}
$$

Now looking at the EM algorithm and in particular M-step, we see that

$$
w^+ = \mathbb{E}[r_1(X_t)] = \mathbb{E}[\frac{1}{2}(1+\tanh(u_t))]
$$

hence, plugging the convergence point of score loss into EM algorithm,

$$
w^+ = \frac{1}{2}(1 - \frac{\mu_t^\top \mathbb{E}[X_t]}{\|\mu_t\|^2}), \text{ in other words EM algorithm is also converged, equivalently,}
$$

$$\mathbb{E}[X_t] = \mu_t(2w-1), \text{ which is exactly the first order moment matching condition.}$$

In other words, convergence of score estimation loss corresponds EM algorithm convergence that satisfies the first order moment equality, since the means and covariances are known this corresponds to the global maximum of the likelihood.

At the convergence point we can measure the mean squared error of estimating the mixing weight from samples. In particular, let us define $\hat{w} = \frac{1}{2}(1 - \frac{\mu_t^\top \hat{X}_t}{\|\mu_t\|^2})$ as the sample mixing weight, where $\hat{X}_t$ is the sample mean at time t. Then the L_2 error can be found to be:

$$\begin{aligned}
\mathbb{E}\|w - \hat{w}\|^2 &= \frac{1}{4}\|\frac{\mu_t^\top}{\|\mu_t\|^2}(\mathbb{E}[X_t] - \hat{X}_t)\|^2 \\
&\leq \frac{1}{4\|\mu_t\|^2}\|\mathbb{E}[X_t] - \hat{X}_t\|^2 \\
&\leq \frac{1}{4\|\mu_t\|^2}\frac{tr(\Sigma)}{n}, \text{ where } \Sigma \text{ is true overall covariance vector} \\
&= \frac{1}{4\|\mu_t\|^2}\frac{d + 4w(1-w)\|\mu_t\|^2}{n} \\
&= \frac{w(1-w)}{n} + \frac{d}{4\|\mu_t\|^2 n},
\end{aligned}$$

which concludes the proof. □

Proof of Theorem 4.3. Let us define $u := X_t^\top \mu_t - \frac{1}{2}\log((1-w)/(w))$, and similarly for $\hat{u}$ with estimated parameters, and $b(w) = \frac{1}{2}\log((1-w)/w)$, we also assume there is a constant such that $\|\mu_t\|^2 \leq B$. Then we have,

$$\begin{aligned}
L_{est}(\hat{\mu}, \hat{w}) &= \mathbb{E}_{X_o \sim \mathcal{D}_i, Z_t}\Big\| \tanh\Big(X_t^\top \mu_t - \frac{1}{2}\log((1-w)/(w))\Big)\mu_t \\
&\quad - \tanh\Big(X_t^\top \hat{\mu}_t - \frac{1}{2}\log((1-\hat{w})/(\hat{w}))\Big)\hat{\mu}_t\Big\|^2 \\
&= \mathbb{E}\| \tanh(u)\mu_t - \tanh(\hat{u})\mu_t + \tanh(\hat{u})\mu_t - \tanh(\hat{u})\hat{\mu}_t\|^2, \\
&\leq \|\mu_t\|^2(\tanh u - \tanh \hat{u})^2 + |\tanh \hat{u}|\|\mu_t - \hat{\mu}_t\|^2 \\
&\leq \mathbb{E}[\|\mu_t\|^2 2(u - \hat{u})^2 + \|\mu_t - \hat{\mu}_t\|^2] \\
&= \mathbb{E}[\|\mu_t\|^2 2(X_t^\top(\mu_t - \hat{\mu}_t) + b(w) - b(\hat{w}))^2 + \|\mu_t - \hat{\mu}_t\|^2], \\
&\leq 4\mathbb{E}[\|\mu_t\|^2\|X_t\|^2\|\mu_t - \hat{\mu}_t\|^2 + \frac{(w-\hat{w})^2}{2w_{min}^2(1-w_{min})^2} + \|\mu_t - \hat{\mu}_t\|^2] \\
&\leq 4B\mathbb{E}\|X_t\|^2\mathbb{E}\|\hat{\mu}_t - \mu_t\|^2 + \mathbb{E}(w-\hat{w})^2 + \mathbb{E}\|\mu_t - \hat{\mu}_t\|^2 \\
&\leq 4B(d + (4w(1-w)+1)B)\mathbb{E}\|\mu - \hat{\mu}\|^2 + \frac{1}{2w_{min}^2(1-w_{min})^2}\frac{w(1-w)}{n},
\end{aligned}$$

From Shah et al. (2023), at the convergence point of their algorithm, we have that with probability at least 1-$2d\exp\Big(-\frac{n\epsilon^2\beta_t^2}{d^4B^6}\Big)$ the convergence is to a ball with $\|\mu_t - \hat{\mu}_t\|^2 = \mathcal{O}(\epsilon)$. Now we convert this high probability bound to a expectation bound. In particular the non-vanishing ϵ term is introduced in Lemma E.7 of Shah et al. (2023). There we have the following high probability bound for bounding the difference between population and empirical gradient,

$$Pr[\Big\|\nabla_{\hat{\mu}_t}\frac{1}{n}\sum_{j=1}^{n} L_t(s_{\hat{\mu}_t}(x_{j,t})) - L_t(s_{\hat{\mu}_t})\Big\| > \epsilon] \leq 2d\exp\Big(-\frac{n\epsilon^2\beta_t^2}{d^2B^6}\Big)$$

equivalently for some constant $C > 0$, at the end of convergence of algorithm (where exponential decaying contraction term has vanished), the result in Shah et al. (2023) yields,

$$Pr[\|\mu_t - \hat{\mu}_t\| > C\epsilon] \leq 2d \exp\Big(- \frac{n\epsilon^2 \beta_t^2}{d^4 B^6}\Big)$$

For conversion, we use the tail definition of expectation, $\mathbb{E}[Y^2] = \int_0^\infty P(Y^2 > y)dy = \int_0^\infty P(Y > \sqrt{y})dy$ which yields,

$$\mathbb{E}\|\mu_t - \hat{\mu}_t\|^2 = \int_0^\infty 2Cd \exp\Big(- \frac{ny\beta_t^2}{d^4 B^6}\Big)dy \leq C\frac{d^5 B^6}{mn\beta_t^2},$$

where we used the fact that $\int_0^\infty \exp(-Cx)dx = \frac{1}{C}$ As a result, the overall bound is,

$$L(\hat{\mu}, \hat{w}) = \mathcal{O}\Big(\frac{d^6 B^8}{mn(1-e^{-2t})^2}\Big) + \mathcal{O}\Big(\frac{1}{n}\Big) \tag{15}$$

where the first term is due to global estimation of mean and the second term is due to the local estimation of mixing weight.

□

C Additional Details and Results on Experiments

Experiment details. We use Adam optimizer with constant learning rate following the original DDPM paper. For CelebA, CIFAR-10 we use learning rate of $1e-4$, and for MNIST we use $1e-3$. During training, each round of local training corresponds to 1 epoch. For CelebA, we traing for a total of 800 epochs with 50 local iterations, for CIFAR-10 we train for 400 epochs with 100 local iterations, and for MNIST we train for 200 epochs using 20 local iterations. For fine-tuning, for competing methods $\times 0.1$ of original learning rate gives the best results, for our method we use 0.01 as the learning rate.

Model architecture. For every dataset we employ the same high-level design: a identity-embedding layer maps each identity token $y \in \{0, \ldots, m-1\}$ to a learnable vector of dimension d_{emb}; this vector is injected into an `UNet2DModel` backbone through modulating activations. Each backbone uses two residual blocks per scale, while its depth and channel widths are dataset-specific:

- **MNIST** (28×28, 1 channel): three scales with feature widths $(32, 64, 64)$. The encoder follows `DownBlock2D` → `AttnDownBlock2D` → `AttnDownBlock2D`; the decoder mirrors this with two `AttnUpBlock2D`s and a final `UpBlock2D`. This lightweight configuration suffices for the low-resolution digit images.
- **CIFAR-10** (32×32, 3 channels): four scales with widths $(64, 128, 128, 128)$. The down path is `DownBlock2D` → `AttnDownBlock2D` → `AttnDownBlock2D` → `DownBlock2D`; the up path is `UpBlock2D` → `AttnUpBlock2D` → `AttnUpBlock2D` → `UpBlock2D`. Both width and depth are doubled relative to MNIST to handle the richer natural-image content.
- **CelebA** (64×64, 3 channels): we reuse the CIFAR backbone.

Across all variants, the parameter count rises from MNIST to CIFAR/CelebA, yet the conditioning interface is unchanged. For our implementation we directly use label conditioning interface of UNet architecture.

CelebA qualitative results. In Figure 5 we can observe generated images for the modesl trained on CelebA dataset.

Shared representation approach architecture. Inspired by Collins et al. (2021), in our framework `SPIRE`, we introduce a novel *shared representation* approach as a strong competing baseline. Concretely, each client's U-Net is partitioned into three semantic parts: (i) the *down-sampling path*, responsible for hierarchical feature extraction; (ii) the *bottleneck (middle) block*, where the highest-level, most compressed latent representation is

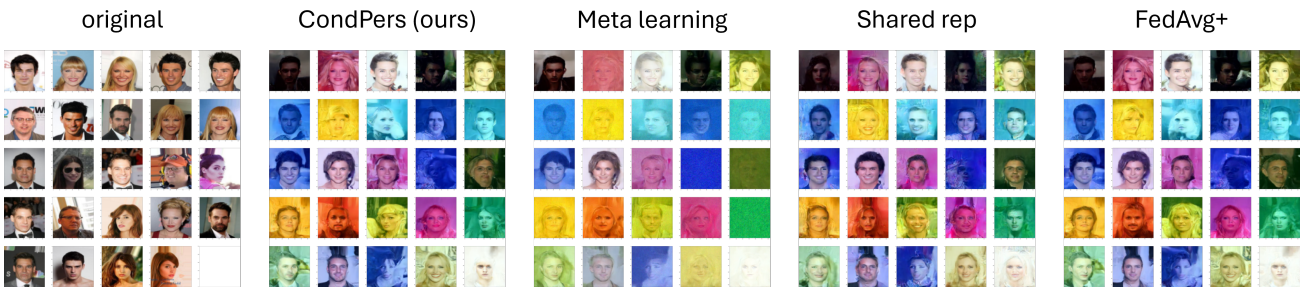


Figure 5: Grid of original CelebA images and generated images by competing methods for particular client, who has sampled data from 5 different individuals; using the same random seed.

processed; and (iii) the *up-sampling path*, which decodes latents back into the pixel space. During federated training, the down- and up-sampling paths are synchronized across clients at every communication round in `SPIRE` —mirroring the global feature extractor in Collins et al. (2021) so that low and high-level visual features are learned from the entire population's data as can be seen in Figure 6. In contrast, the bottleneck block is kept *client-specific* and never uploaded to the server. This design has several advantages:

1. **Personalization without classification head swapping.** Unlike classifiers, diffusion U-Nets do not have a lightweight prediction head that can be individualized. By individualizing the bottleneck, we inject client-specific inductive bias at the deepest latent level, where semantic factors are most disentangled, enabling fine-grained customization of generation quality and style.

2. **Better fine-tuning performance.** In `SPIRE`, global synchronization of the encoder and decoder paths anchors each client's individual bottleneck in a common feature space, mitigating overfitting to narrow data distributions and preserving cross-client coherence—a problem that pure fine-tuning baselines (e.g., FedAvg+FT) often suffer from. Note that, as we see in Section 5, conditional personalization within `SPIRE` still results in better fine-tuning performance.

Training hardware. For training we use a GPU server of 6 NVIDIA RTX2080 GPUs. The longest training is done on CelebA, which lasts around 60 hours.

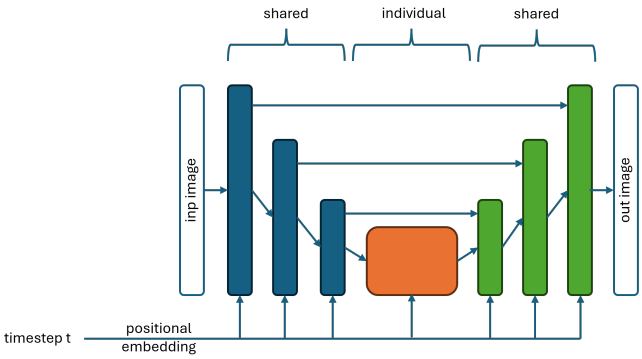


Figure 6: Shared representation partitioning.