

Rejection Sampling IMLE: Designing Priors for Better Few-Shot Image Synthesis

Chirag Vashist[✉], Shichong Peng[✉], and Ke Li[✉]

APEX Lab
School of Computing Science
Simon Fraser University
`{chirag_vashist, shichong_peng, keli}@sfu.ca`

Abstract. An emerging area of research aims to learn deep generative models with limited training data. Prior generative models like GANs and diffusion models require a lot of data to perform well, and their performance degrades when they are trained on only a small amount of data. A recent technique called Implicit Maximum Likelihood Estimation (IMLE) has been adapted to the few-shot setting, achieving state-of-the-art performance. However, current IMLE-based approaches encounter challenges due to inadequate correspondence between the latent codes selected for training and those drawn during inference. This results in suboptimal test-time performance. We theoretically show a way to address this issue and propose RS-IMLE, a novel approach that changes the prior distribution used for training. This leads to substantially higher quality image generation compared to existing GAN and IMLE-based methods, as validated by comprehensive experiments conducted on nine few-shot image datasets.

Keywords: Few-shot · Image Synthesis · Implicit Maximum Likelihood Estimation

1 Introduction

Recent years have witnessed significant advances in image synthesis, driven by the development of a broad variety of powerful generative models. Generative adversarial networks (GANs) [2, 7, 10, 11, 14], variational autoencoders (VAEs) [3, 16, 29, 36], diffusion models [4, 9], score-based models [34, 35], normalizing flows [5, 15, 17], and autoregressive models [6, 27, 28, 30] have demonstrably improved the quality of synthesized images, often achieving photorealism. However, to achieve this high fidelity, generative models often require large amounts of training data.

In some scenarios, there are not a lot of training examples available. Suppose we want to emulate the types of edits that a user manually made to a few images. In this scenario, we only have access to a limited number of training examples to begin with. In other cases, the training data can be hard to collect. In autonomous driving, synthesizing images for rare conditions like near misses can be challenging. There are also cases where collecting data is expensive. Suppose we want to train a 3D generative model, which requires 3D objects. Creating these 3D objects often involves expensive manual

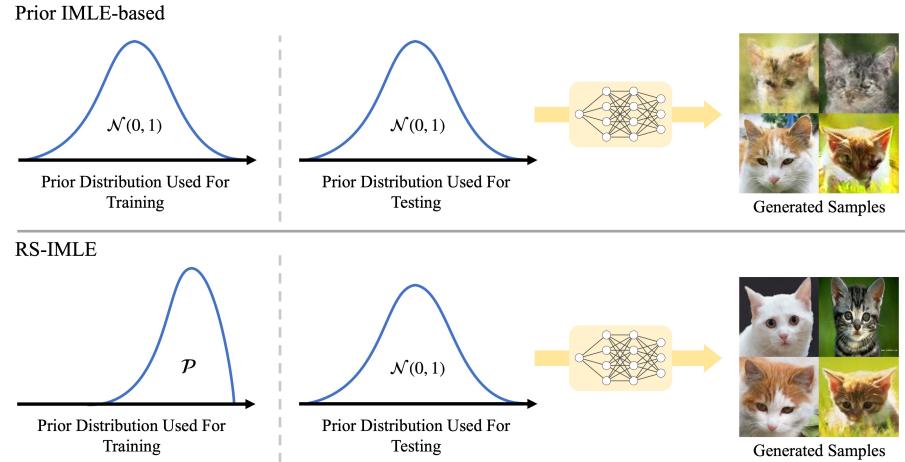
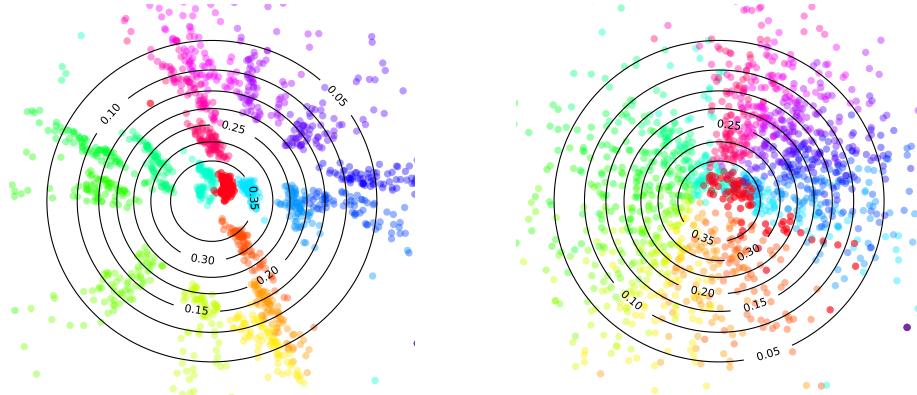


Fig. 1: IMLE is an implicit generative model that maps a latent code sampled from a prior distribution to an image output. In previous IMLE-based methods, both the training and testing phases adopt a standard normal distribution as the prior distribution. However, this approach often results in poor generalization during inference. To address this limitation, we introduce RS-IMLE, which uses rejection sampling to alter the prior distribution used for training to a different distribution \mathcal{P} . This modification significantly enhances the quality of generated images during testing.

labour or running reconstruction algorithms. In this paper, we aim to tackle the problem of high-quality image synthesis using limited training data.

The limited availability of training data in this context makes it crucial for generative models to fully leverage every provided example. Generative models that perform well in the large-scale setting, do not perform well in the few-shot setting. In diffusion models, the marginal likelihood under the forward process is a mixture of isotropic Gaussians. This modeling assumption smooths out the learned manifold along all directions, including those that are orthogonal to the actual data manifold. This becomes particularly problematic when there are a limited number of training examples (Figure 3). Hence, implicit generative models are commonly employed for few-shot generation, with the generator in GANs [18, 23, 24, 32, 38] serving as a notable example. However, GAN-based methods continue to be afflicted by mode collapse. Mode collapse occurs when the generator network fails to capture the full training data distribution and instead produces a limited subset of outputs. This phenomenon is especially problematic in scenarios where only a small number of training examples are provided.

Implicit Maximum Likelihood Estimation (IMLE) [21] is an alternative to the GAN objective and has shown promising results in addressing mode collapse. In contrast to GANs, which aim to make each generated image resemble some training data, IMLE instead ensures that each training image has *some* generated sample close to it, and therefore cannot drop any of the modes present in the training data. Adaptive IMLE [1] further extends IMLE to the few-shot image synthesis setting and achieves state-of-the-art generated image quality and mode coverage.



(a) Latent space of model trained by IMLE objective using standard normal prior. Dots represent the points selected by the model *over the course of training*, with dots of the same colour belonging to the same data point. The contours of standard normal distribution have been shown for comparison.

(b) Latent space of model trained by RS-IMLE objective using prior obtained via rejection sampling. Compared to latent space of model trained by IMLE, our method over the course of training, samples latent codes that follow the distribution at test time more faithfully.

Fig. 2: Difference between the latent codes picked by IMLE and RS-IMLE over the course of training.

However, in existing IMLE-based approaches, we observe that the latent codes used during training and those sampled during testing have different distributions, even though the same prior is used during training and testing. This phenomenon arises because of how IMLE selects latent codes during training. Some regions of the latent space are consistently rarely picked for training, despite having a high likelihood under the prior distribution (often the standard Gaussian). This is illustrated in Fig. 2a. Consequently, at test time, when latent codes drawn from the prior happen to fall in these regions, they yield low-quality samples that are far from the real data points, as illustrated in Fig. 1.

This issue has been observed in other generative models like VAEs. Hoffman et al [?] show that in practice the prior distribution $p(z)$ and the approximate posterior $q(z)$ are substantially different. Subsequent work [?] attempt to mitigate this mismatch by minimizing the KL-divergence between the prior distribution and the aggregate posterior. This approach in turn has its own drawbacks as it can lead to posterior collapse, which diminishes the generative capabilities of VAEs.

Rather than trying to change the objective like in the previous line of work, we address this issue by carefully choosing a different prior so that the samples selected for training have a distribution more similar to those sampled at inference. Our method, which we call Rejection Sampling IMLE or RS-IMLE for short, demonstrably improves coverage of the latent space used during training, thereby ensuring better alignment with the prior as shown in Fig. 2b. As a result, our method yields higher quality samples during testing compared to existing GAN and IMLE-based methods. We substantiate this claim through theoretical analysis and extensive experiments conducted on nine few-shot image datasets. We achieve an average of 45.9% decrease in FID [8] across datasets compared to the best baseline.

2 Related Work

Training deep generative models with limited data remains a significant challenge. One approach involves adapting a model pretrained on a large-scale auxiliary dataset from similar domains [22, 25, 26, 31, 37, 40]. However, the availability of such large-scale auxiliary datasets across all domains is not guaranteed. Therefore, another emerging line of work focuses on training models from scratch. In this context, due to the scarcity of training data, diffusion models struggle to achieve high-quality generated images and have been demonstrated to be ineffective [1]. As a result, previous works in this area predominantly build on Generative Adversarial Networks (GANs) and design various methods to address the well-known mode collapse issue. Techniques such as ADA [13] and DifffAug [41] aim to expand training data using adaptive and differentiable augmentation strategies. FastGAN [24] introduced a skip-layer excitation module for accelerated training and used self-supervision in the discriminator to enhance feature learning, thereby improving mode coverage of the generator. FakeCLR [23] enhances image synthesis by extensive data augmentation and applies contrastive learning solely on perturbed fake samples. FreGAN [38] introduces a frequency-aware model with a self-supervised constraint to avoid generating arbitrary frequency signals. ReGAN [32] dynamically adjusts GANs’ architecture during training to explore diverse sub-network structures at different training times. However, despite these advances, some degree of mode collapse persists.

In contrast, Implicit Maximum Likelihood Estimation (IMLE) [21] shows promising results in addressing mode collapse through the use of an alternative objective function compared to GANs. Building upon IMLE, Adaptive IMLE [1] adapts this approach to the few-shot image synthesis scenario by introducing individual target thresholds for each training data point. This dynamic adjustment of training progress accounts for varying difficulties across different data points, thereby effectively leveraging the limited training data. In this work, we introduce a novel algorithm, orthogonal to Adaptive IMLE, for sample selection during training.

3 Method

3.1 Background

In the context of unconditional image synthesis, the primary objective is to learn the unconditional probability distribution of images $p(\mathbf{x})$. This distribution enables the generation of novel synthesized images through sampling. Generator in GANs are represented by a function $T_\theta : Z \rightarrow X$, implemented as a neural network with parameters denoted as θ . The function T_θ learns a transformation from the latent space Z to the image space X by using adversarial training, which employs a discriminator that tries to distinguish between generated images $T_\theta(\mathbf{z})$ and real images \mathbf{x} , while the generator tries to produce increasingly realistic images to deceive the discriminator. However, this objective often leads to mode collapse, a well-known issue of GANs, where the generated output $T_\theta(\mathbf{z})$ only models a subset of the training examples.

To address the issue of mode collapse, an alternative method Implicit Maximum Likelihood Estimation (IMLE) [21] has been introduced. While IMLE, like GANs, uses a generator, it differs from GANs by using an alternative objective. The IMLE objective

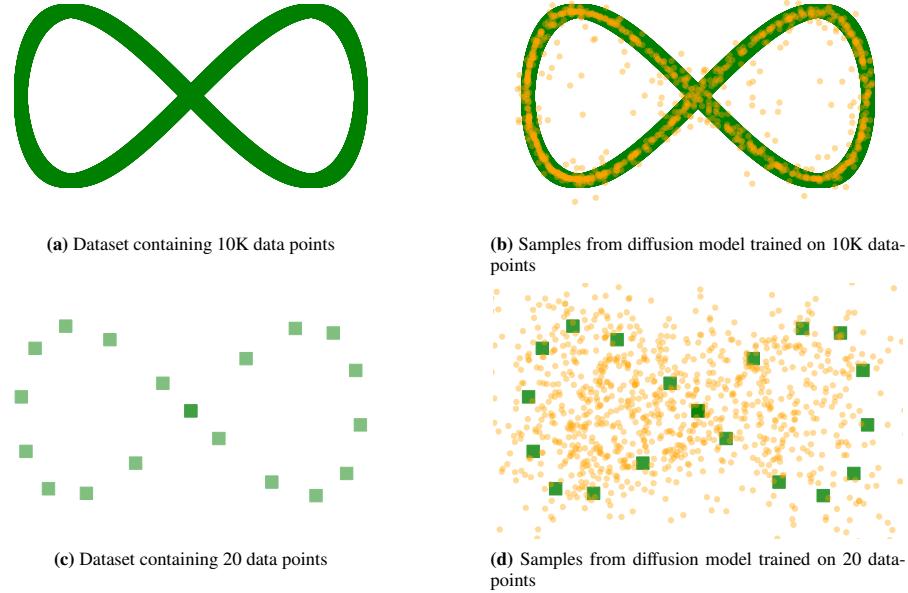


Fig. 3: Comparison between performance of diffusion models on large-scale and few-shot setting. We have two 2D datasets of the same shape (infinity symbol) but different number of data points: 10K data points 3a and 20 data points 3c. We train the *same model* but get very different performance. For the few-shot case (20 data points), the diffusion model fails to learn a distribution that matches the data distribution. Data points are denoted by ■ and samples are denoted by ●.

ensures that *each* training data point has similar generated samples, thereby encouraging coverage of all the modes of the training data.

The IMLE objective is given as follows, where $d(\cdot, \cdot)$ is a distance metric:

$$\theta_{\text{IMLE}} = \arg \min_{\theta} \mathbb{E}_{z_1, \dots, z_m \sim \mathcal{N}(0, I)} \left[\sum_{i=1}^n \min_{j \in [m]} d(\mathbf{x}_i, T_{\theta}(\mathbf{z}_j)) \right] \quad (1)$$

Here m denotes the number of samples and n denotes the number of data points. In simple terms, the IMLE objective first draws m samples \mathbf{z}_j from the standard Gaussian distribution and transforms them into the image space using the function T_{θ} . From these pool of samples in the image space, for each data point x_i , IMLE selects a sample that is *closest* to the data point in some distance metric $d(\cdot, \cdot)$. This operation can be done efficiently due to advances in high-dimensional nearest neighbour search [20]. Note that the number of samples m must at least be equal to the number of data points n , since otherwise by the pigeonhole principle, some samples would be picked by multiple data points. In practice, we find that setting m to be a multiplicative factor (like 10 or 20) times larger than n works the best.

3.2 Observation

In the existing IMLE-based methods, we observe that the distributions of the latent codes used for training the objective differs from the distribution of latent encountered at test time. Consider an illustrative example where the latent space is two dimensional. We train a simple generative model using IMLE on two dimensional toy dataset. The latent codes used for training over the course of training are illustrated in Figure 2a. We notice that for the latent codes belonging to the same data point (denoted by the same colour) form well-separated tight bands in the latent space. We also observe that there are large gaps between these bands, indicating that these segments of the latent space are consistently overlooked during training. Since at test time we sample from the same standard normal distribution, these unsupervised segments in the latent space have arbitrary outputs, which result in bad samples. We term this phenomenon the “*misalignment issue*.”

3.3 Analysis of the Misalignment Issue

In this section, we will explore why the phenomenon mentioned above occurs. Let us clarify the notation used in subsequent sections: \mathbf{x}_i denotes the i^{th} data point, $d(\cdot, \cdot)$ denotes the distance function and $T_\theta(\mathbf{z}_j)$ denote the j^{th} sample where $\mathbf{z}_j \sim \mathcal{N}(0, I)$.

Let us define a random variable, D_{ij} to denote the distance of i^{th} data point to the j^{th} sample. We can also define another random variable D_i^* to denote the distance of i^{th} data point to the sample *closest* to it. Further, let $F_{D_{ij}}$ and $F_{D_i^*}$ be the CDF of D_{ij} and D_i^* respectively. Let $f_{D_{ij}}$ and $f_{D_i^*}$ be the PDF of D_{ij} and D_i^* respectively. Now we will relate the CDF of the distances between a data point and its selected latent code, $F_{D_i^*}$ to the CDF of the distances between the same data point and a random latent code $F_{D_{ij}}$:

$$\begin{aligned} F_{D_i^*}(t) &= \Pr(D_i^* \leq t) = 1 - \Pr(D_i^* > t) \\ &= 1 - \Pr(D_{ij} > t, \forall j \in [m]) \quad (\text{Def. of } D_{ij}) \\ &= 1 - \prod_{j=1}^m \Pr(D_{ij} > t) \\ &= 1 - (\Pr(D_{i1} > t))^m \quad (D_{ij} \text{ are i.i.d}) \end{aligned} \quad (2)$$

$$= 1 - (1 - F_{D_{i1}}(t))^m \quad (3)$$

Note that Equation 2 is true because each \mathbf{z}_j is drawn independently from the same probability distribution which makes $D_{i1}, D_{i2} \dots D_{im}$ identical in distribution for a particular data point \mathbf{x}_i .

Now we can try to justify our observations from Figure 2a. Equation 3 shows us how the CDF of the distance of the selected latent code (used in training) differs significantly from the CDF of distance of a random latent code (encountered at test time). This shows that the distance of the sample we choose for training is typically lower than the distance of the sample at testing. We can obtain a deeper understanding by analyzing the plots

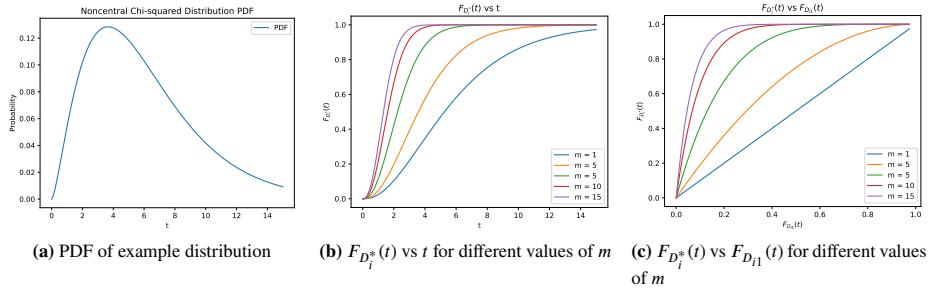


Fig. 4: Illustrative figure for demonstrating the behaviour of $F_{D_i^*}(t)$ and $F_{D_{i1}}(t)$ using Noncentral Chi-squared distribution as the example distribution.

in Figure 4 for an example distribution. Notice that $\forall m > 1, F_{D_i^*}(t) > F_{D_{i1}}(t)$. We can observe the following from Equation 3 and the aforementioned plots:

1. The $f_{D_i^*}$ is skewed towards the origin compared to $f_{D_{i1}}$. This is intuitive because the latent codes are selected by the min operation and so their distance to their respective data point would be less than that of a random sample.
2. The skew towards the data point increases as m increases. This observation will be important shortly.

We can also compute the PDF $f_{D_i^*}$ in terms of $f_{D_{i1}}$ by differentiating the CDF $F_{D_i^*}(t)$ as follows:

$$f_{D_i^*}(t) = \frac{dF_{D_i^*}(t)}{dt} = m \left(1 - F_{D_{i1}}(t)\right)^{m-1} f_{D_{i1}}(t) \quad (4)$$

3.4 Solving the Misalignment Issue

Now that we know the reason behind the misalignment between latent codes used at training and testing, we wish to come up with a method to mitigate this phenomenon. Recall that D_{ij} and D_i^* are determined by the distance function $d(\cdot, \cdot)$, neural network T_θ and the prior distribution. For a given generative modelling task, it is not trivial to change the $d(\cdot, \cdot)$ or T_θ . Hence, in this paper, we aim to change the prior distribution used at training such that the distribution of latent codes at training time closely match with the distribution of latent codes (drawn from the standard normal distribution) encountered at test time. Notice that the IMLE objective (Equation 1) allows us to sample from the prior without knowing the closed form expression for its probability density function. This allows us the flexibility of choosing a non-analytical prior distribution. To distinguish from $\mathbf{z}_j \sim \mathcal{N}(0, I)$, we will use $\tilde{\mathbf{z}}$ to denote the latent codes drawn from our desired target distribution \mathcal{P} . Identical to the previous section, let us define random variable $\tilde{D}_{ij} = d(\mathbf{x}_i, T_\theta(\tilde{\mathbf{z}}_j))$ to denote the distance between data point \mathbf{x}_i from a random sample $T_\theta(\tilde{\mathbf{z}}_j)$. Similarly, we define $\tilde{D}_i^* = \min_{j \in [m]} \tilde{D}_{ij}$. Similar to the section above, let $F_{\tilde{D}_{ij}}$

Algorithm 1 RS-IMLE Procedure

Require: The set of inputs $\{\mathbf{x}_i\}_{i=1}^n$, radius ϵ

- 1: Initialize the parameters θ of the generator T_θ
- 2: **for** $k = 1$ **to** K **do**
- 3: Draw latent codes $Z \leftarrow \mathbf{z}_1, \dots, \mathbf{z}_m$ from $\mathcal{N}(0, \mathbf{I})$
- 4: Compute $\tilde{Z} \leftarrow \tilde{\mathbf{z}}_1, \dots, \tilde{\mathbf{z}}_p$ from Z such that

$$d(\mathbf{x}_i, T_\theta(\tilde{\mathbf{z}}_j)) \geq \epsilon, \quad \forall \tilde{\mathbf{z}}_j \in \tilde{Z}, i \in [n]$$
- 5: $\sigma(i) \leftarrow \arg \min_{j \in [m]} d(\mathbf{x}_i, T_\theta(\tilde{\mathbf{z}}_j)), \quad \forall i \in [n]$
- 6: **for** $l = 1$ **to** L **do**
- 7: Pick a random batch $S \subseteq [n]$
- 8: $\theta \leftarrow \theta - \eta \nabla_\theta (\sum_{i \in S} d(\mathbf{x}_i, T_\theta(\tilde{\mathbf{z}}_{\sigma(i)}))) / |S|$
- 9: **end for**
- 10: **end for**
- 11: **return** θ

and $F_{\tilde{D}_i^*}$ be the CDF of \tilde{D}_{ij} and \tilde{D}_i^* respectively. Then $f_{\tilde{D}_{ij}}$ and $f_{\tilde{D}_i^*}$ would be the PDF of \tilde{D}_{ij} and \tilde{D}_i^* respectively.

Designing the target prior Now, we discuss the desired properties for our target prior. Recall that the misalignment issue is mitigated as the number of samples, denoted by m , decreases. In order to differentiate between the number of samples of different priors, we use the notation m' to denote the number of samples for the objective using the Gaussian prior. As hinted in the previous section, one way to avoid the misalignment issue is to set m' to low values. However we cannot directly use a Gaussian prior with arbitrary low values of m' as our target prior. This is because having too few samples to choose from would cause many data points to pick the same sample as their nearest neighbour. Since the objective function tries to pull the nearest sample toward each data point, pulling the same sample towards different data points creates conflicting supervision signals, leading to slow convergence or even no learning, especially when target data points for the same sample lie in opposite directions. Hence when using a Gaussian prior, we need to pick m' large enough to allow convergence and yet small enough such that the misalignment issue does not affect test time performance. In our case, we are trying to design a new distribution that solves the misalignment issue by having desirable properties of an ideal distribution. The ideal distribution is a Gaussian prior with m' set to the lowest possible value, which is $m' = n$.

To this end, we can choose a prior distribution \mathcal{P} that matches the ideal prior distribution. Similar to the analysis till Equation 4, we derive the PDF of \mathcal{P} . We get $f_{\tilde{D}_i^*}(t) = m \left(1 - F_{\tilde{D}_{i1}}(t)\right)^{m-1} f_{\tilde{D}_{i1}}(t)$. Equating this PDF of \mathcal{P} to the PDF of the ideal distribution gives:

$$\begin{aligned}
m \left(1 - F_{\tilde{D}_{i1}}(t)\right)^{m-1} f_{\tilde{D}_{i1}}(t) &= n \left(1 - F_{D_{i1}}(t)\right)^{n-1} f_{D_{i1}}(t) \\
\implies f_{\tilde{D}_{i1}}(t) &= \frac{n}{m} \frac{\left(1 - F_{D_{i1}}(t)\right)^{n-1}}{\left(1 - F_{\tilde{D}_{i1}}(t)\right)^{m-1}} f_{D_{i1}}(t)
\end{aligned} \tag{5}$$

We introduce $\phi(t) = \frac{n}{m} \frac{\left(1 - F_{D_{i1}}(t)\right)^{n-1}}{\left(1 - F_{\tilde{D}_{i1}}(t)\right)^{m-1}}$ to simplify notation. Hence, we can write Equation 5 as:

$$f_{\tilde{D}_{i1}}(t) = \phi(t) f_{D_{i1}}(t) \tag{6}$$

Rejection sampling We have expressed our target prior \mathcal{P} in terms of distribution from which we can easily sample. Now, we can use rejection sampling to sample from our target prior \mathcal{P} .

To be concrete: $f_{\tilde{D}_{i1}}(t)$ is our target distribution and $f_{D_{i1}}(t)$ acts as our proposal distribution, since we can sample from the standard Gaussian easily. In order to ensure that the acceptance ratio is bounded, we introduce a constant c associated with truncating $F_{D_{i1}}(t)$. We discuss these technical details in the appendix.

We can write the acceptance ratio in the standard rejection sampling notation: $\frac{f_{\tilde{D}_{i1}}(t)}{M f_{D_{i1}}(t)} = \frac{c\phi(t)}{M}$

Here, M is the scaling factor associated with rejection sampling. We approximate the function above using a step function. The step needs to happen at t where $F_{\tilde{D}_{i1}}(t)$ gets close to 1. Instead of trying to estimate $F_{\tilde{D}_{i1}}(t)$, we instead use a hyperparameter ϵ to represent where $F_{\tilde{D}_{i1}}(t)$ gets close to 1. We find the value of this hyperparameter ϵ by cross-validation.

The final procedure simplifies to this: we sample $\mathbf{z} \sim \mathcal{N}(0, I)$. If $t = d(\mathbf{x}_i, T_\theta(\mathbf{z})) < \epsilon$, we reject the sample; otherwise, we accept it. Since the sampling procedure is based on rejection sampling, we call our method *RS-IMLE*. The resulting RS-IMLE procedure is included in Algorithm-1.

3.5 Intuitive Interpretation of the Algorithm Behaviour

Prior to proceeding to the implementation details, gaining a gradient-based understanding of our new objective would provide valuable insight. Let us revisit the vanilla IMLE objective (Equation 1) again. As training progresses, we expect the loss to reduce such that: $\forall \mathbf{x}_i, \mathbb{E}_{z_1, \dots, z_m \sim \mathcal{N}(0, I)} \min_{j \in [m]} d(\mathbf{x}_i, T_\theta(\mathbf{z}_j)) \rightarrow 0$. As the objective approaches convergence, the loss will decrease, resulting in gradients with lower magnitude. This causes smaller updates to the model parameters during training, leading to slower progress. Note that this loss is with respect to the *closest* sample to each data point. Thus it can

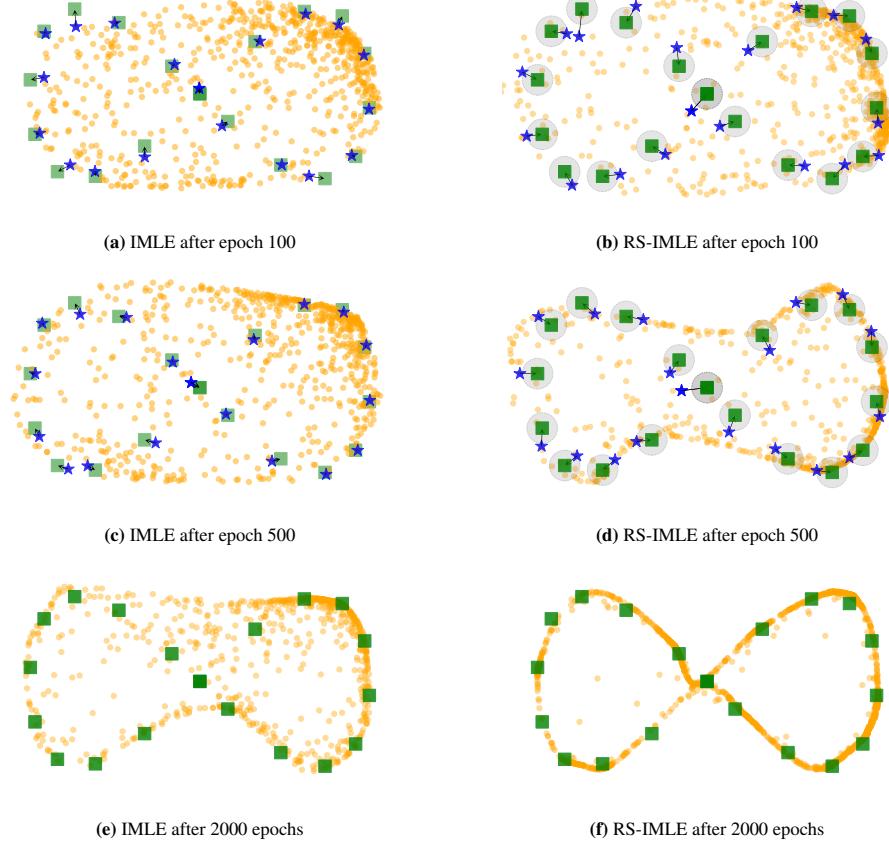


Fig. 5: Comparison between IMLE and RS-IMLE for 2D toy problem. Data points are denoted by \blacksquare and samples are denoted by \bullet . Samples picked as nearest neighbours are denoted by \star .

be the case that even after a lot of training, although the *closest* sample are pretty close to their respective data point, the rest of samples are pretty far away.

$$\theta_{\text{RS-IMLE}} = \arg \min_{\theta} \mathbb{E}_{z_1, \dots, z_m \sim \mathcal{P}} \left[\sum_{i=1}^n \min_{j \in [m]} d(\mathbf{x}_i, T_{\theta}(\mathbf{z}_j)) \right] \quad (7)$$

Consider our objective in Equation 7 and recall that we have constructed the probability distribution \mathcal{P} such that $\forall \mathbf{x}_i \quad d(\mathbf{x}_i, T_{\theta}(\tilde{\mathbf{z}})) \geq \epsilon$, where $\tilde{\mathbf{z}} \sim \mathcal{P}$. In other words, all samples we obtain by using the prior \mathcal{P} are guaranteed to be ϵ -distance away from all data points. This ensures that the loss per data point is always greater than ϵ . The approach can be interpreted as ignoring the samples that are already close to some data point and instead training on challenging, non-trivial samples.

To compare the sampling behavior of the vanilla IMLE and the proposed RS-IMLE, we trained two models on a 2D toy problem as illustrated in Figure 5. The first model uses the vanilla IMLE objective (Equation 1), while the second model is trained with our proposed RS-IMLE objective (Equation 7).

At the initial stage of training, both the methods learn similar distributions, indicated by the straight line of orange dots (●) in Figure-5a,5b. Our method first removes all the samples that fall within an ϵ distance from any data point before doing the nearest neighbour search. We illustrate this in Figure-5b, where samples that lie within the gray circles are not considered for the nearest neighbour search.

As training progresses, we notice that for both the algorithms' samples move closer to the ground truth data points. However, for vanilla IMLE, we observe that for many data points the sample picked after the nearest neighbour search is already close to the ground truth. In this case, the loss associated with these data points would be low (indicated by the short length of arrow in the Figure-5c). As a result, the model trained by the vanilla IMLE objective does not learn anything significantly novel. In our proposed method (Figure-5d), each data point selects a sample that is at least ϵ distance away from it. This ensures that the loss for each data point is always sufficiently high (indicated by the long arrows), resulting in meaningful updates to the model parameters.

3.6 Implementation Details

Note that computing the distance of each sample with each data point is computationally expensive. Suppose we have n data points in \mathbb{R}^d and m samples, calculating the distance between each pair has a time complexity of $\mathcal{O}(mnd)$. To get around this issue, we leverage a fast k-nearest neighbor search method, DCI [20]. This method reduces the runtime of a single query from linear to sublinear in m , enabling us to efficiently *filter out* all the samples that are within an ϵ distance from any data point. Subsequently, from this filtered pool of remaining samples, we select, for each data point, the sample that is closest to it.

In order to reduce the search time complexity further, we can project the training data to a lower dimensional subspace (which would decrease d). We use this while implementing the procedure for image synthesis task: we first flatten each image and project it to a lower dimension by using random projection. We normalize these projected vectors before using them for nearest neighbour search.

4 Experiments

Datasets We assess our method and the baseline approaches across a variety of datasets with 256×256 resolution. These datasets include Animal-Face Dog [33], Animal-Face Cat [33], Obama [41], Panda [41], Grumpy-cat [41], Anime [24], Shells [24], Skulls [24] and a subset of Flickr-FaceHQ (FFHQ) [12] which are standard datasets used in the few-shot learning literature.

Baselines We compare our method to recent state-of-the-art few-shot image generation methods. These include FastGAN [24], FakeCLR [23], FreGAN [38], Re-GAN [32] and AdaIMLE [1].

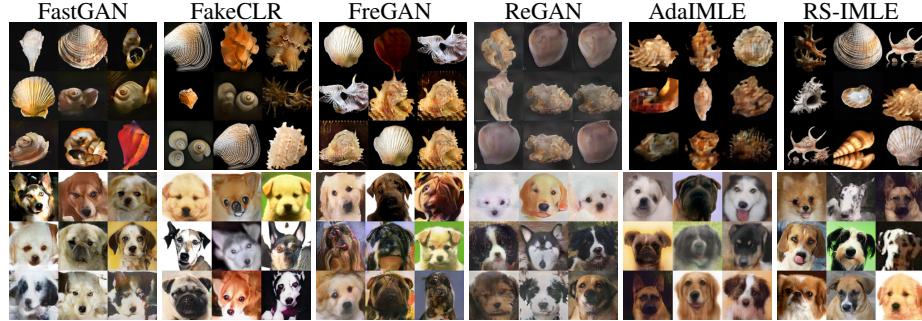


Fig. 6: Qualitative comparison between our method and baselines. While analyzing the images, look for the sharpness of each image and diversity in the content of all images for a method.

Dataset	FastGAN [24]	FakeCLR [23]	FreGAN [38]	ReGAN [32]	AdaIMLE [1]	RS-IMLE (Ours)	Imp. %
Obama	41.1	29.9	33.4	45.7	25.0	14.0	44.0
Grumpy Cat	26.6	20.6	24.9	27.3	19.1	11.5	39.8
Panda	10.0	8.8	9.0	12.6	7.6	3.5	54.0
FFHQ-100	54.2	62.1	50.5	87.4	33.2	12.9	61.1
Cat	35.1	27.4	31.0	42.1	24.9	15.9	36.1
Dog	50.7	44.4	47.9	57.2	43.0	23.1	46.3
Anime	69.8	77.7	59.8	110.8	65.8	35.8	45.6
Skulls	109.6	106.5	163.3	130.7	81.9	51.1	37.6
Shells	120.9	148.4	169.3	236.1	108.5	55.4	48.9

Table 1: We compute FID [8] between the real data and 5000 randomly generated samples for all the methods. Lower is better.

Evaluation Metrics We employ Fréchet Inception Distance (FID) [8] to assess the perceptual quality of the generated images. This involves randomly generating 5000 images and calculating the FID between these generated samples and the real images for each dataset. Additionally, we evaluate the modelling accuracy and coverage by computing precision and recall for 1000 images using the metric defined by Kynkänniemi et al. [19]. In image synthesis, precision refers to the model’s capacity to generate images closely resembling the desired target or distribution. Recall measures the model’s ability to encompass a wide array of diverse images within the target distribution. To ensure that the computed metrics have low variance, we generate many more samples than training images.

Network Architecture We construct our generator network using decoder modules from VDVAE [3]. More details about the network architecture can be found in the Appendix.

4.1 Quantitative Results

In Table 1, we present the FID scores computed for all the datasets across different methods. Lower FID scores indicates that the distribution of generated images is closer

to the distribution of real images. Our method performs significantly better compared to baselines.

In Table 2, we show the precision and recall scores. Our method has a near perfect precision (close to 1), while having a significantly higher recall compared to the baselines. In the few cases where our method is not the best, it is very close to the best metric.

Dataset		<i>FastGAN</i> [24]	<i>FakeCLR</i> [23]	<i>FreGAN</i> [38]	<i>ReGAN</i> [32]	<i>AdaIMLE</i> [1]	<i>RS-IMLE</i> (<i>Ours</i>)
Obama	Prec.	0.92	0.96	0.82	0.62	0.99	0.98
	Rec.	0.09	0.30	0.33	0.01	0.68	0.82
Grumpy Cat	Prec.	0.91	0.97	0.90	0.78	0.97	0.93
	Rec.	0.13	0.39	0.23	0.04	0.72	0.95
Panda	Prec.	0.96	0.97	0.92	0.93	0.98	0.99
	Rec.	0.16	0.41	0.17	0.01	0.63	0.97
FFHQ-100	Prec.	0.91	0.71	0.86	0.39	0.99	1.00
	Rec.	0.13	0.25	0.21	0.01	0.77	0.99
Cat	Prec.	0.97	0.99	0.95	0.90	0.98	0.96
	Rec.	0.08	0.55	0.31	0.15	0.86	0.98
Dog	Prec.	0.96	0.95	0.92	0.84	0.97	0.98
	Rec.	0.19	0.34	0.20	0.10	0.61	0.94
Anime	Prec.	0.86	0.88	0.88	0.29	0.92	0.95
	Rec.	0.08	0.09	0.09	0.01	0.59	0.91
Skulls	Prec.	0.78	0.66	0.66	0.66	0.95	0.99
	Rec.	0.03	0.01	0.01	0.01	0.32	0.65
Shells	Prec.	0.92	0.87	0.87	0.50	0.97	0.98
	Rec.	0.03	0.01	0.01	0.01	0.62	0.59

Table 2: We compute precision and recall [42] between the real data and 1000 randomly generated samples for all the methods. Higher values are better.

4.2 Qualitative Results

Figure 6 compares the random samples of our method to that of several baselines, and we observe that our method produces overall sharper and more diverse images. In addition, we propose Visual Recall, a simple test to substantiate the qualitative superiority of our method. For each method, we first generate 1000 samples. Next, using a real image from the dataset as a query, we find the images from the pool of generated samples that are closest to the query image. We use LPIPS features [39] for the computing the distance between real images and samples. Figure 7 shows the results for the proposed test for different datasets and methods. We see that the samples produced by our method are visually similar to the query, while being sharp and diverse in attributes like hair colour, smile and jaw structure. Note that other methods do not have samples that closely resemble the query image.

Figure 8 shows results of spherical linear interpolation between two random points in the latent space for different datasets. The images transition in a meaningful manner,

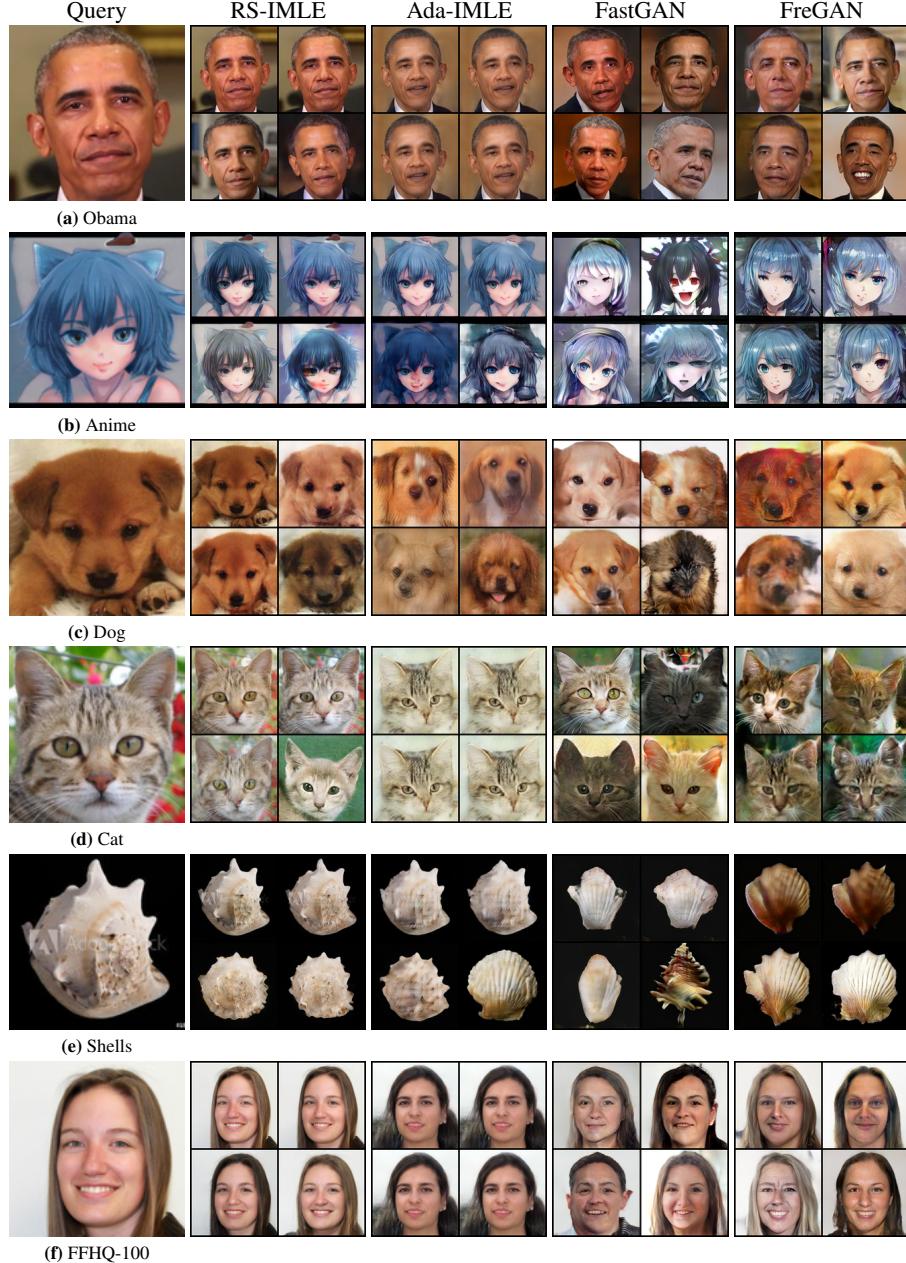


Fig. 7: Visual Recall test. The first column is the query image from the dataset. Subsequent columns are the samples produced by different methods that are closest to the query image in LPIPS feature space. The samples produced by our method are closer to the query images compared to the baselines, while remaining diverse.

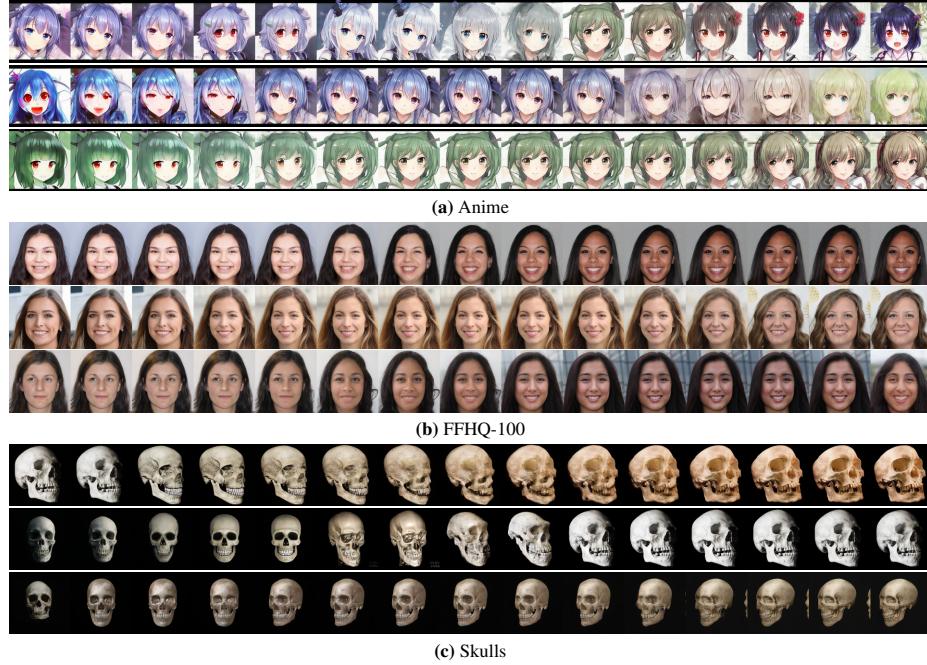


Fig.8: Latent space interpolation. We observe that the output images changes smoothly in a meaningful manner.

indicating that our model has learnt a continuous and structured latent space representation of the image distribution.

4.3 Ablation Study

Table-3 presents the FID computed for different values of ϵ for three datasets. We observe that our approach works best for values of ϵ close to 0.15 and that increasing the value of ϵ beyond a certain range degrades the performance.

Dataset	0.12	0.15	0.18	0.22
FFHQ-100	13.6	12.91	13.01	14.8
Obama	14.44	14.03	14.62	14.34
Cat	17.29	15.96	16.12	16.27

Table 3: FID for different values of ϵ

5 Conclusion

In this paper, we identified a latent space misalignment between the training and testing phases of existing IMLE-based methods, resulting in poor performance in few-shot image synthesis tasks. To address this issue, we introduced a novel algorithm, RS-IMLE, which modifies the prior distribution used for training. Our experimental results demonstrate that our method significantly enhances the quality of generated images and mode coverage during inference.

Acknowledgements: This research was enabled in part by support provided by NSERC, the BC DRI Group and the Digital Research Alliance of Canada. The authors would also like to thank Tristan Engst for extensive help polishing our paper.

References

1. Aghabozorgi, M., Peng, S., Li, K.: Adaptive IMLE for few-shot pretraining-free generative modelling. In: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (eds.) Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 202, pp. 248–264. PMLR (23–29 Jul 2023), <https://proceedings.mlr.press/v202/aghabozorgi23a.html>
2. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. ArXiv **abs/1809.11096** (2019)
3. Child, R.: Very deep vaes generalize autoregressive models and can outperform them on images. ArXiv **abs/2011.10650** (2021)
4. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. ArXiv **abs/2105.05233** (2021)
5. Dinh, L., Sohl-Dickstein, J.N., Bengio, S.: Density estimation using real nvp. ArXiv **abs/1605.08803** (2017)
6. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 12868–12878 (2020), <https://api.semanticscholar.org/CorpusID:229297973>
7. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
8. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: NIPS (2017)
9. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. ArXiv **abs/2006.11239** (2020)
10. Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., Aila, T.: Alias-free generative adversarial networks. In: NeurIPS (2021)
11. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4401–4410 (2019)
12. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks (2019)
13. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 8107–8116 (2019), <https://api.semanticscholar.org/CorpusID:209202273>
14. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 8107–8116 (2020)
15. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. ArXiv **abs/1807.03039** (2018)
16. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
17. Kobyzhev, I., Prince, S., Brubaker, M.A.: Normalizing flows: An introduction and review of current methods. IEEE Transactions on Pattern Analysis and Machine Intelligence **43**, 3964–3979 (2021)
18. Kong, C., Kim, J., Han, D., Kwak, N.: Few-shot image generation with mixup-based distance learning (2022)
19. Kynkänniemi, T., Karras, T., Laine, S., Lehtinen, J., Aila, T.: Improved precision and recall metric for assessing generative models. Advances in Neural Information Processing Systems **32** (2019)

20. Li, K., Malik, J.: Fast k-nearest neighbour search via prioritized dci (2017)
21. Li, K., Malik, J.: Implicit maximum likelihood estimation. arXiv preprint arXiv:1809.09087 (2018)
22. Li, Y., Zhang, R., Lu, J., Shechtman, E.: Few-shot image generation with elastic weight consolidation. ArXiv **abs/2012.02780** (2020)
23. Li, Z., Wang, C., Zheng, H., Zhang, J., Li, B.: Fakeclr: Exploring contrastive learning for solving latent discontinuity in data-efficient gans. In: ECCV (2022)
24. Liu, B., Zhu, Y., Song, K., Elgammal, A.: Towards faster and stabilized GAN training for high-fidelity few-shot image synthesis. CoRR **abs/2101.04775** (2021), <https://arxiv.org/abs/2101.04775>
25. Mo, S., Cho, M., Shin, J.: Freeze discriminator: A simple baseline for fine-tuning gans. ArXiv **abs/2002.10964** (2020)
26. Ojha, U., Li, Y., Lu, J., Efros, A.A., Lee, Y.J., Shechtman, E., Zhang, R.: Few-shot image generation via cross-domain correspondence. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 10738–10747 (2021)
27. van den Oord, A., Kalchbrenner, N., Espeholt, L., Kavukcuoglu, K., Vinyals, O., Graves, A.: Conditional image generation with pixelcnn decoders. ArXiv **abs/1606.05328** (2016)
28. van den Oord, A., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. ArXiv **abs/1601.06759** (2016)
29. Razavi, A., van den Oord, A., Vinyals, O.: Generating diverse high-fidelity images with vq-vae-2. ArXiv **abs/1906.00446** (2019)
30. Salimans, T., Karpathy, A., Chen, X., Kingma, D.P.: Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. ArXiv **abs/1701.05517** (2017)
31. Sauer, A., Chitta, K., Muller, J., Geiger, A.: Projected gans converge faster. In: Neural Information Processing Systems (2021), <https://api.semanticscholar.org/CorpusID:240354401>
32. Saxena, D., Cao, J., Xu, J., Kulshrestha, T.: Re-gan: Data-efficient gans training via architectural reconfiguration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16230–16240 (June 2023)
33. Si, Z., Zhu, S.C.: Learning hybrid image templates (hit) by information projection. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**, 1354–1367 (2012)
34. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. ArXiv **abs/1907.05600** (2019)
35. Song, Y., Sohl-Dickstein, J.N., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. ArXiv **abs/2011.13456** (2021)
36. Vahdat, A., Kautz, J.: Nvae: A deep hierarchical variational autoencoder. ArXiv **abs/2007.03898** (2020)
37. Wang, Y., Gonzalez-Garcia, A., Berga, D., Herranz, L., Khan, F.S., van de Weijer, J.: Minegan: Effective knowledge transfer from gans to target domains with few images. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 9329–9338 (2020)
38. Yang, M., Wang, Z., Chi, Z., Zhang, Y.: Fregan: Exploiting frequency components for training gans under limited data (2022)
39. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
40. Zhao, M., Cong, Y., Carin, L.: On leveraging pretrained gans for generation with limited data. In: ICML (2020)
41. Zhao, S., Liu, Z., Lin, J., Zhu, J.Y., Han, S.: Differentiable augmentation for data-efficient gan training. ArXiv **abs/2006.10738** (2020)

42. Zhu, P., Abdal, R., Qin, Y., Wonka, P.: Improved stylegan embedding: Where are the good latents? ArXiv **abs/2012.09036** (2020)

A Theory

Since we use rejection sampling, we need to ensure that the acceptance ratio is bounded. Recall that:

$$\begin{aligned} m \left(1 - F_{\tilde{D}_{i1}}(t)\right)^{m-1} f_{\tilde{D}_{i1}}(t) &= n \left(1 - F_{D_{i1}}(t)\right)^{n-1} f_{D_{i1}}(t) \\ \implies f_{\tilde{D}_{i1}}(t) &= \frac{n}{m} \frac{\left(1 - F_{D_{i1}}(t)\right)^{n-1}}{\left(1 - F_{\tilde{D}_{i1}}(t)\right)^{m-1}} f_{D_{i1}}(t) \end{aligned} \quad (8)$$

Notice that because of $\left(1 - F_{\tilde{D}_{i1}}(t)\right)^{m-1}$ term in the denominator, we have to make sure that the expression for $f_{\tilde{D}_{i1}}(t)$ is bounded. One way to do that is to truncate the right tail of the ideal distribution $f_{D_{i1}}(t)$ to 0. More explicitly, for a very large T (e.g., $T = 100,000$), we can write:

$$g_{D_{i1}}(t) = \begin{cases} f_{D_{i1}}(t) & \text{if } t \leq T \\ 0 & \text{if } t > T \end{cases}$$

Here $g_{D_{i1}}(t)$ is the PDF of the truncated distribution. Since very large values of distances (t) are rarely observed at test time, so applying this truncation has little effect in practice. Instead of writing the expression for Equation 8 in terms of $g_{D_{i1}}(t)$, we continue to use $f_{D_{i1}}(t)$ along with a constant c associated with the truncation.

Hence using $\phi(t) = \frac{n}{m} \frac{\left(1 - F_{D_{i1}}(t)\right)^{n-1}}{\left(1 - F_{\tilde{D}_{i1}}(t)\right)^{m-1}}$ and c as the constant associated with the truncation described above, we can write Equation 8 as:

$$f_{\tilde{D}_{i1}}(t) = c\phi(t)f_{D_{i1}}(t) \quad (9)$$

B Network Architecture

Our network architecture is illustrated in Figure 9, comprising a fully-connected mapping network inspired by [11] and a generator network constructed using decoder modules from VDVAE [3]. We choose an input latent dimension of 1024 for all datasets.

C Experiments

Table-4 gives the details about the number of images in each dataset as well as the value of radius used in the rejection sampling procedure (epsilon , ϵ) used in the results presented in the main paper. The selection of epsilon values was conducted through the process of hyperparameter tuning. We present an ablation study with different values of epsilon later in the paper.

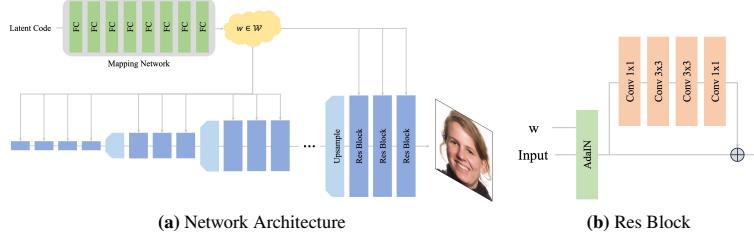


Fig. 9: (a) Network architecture, which comprises of a mapping network, upsampling layers and res blocks (details in (b)). (b) Inner workings of res blocks.

	Obama	Grumpy Panda Cat	FFHQ-100	Cat	Dog	Anime	Skulls	Shells
Num. of Images	100	100	100	100	160	389	120	96
Epsilon Used	0.15	0.18	0.18	0.15	0.15	0.15	0.18	0.18

Table 4: Number of images in each dataset and the value of epsilon used.

C.1 Random samples

In Figure 10, we compare the random samples of our method to that of the baseline for more datasets.

C.2 Visual Recall

Figure 11 shows the results for the proposed Visual Recall test for more queries. Note how the images produced by our method are the closest to the query and yet have diverse *meaningful* changes.

Since the images displayed are the *nearest neighbours* of the query images, it would be valuable to emphasize the subtle distinctions in the samples produced by our method. In Figure 11a and 11b, we can notice a change in the texture and color of the skin and hair of our samples. In Figure 11c and 11d, we can observe subtle changes to the jaw structure, number of teeth and hue of the different skull samples. Similarly in Figure 11e, we can notice subtle changes in the color of the fur and tilt of the head for different cat samples. In Figure 11g, we observe diversity in hair color, background and ear of the produce samples.

C.3 Ablation on latent dimensions and model parameters

Table 5 gives the details about the architectures used by the different methods. To decouple the impact of our proposed method (RS-IMLE) from architectural choices, we train using our method using lower latent dimensions. At lower dimensions, the number of parameters for RS-IMLE are significantly lower compared to the other methods. We tabulate the FID for the three most challenging datasets in the last three columns

Method	Dim.	Params.	Anime	Shells	Skulls
FastGAN	256	29M	69.8	120.9	109.6
FakeCLR	512	24M	77.7	148.4	106.5
FreGAN	256	147M	59.8	169.3	163.3
ReGAN	512	24M	110.8	236.1	130.7
AdaIMLE	1024	36M	65.8	108.5	81.9
RS-IMLE	1024	36M	35.8	55.4	51.1
	512	19M	48.5	52.9	60.1
	256	12M	53.8	71.7	64.3

Table 5: Comparison between different methods: latent dimensions and number of trainable parameters. Last three columns are FID on Anime, Shells and Skulls dataset.

of Table 5. As we decrease the number of dimensions (and consequently the number of parameters), we observe a slight drop in the FID for our method. However, even at *significantly* lower parameter count, our method *outperforms* the baselines.

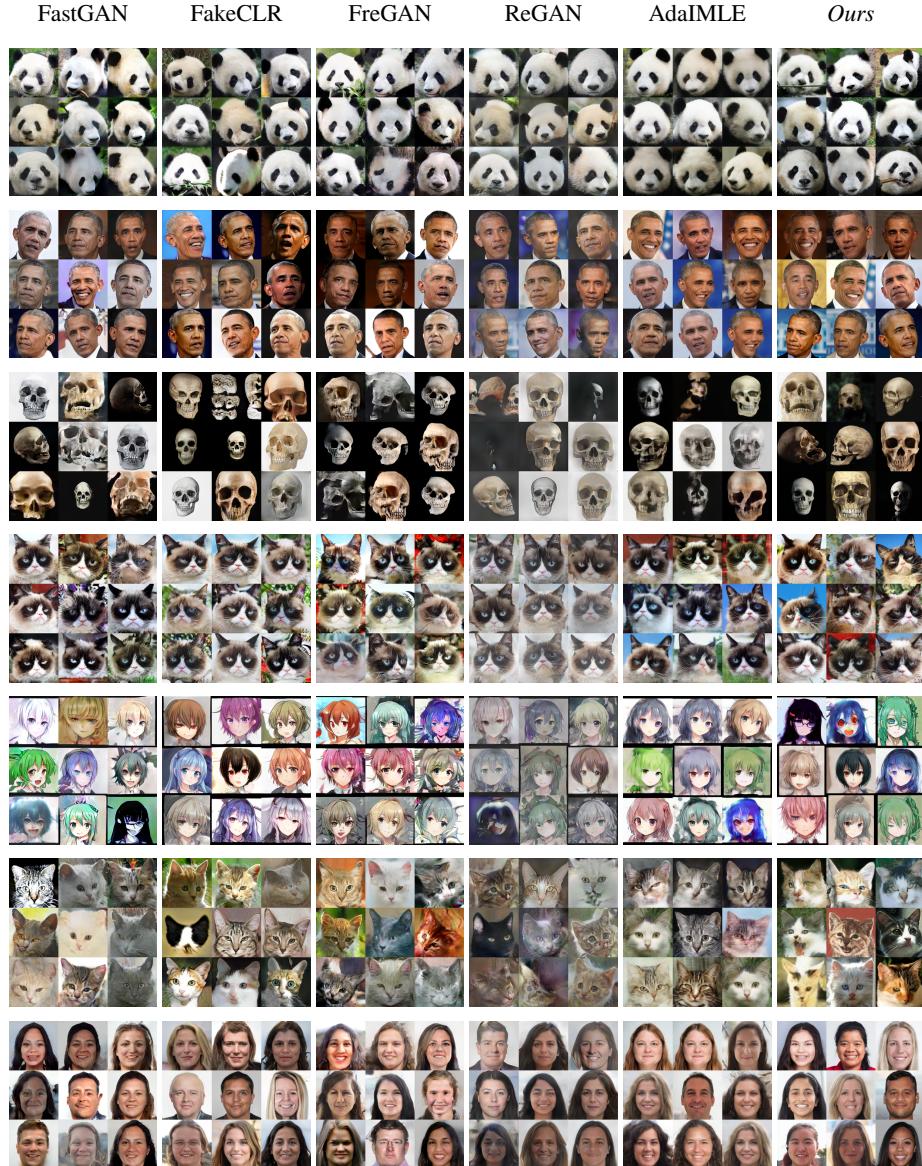


Fig. 10: Qualitative comparison between our method and baselines. While analyzing the images, look for the sharpness of each image and diversity in the content of all images for a method.

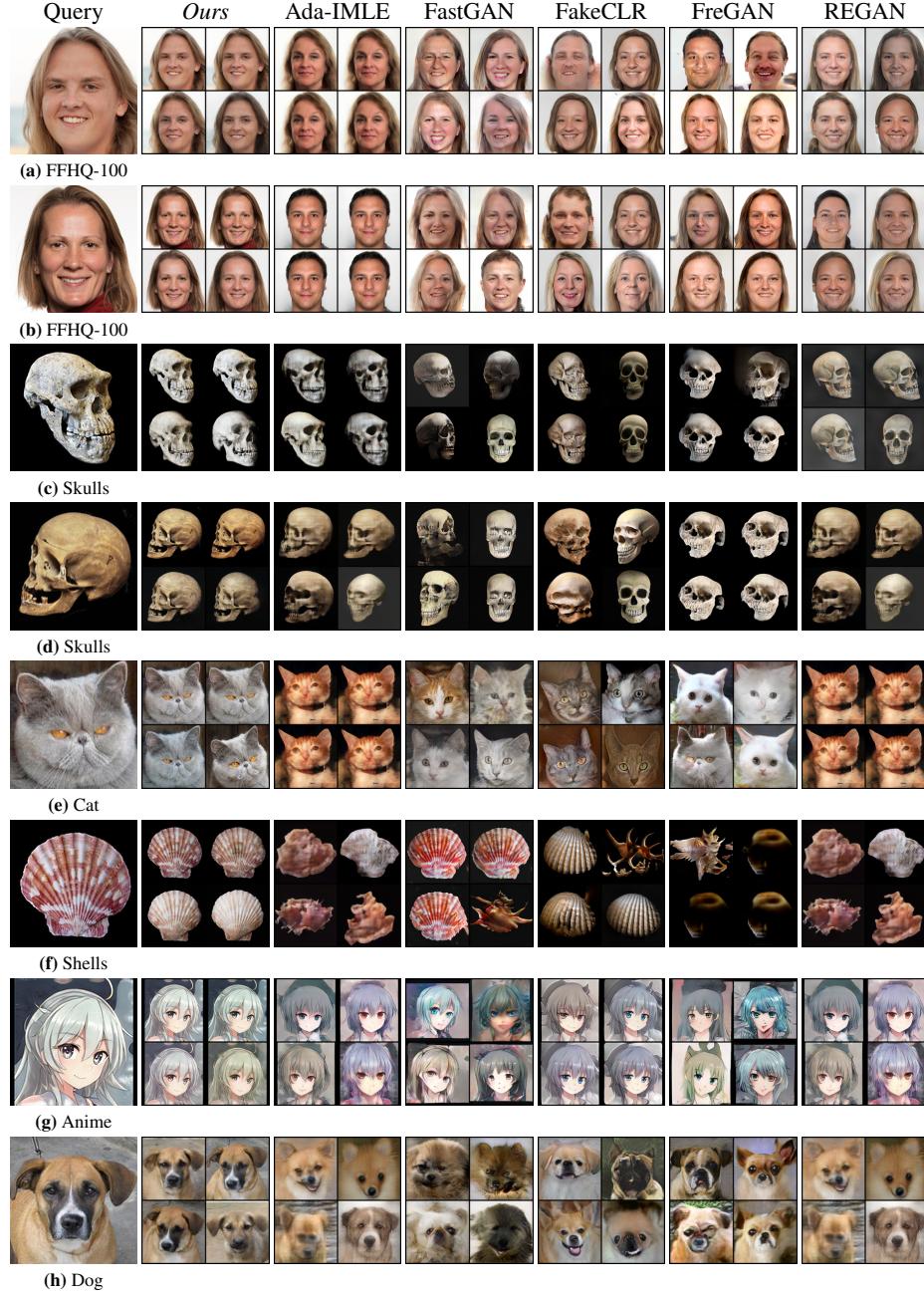


Fig. 11: Visual Recall Test: First column is the query image from the dataset. Subsequent columns are the samples produced by different methods that are closest to the query image in LPIPS feature space. The samples produced by our method are closer to the query images compared to the baselines, while being sufficiently diverse.