Understanding Hallucinations in Diffusion Models through Mode Interpolation

Sumukh K Aithal[1] **Pratyush Maini**[1,2] **Zachary C. Lipton**[1] **J. Zico Kolter**[1]
Carnegie Mellon University[1] DatologyAI[2]
{saithal, pratyus2, zlipton, zkolter}@cs.cmu.edu

Abstract

Colloquially speaking, image generation models based upon diffusion processes are frequently said to exhibit "hallucinations"—samples that could never occur in the training data. But where do such hallucinations come from? In this paper, we study a particular failure mode in diffusion models, which we term *mode interpolation*. Specifically, we find that diffusion models smoothly "interpolate" between nearby data modes in the training set to generate samples that are completely outside the support of the original training distribution; this phenomenon leads diffusion models to generate artifacts that never existed in real data (i.e., hallucinations). We systematically study the reasons for, and the manifestation of this phenomenon. Through experiments on 1D and 2D Gaussians, we show how a discontinuous loss landscape in the diffusion model's decoder leads to a region where any smooth approximation will cause such hallucinations. Through experiments on artificial datasets with various shapes, we show how hallucination leads to the generation of combinations of shapes that never existed. We extend the validity of mode interpolation in real-world datasets by explaining the unexpected generation of images with additional or missing fingers similar to those produced by popular text-to-image generative models. Finally, we show that diffusion models in fact *know* when they go out of support and hallucinate. This is captured by the high variance in the trajectory of the generated sample towards the final few backward sampling steps. Using a simple metric to capture this variance, we can remove over 95% of hallucinations at generation time while retaining 96% of in-support samples in the synthetic datasets. We conclude our exploration by showing the implications of such hallucination (and its removal) on the collapse (and stabilization) of recursive training on synthetic data with experiments on MNIST and a 2D Gaussians dataset. We release our code at https://github.com/locuslab/diffusion-model-hallucination.

1 Introduction

The high quality and diversity of images generated by diffusion models [15, 38] have made them the de facto standard generative models across various tasks including video generation [6], image inpainting [24], image super-resolution [11], data augmentation [44], and others. As a result of their uptake, large volumes of synthetic data are rapidly proliferating on the internet. The next generation of generative models will likely be exposed to many machine-generated instances during their training, making it crucial to understand ways in which diffusion models fail to model the true underlying data distribution. Like other generative model families, much research has been done to understand the failure modes of diffusion models as well. Past works have identified, and attempted to explain and remedy failures such as, training instabilities [17], memorization [7, 39] and inaccurate modeling of objects such as hands and legs [4, 23, 28].

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-4278

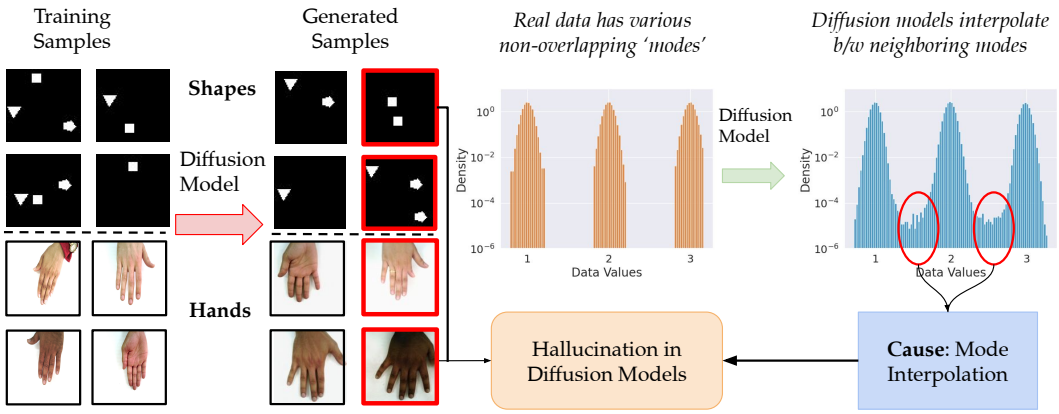

Figure 1: **Hallucinations in Diffusion Models**: Original Dataset (Left) & Generated Dataset (Right). (Top) The original dataset consists of 64x64 images divided into three columns, each containing a triangle, square, or pentagon with a 0.5 probability of the shape being present. Each shape appears at most once per image. The generated dataset created using an unconditional DDPM includes some samples (*hallucinations*) with multiple occurrences of the same shape that is unseen in the original dataset. (Bottom) We also train a ADM [29] on a dataset of high-quality images of human hands and show that the diffusion model generates hallucinated images of hands with additional fingers.

In this work, we formalize and study a particular failure mode of diffusion models that we call hallucination—a phenomenon where diffusion models generate samples that lie completely out of the support of the training distribution of the model. As a contemporary example, hallucinations manifest in large generative models like StableDiffusion [33] in the form of hands with extra (or missing) fingers or limbs. We begin our investigation with a surprising observation that an unconditional diffusion model trained on a distribution of simple shapes, generates images with combinations of shapes (or artifacts) that never existed in the original training distribution (Figure 1). While extensive research on generative models has focused on the phenomenon of 'mode collapse' [48], which leads to a loss of diversity in the sampled distribution, such studies often overlook the complex nature of real data which typically comprise multiple distinct modes on a complex data manifold, and the effects of their mutual interactions are thus neglected. In our work, we explain hallucinations by introducing a novel phenomenon we term 'mode interpolation' that considers this mutual interaction.

To understand the cause of these hallucinations and their relationship to mode interpolation, we construct simplified 1-d and 2-d mixture of Gaussian setups and train diffusion models on them (§ 4). We observe that when the true data distribution occurs in disjoint modes, diffusion models are unable to model a true approximation of the underlying distribution. This is because there exist 'step functions' between different modes, but the score function learned by the DDPM is a smooth approximation of the same, leading to interpolation between the nearest modes, even when these interpolated values are entirely absent from the training data. Moreover, we observation that hallucinated samples usually have very high variance towards the end of their trajectory during the reverse diffusion process. Based on this observation, we use the trajectory variance during sampling as a metric to detect hallucinations (§ 5), and show that diffusion models usually 'know' when they hallucinate, allowing detection with sensitivity and specificity > 0.92 in our experiments.

We explore mode interpolation as a potential explanation for the common failure of large-scale generative models, to accurately generate human hands. To demonstrate this concretely, we trained a diffusion model on a dataset of high-quality hand images and observed that it generated hands with additional fingers. We then applied our proposed metric to effectively detect these hallucinated generations. Finally, we study the implications of this phenomenon in recursive generative model retraining where we train generative models on their own output (§ 6). Recently, recursive training and its downsides in model collapse have garnered a lot of attention in both language and diffusion modeling literature [2, 3, 5, 9]. We observe that the proposed detection mechanism is able to mitigate the model collapse during recursive training on 2D Grid of Gaussians, Shapes and MNIST dataset.

1.1 Hallucination in Diffusion Models

Before formalizing our notions and definitions in § 3, let us first consolidate the observation that has been loosely labeled as 'hallucination' until now. To illustrate this phenomenon, we design a synthetic dataset called SIMPLE SHAPES, and train a diffusion model to learn its distribution.

SIMPLE SHAPES Setup. Consider a dataset consisting of black and white images that contain three shapes: triangle, square, and pentagon. Each image in the dataset is 64x64 pixels in size and divided into three (implied) columns. The first, second, and third columns contain a triangle, square, and pentagon, respectively. Each column has a 0.5 probability of containing the corresponding shape. A representation of this setup is shown in Fig 1. It is important to note that in this data generation pipeline, each shape is present at most once in each image.

Observation. We train an unconditional Denoising Diffusion Probabilistic Model (DDPM) [15] on this toy dataset with $T = 1000$ timesteps. We observe that the DDPM generates a small fraction of images that are never observed in the training dataset, nor a part of the 'support' of the data generation pipeline. Specifically, the model generates some images that contain two occurrences of the same shape, as shown in Fig 1. Furthermore, when the model is iteratively trained on its own sampled data, the fraction of these occurrences increases significantly as the generation process progresses.

Inspired by these observations and their implications, we will perform experiments through the rest of this work to formalize what we mean by hallucinations (§ 3), why do they occur (§ 4), how can we mitigate them (§ 5), and what are their implications for real-world datasets (§ 6).

2 Related Work

Diffusion Models. Diffusion models [15, 38, 43] are a class of generative models characterized by a forward process and a reverse process. In the forward process, noise is incrementally added to an image over time steps, ultimately converting the data into noise. The reverse process learns to denoise the image using a neural network essentially learning to convert noise to data. Diffusion models have various interpretations. Score-based generative modeling [41, 42] and DDPMs [15] are closely related, with [43] proposing a unified framework using stochastic differential equations (SDEs) that generalizes both Score Matching with Langevin Dynamics (SMLD) [43] and DDPM. In this framework, the forward process is a SDE with a continuous-time generalization instead of discrete timesteps and the reverse process is also an SDE that can be solved using a numerical solver. Another perspective is to view diffusion models as hierarchical Variational Autoencoders (VAEs) [25]. Recent research [19] suggests that diffusion models learn the optimal transport map between Gaussian distribution and data distribution. In this paper, we discover a surprising phenomenon in diffusion which we coin mode interpolation.

Recursive Generative Model Training. Recent works [2, 3, 26, 27, 37] demonstrated that iteratively training the generative models on their own output (i.e recursive training) leads to model collapse. The model collapse can happen in two ways: either all samples collapse to a single mode (low diversity) or the model generates very low fidelity, unrealistic images (low sample quality). This has been shown in the visual domain with StyleGAN2 and diffusion models [2, 3], as well as in the text domain with Large Language Models (LLMs) [5, 9, 37]. The current solution to mitigate this collapse is to include a fraction of real data in the training loop at all the generations [2, 3]. Theoretical results have also proved that super-quadratic number of synthetic samples are necessary to prevent model collapse [10] in the absence of support from real data. A concurrent work [12] studied the setup of data accumulation in recursive training where data from previous iterations of generative models together with real data are accumulated over time. The authors conclude that data accumulation (including real data) can avoid model collapse in various settings including language modeling and image data.

Past works have only studied the collapse of the generative model to the mode of the existing distribution. Through some controlled experiments, we study the interaction between different modes (a mode can be a class) or novel modes being developed in the generative models. This provides novel insights into the reasons behind the collapse of generative models during recursive training.

Failure Modes of Diffusion Models. One of the common failure modes of diffusion models is the generation of images where the hands and legs appear distorted or deformed which is commonly observed in Stable Diffusion [33] and Sora [6]. Diffusion models also fail to learn rare concepts [35]

 https://doi.org/10.52202/079017-4278

which have less than 10k samples in the training set. Various other failure modes including ignoring spatial relationships or confusing attributes have been discussed in [4, 23].

Hallucination in Language Models. Hallucination in LLMs [46, 47] is a huge barrier to the deployment of LLMs in safety-critical systems. The LLMs may provide a factually incorrect output or incorrectly follow the instructions or be logically wrong. A simple example is that LLMs can generate new facts when asked to summarize a block of text (input-conflicting hallucination) [47]. Current hallucination mitigation techniques in LLMs include factual data enhancement [13], retrieval augmentation [32] among other methods. Given the widespread adoption of image generation models, we argue that hallucination in diffusion models must also be studied carefully to identify its causes and mitigate it.

3 Definitions and Preliminaries

Let $q(x)$ be the real data distribution. We define a forward process where Gaussian noise is iteratively added at each timestep for a total of T timesteps. Let $x_0 \sim q(x)$, and x_t be the perturbed (noisy) sample after adding t timesteps of noise. The noise schedule is defined by $\beta_t \in (0, 1)$, which represents the variance of Gaussian (added noise) at time t. For large enough T, $x_T \sim \mathcal{N}(0, \mathbf{I})$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}); \qquad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{1}$$

In the forward diffusion process, we can directly sample x_t at any time step using the closed form $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$ where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{j=1}^{t} \alpha_j$.

The reverse diffusion process aims to learn the process of denoising i.e, learning $p_\theta(x_{t-1}|x_t)$ using a model (such as a neural network) with θ as the learnable parameters.

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t); \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)) \tag{2}$$

The mean can be derived as $\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right)$ where $\epsilon_\theta(\mathbf{x}_t, t)$ is the predict noise at timestep t using the neural network. The original DDPM is trained to predict the noise ϵ_t instead of x_t and the variance $\Sigma_\theta(\mathbf{x}_t, t)$ is fixed and time-dependent. Since then, improved methods have learned the variance [29]. We define predicted x_0 as $\hat{x_0} = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathbf{x}_t, t)\right)$

Connections to Score Based Generative Models. The score function $s(x)$ of a distribution $p(x)$ is the gradient of the log probability density function i.e, $\nabla_x \log p(x)$. The main premise of score-based generative modeling is to learn the score function of the data distribution given the samples from the same distribution. Once this score function is learned, annealed Langevin dynamics can be used to sample from the distribution using the formula $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t + \eta\nabla_\mathbf{x} \log p(\mathbf{x}) + \sqrt{2\eta}\mathbf{z}_t$, where η is the step size and z_t is sampled from standard normal. The score function can be obtained from the diffusion model using the equation $s_\theta(x_t, t) = -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}$ [45].

4 Understanding Mode Interpolation and Hallucination

In this section, we provide initial investigations into the central phenomenon of hallucinations in diffusion models. Formally, we consider a hallucination to be a generation from the model that lies entirely outside the support of the real data distribution (or, for distributions that theoretically have full support, in a region with negligible probability). That is, the ϵ-Hallucination set $H_\epsilon(q)$

$$H_\epsilon(q) = \{x : q(x) \leq \epsilon\}, \tag{3}$$

where we typically take $\epsilon = 0$ or take ϵ to be vanishingly small (well beyond numerical precision). We similarly define the ϵ-support set $S_\epsilon(q)$ to simply be the complement of the ϵ-Hallucination set.

Mode interpolation occurs when a model generates samples that directly *interpolate* (in input space) between two samples in the ϵ-support set, such that the interpolation lies in the ϵ-Hallucination set. That, is for $x, y \in S_\epsilon(q)$ the model generates $\theta x + (1-\theta)y \in H_\epsilon(q)$. The main argument of this

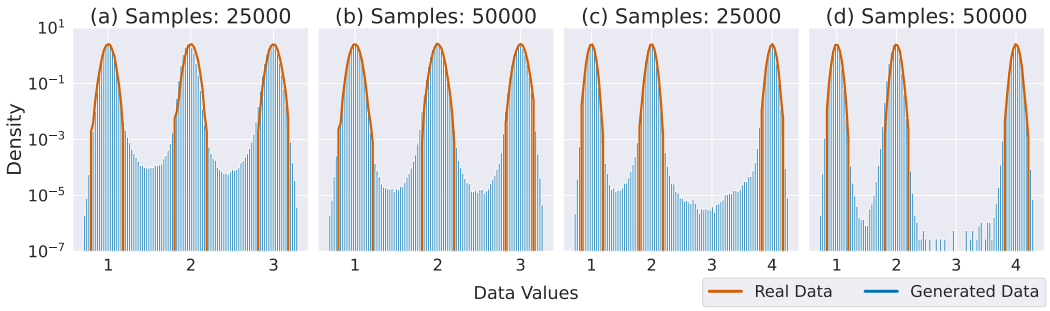

Figure 2: **Mode Interpolation in 1D GAUSSIAN**. The red curve indicates the PDF of the true data distribution $q(x)$, which is a mixture of 3 Gaussians (notice that the y-axis is in log-scale). In blue, we show a density histogram of the samples generated by a DDPM trained on varying number of samples from the true data distribution. For each histogram, we sampled 100 million examples from the diffusion model to observe the interpolated distribution. **(a,b)** show how the density of samples generated in the interpolated region reduces with an increase in the number of samples from the real distribution (used for training the DDPM). **(c,d)** show the impact of moving one of the modes (originally at $\mu = 3$) to $\mu = 4$. We see how the density of samples generated in the region between distant (but neighboring) modes is significantly lesser than that between nearby modes.

paper, shown through examples and numerical analysis of special cases, is that diffusion models frequently exhibit mode interpolation between "nearby" modes in the data distributions, and such interpolation leads to the generation of artifacts that did not exist in the original data (hallucinations).

4.1 1D GAUSSIAN Setup

We have already seen how hallucinations manifest in the SIMPLE SHAPES set-up (§ 1.1). To investigate hallucinations via mode interpolation, we begin with a synthetic toy dataset characterized by a mixture of 1D Gaussians given by: $p(x) = \frac{1}{3}\mathcal{N}(\mu_1, \sigma^2) + \frac{1}{3}\mathcal{N}(\mu_2, \sigma^2) + \frac{1}{3}\mathcal{N}(\mu_3, \sigma^2)$. For our initial experiments, we set $\mu_1 = 1, \mu_2 = 2, \mu_3 = 3$ and $\sigma = 0.05$. We sample 50k training points from this true distribution and train an unconditional DDPM using these samples with $T = 1000$ timesteps for $10,000$ epochs. Additional experimental details are present in the Appendix A.

We observe that diffusion models can generate samples that interpolate between the two nearest modes of the mixture of Gaussians (Figure 2). To clearly observe the distribution of these interpolated samples, we generated 100 million samples from the diffusion models. The probability of sampling from the interpolated regions (regions outside the support of the real data density, outlined in red) is non-zero, and decays with the distance from the modes. This region has nearly 0 probability mass of the true distribution, and no samples in this region occurred in the data used to train the DDPM.

The rate of mode interpolation depends primarily on three factors: **(i)** Number of training data points, **(ii)** variance of (and distance between) the distributions, and **(iii)** the number of sampling timesteps (T). As the number of training samples increases, we observe that the proportion of interpolated samples decreases. In this setup, the variance of $p(x)$ not only depends on σ but also the distance between the modes i.e, $|\mu_1 - \mu_2|$ and $|\mu_2 - \mu_3|$. We run another experiment with $\mu_1 = 1, \mu_2 = 2$ and $\mu_3 = 4$. In this case, we observe that the frequency of samples between μ_2 and μ_3 is much lower than μ_1 and μ_2. The number of interpolated samples also decreases as the distance from the modes increases. The frequency of interpolated samples is also inversely proportional to the number of timesteps T. Additional experiments with varying Gaussian counts are in Appendix C.

4.2 2D GAUSSIAN Grid

The reduction in density of mode interpolation as two modes with $\mu = [2, 3]$ are moved apart calls for closer inspection into when and how diffusion models choose to interpolate between nearby modes. To investigate this, we make a toy dataset with a mixture of 25 Gaussians arranged in a two-dimensional square grid. A total of 100,000 samples are present in the training set. Similar to the 1D case, we observe interpolated samples between the two nearest modes of the Gaussian. Again, these samples have close to zero probability if sampled from the original distribution (Figure 3).

https://doi.org/10.52202/079017-4278

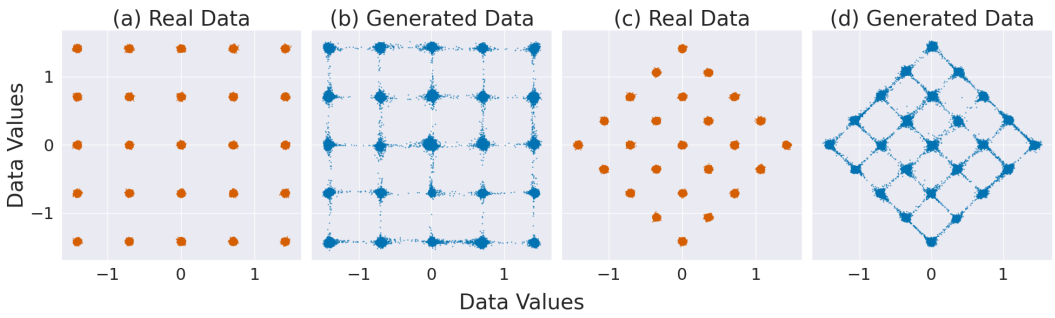

Figure 3: **Mode Interpolation in 2D GAUSSIAN**. The dataset consists of a mixture of 25 Gaussians arranged in a square grid, with a training set containing 100,000 samples. **(a,b)** The blue points represent samples generated by a DDPM, with visible density between the nearest modes of the original Gaussian mixture (in orange). These interpolated samples have near-zero probability in the original distribution. **(c,d)** We trained a DDPM on a rotated version of the dataset where the modes form a diamond shape. In this configuration, we see no interpolation along the x-axis, illustrating that diffusion models interpolate between nearest modes.

Figure 4: **Explaining Mode Interpolation via Learned Score Function.** The left panel shows the ground truth score function for a mixture of Gaussians across various timesteps, while the right panel illustrates the score function learned by the neural network. While the true score function exhibits sharp jumps that separate distinct modes (particularly in the initial time steps), the neural network approximates a smoother version.

We note that mode interpolation only happens between the nearest neighbors. To demonstrate this occurrence, we also train a DDPM on the rotated version of the dataset where the modes are arranged in the shape of a diamond (Figure 3.c,d). The mode interpolation can be more clearly observed in this setting. Interestingly, there appears to be no interpolation between modes along the x-axis, indicating that only the nearest modes are being interpolated. We believe this empirical observation of mode interpolation being confined to nearby modes will spark further investigation in future research.

4.3 What causes mode interpolation?

To understand the reason behind the observed mode interpolation, we analyze the score function learned by the model. The model learns to predict ϵ_θ which is related to the score function as $s_\theta(x_t, t) = -\frac{-\epsilon_\theta(x_t,t)}{\sqrt{1-\bar{\alpha}_t}}$. We know the true score function for the given mixture of Gaussians, and we can estimate the learned score function using the model's output. In Figure 4, we plot the ground truth score (left) and the learned score (right) across various timesteps. We observe that the neural network learns a smooth approximation of the true score function, particularly around the regions between disjoint modes of the distribution from timesteps $t = 0$ to $t = 20$. Notice that the true score function has sharp jumps that separate two modes, however, the neural network can not learn such sharp functions and smoothly approximates a tempered version of the same. We also plot the estimated $\hat{x_0}$ and observe a smooth approximation of the step function instead of the exact step function. There is a region of uncertainty in the region between the two modes which leads to mode interpolation

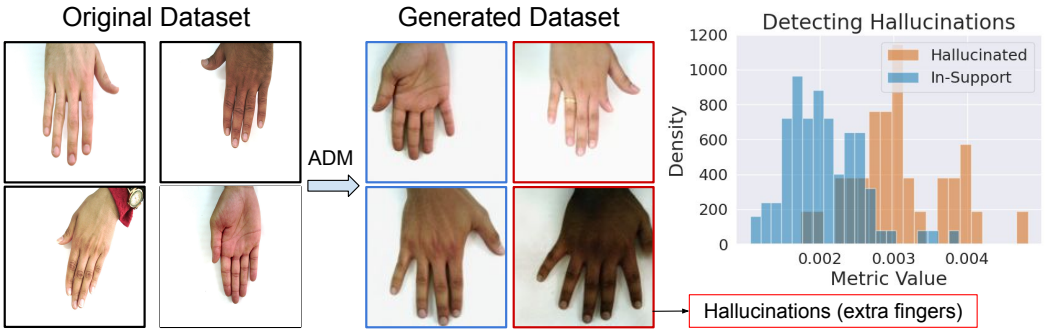

Figure 5: **Hands Dataset**. We train a ADM on the Hands dataset with 5000 images (first column) and show that the generated samples (second column) consists of hallucinated samples (additional/missing fingers). We then apply our proposed metric to detect these hallucinated samples (third column).

i.e sampling in the regions between the two modes. As another sanity check, we used the true score function in the reverse diffusion process for sampling (instead of the learned network). In this case, we did not see any instance of mode interpolation. This explains why the diffusion model generates samples between two modes of a Gaussian when it was never in the training distribution.

4.4 Simple Shapes

We now discuss the mode interpolation in the Simple Shapes dataset. In this context, the interpolation is not happening in the output space, but rather in the representation space. To investigate this, we performed a t-SNE visualization of the outputs from the bottleneck layer of the U-Net used in the Simple Shapes experiment, as shown in Figure 10 . Regions 1 and 3 in the representation space semantically correspond to the images where squares are at the top and bottom of the image respectively. At inference time, we can see a clear emergence of region 2 which is between regions 1 and 3 (interpolated), and contains two squares (hallucinations) at the top and bottom of the image. This experiment concretely confirms that interpolation happens in representation space.

4.5 Mode Interpolation in Real World datasets: Hands

We sought to demonstrate the occurrence of mode interpolation in a real-world setting. A well-documented challenge with popular text-to-image generative models is their difficulty in accurately generating human hands [28]. Despite extensive research in modern diffusion models, there is no conclusive explanation for the missing/additional fingers generated by these models. One hypothesis attributes this difficulty to the anatomical complexity of human hands, which involve numerous joints, fingers, and diverse poses. Another hypothesis suggests that, although large datasets contain many images of hands, these hands are often partially obscured (e.g., when a person is holding a cup) and occupy only a small region of the overall image.

To investigate this further, we trained a diffusion model on a datasets with high-quality images of human hands. The Hands dataset [1] consists of high resolution images of hands from 190 subjects of various ages. Each subject's right and left hands were photographed while opening and closing fingers against a uniform white background. We sample 5000 images from the Hands dataset and train an ADM [8] model on this dataset. We resize the images to 128x128 and use the same hyperparameters as that of the FFHQ dataset [18]. We mention all the hyperparameters in the Appendix A. We observe images with additional and missing fingers in the generated samples as seen in Figure 5. This is a pretty surprising result as it is non-trivial to assume that diffusion model generates images with additional fingers. Despite the potential for various failure modes, such as blurred hand images, these issues were not observed in our results. In some ways, the occurrence of 6-8 fingers is analogous to the occurrence of 2 squares in the Simple Shapes dataset. Thus, the presence of additional fingers in these images (i.e hallucinated images) generated by the diffusion model demonstrates the phenomenon of mode interpolation in real-world datasets. More example are shown in Fig. 20 & 21.

5 Diffusion Models know when they Hallucinate

Our previous sections established that hallucinations in diffusion models arise during sampling. More specifically, intermediate samples land in regions between different modes where the score

https://doi.org/10.52202/079017-4278

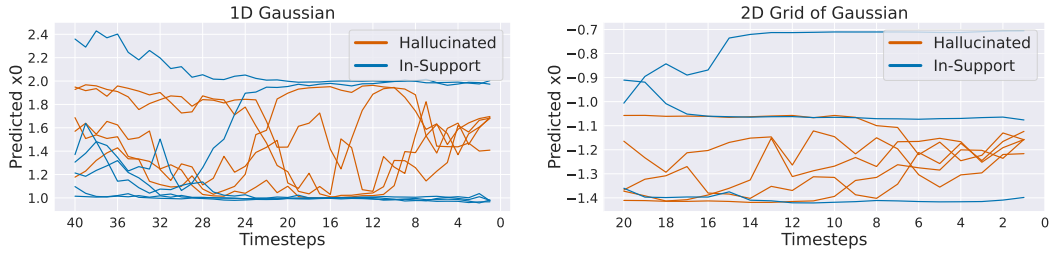

Figure 6: **Variance of $\hat{x_0}$ Trajectories**. The trajectory of the predicted $\hat{x_0}$ for hallucinated (shades of red), and non-hallucinated samples (shades of blue). We see that non-hallucinated samples stabilize in their prediction in the last 20 time steps for both 1D GAUSSIAN and 2D GAUSSIAN setups, whereas the hallucinated samples have high variance in the predicted $\hat{x_0}$ across time steps.

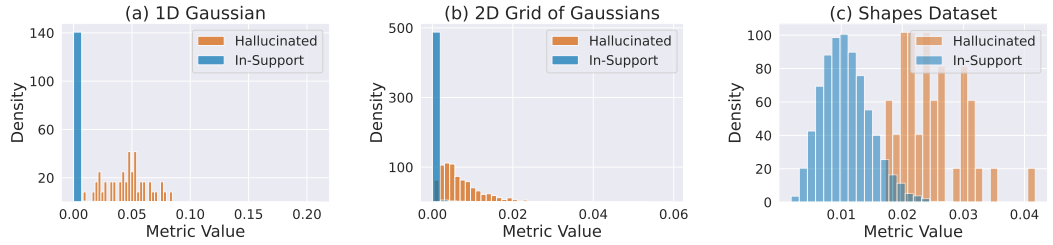

Figure 7: **Histogram of Hallucination Metric**. We depict the hallucination metric values for (a) 1D GAUSSIAN, (b) 2D GAUSSIAN, and (c) SIMPLE SHAPES setups. The histograms show that trajectory variance can capture a separation between hallucinated (orange) and non-hallucinated (blue) samples.

function has high uncertainty. Since neural networks find it hard to learn discrete 'jumps' between different modes (or a perfect step function), they end up interpolating between different modes of the distribution. This understanding suggests that the trajectory of the samples that generate hallucinations must have high variance due to the highly steep score function in the region of uncertainty. We will build upon this intuition to identify hallucinations in diffusion models.

5.1 Variance in the trajectory of prediction

We revisit the hallucinated samples in the 1D GAUSSIAN setup, and examine the trajectory of the predicted value of $\hat{x_0}$ during the reverse diffusion process. Figure 6 depicts the variance of trajectories leading to hallucinations (red shades) and those generating samples within the original data distribution (blue shades). For trajectories in shades of blue (non-hallucinations), the variance remains low beyond timestep $t = 20$. This indicates there is a minimal change in the predicted $\hat{x_0}$ during the final stages of reverse diffusion, signifying convergence. Conversely, the red trajectories (hallucinations) exhibit instability in the value of $\hat{x_0}$ in the same region. This suggests a high overall variance in these trajectories.

5.2 Metric for detecting hallucination

Based on the above observation about high variance in predicted values of x_0 in the reverse diffusion process, we use the same observation as a metric to distinguish hallucinated and non-hallucinated (in-support) samples. The intuition behind the metric is to capture the variance in the trajectory of $\hat{x_0}$. Let T_1 be the starting timestep and T_2 be the end timestep. Mathematically, the metric can be defined as follows:

$$\texttt{Hal}(x) = \frac{1}{|T_2 - T_1|} \sum_{i=T_1}^{T_2} \left(\hat{x_0}^{(i)} - \overline{\hat{x_0}^{(t)}} \right)^2 \tag{4}$$

where $\hat{x_0}^{(t)}$ represents the predicted values of the final image at different time steps (t), and $\overline{\hat{x_0}^{(t)}}$ is the mean of these predictions over the same time steps. We now utilize this metric to analyze the histogram values of each sample from the three experimental setups studied thus far. This metric can be implemented in two ways. One approach is to store $\hat{x_0}$ during the reverse diffusion process and then compute the variance. Alternatively, we explore a method where forward diffusion is performed for k steps between T_1 and T_2, predicting $\hat{x_0}$ at each step, and then computing the variance.

SIMPLE SHAPES. In the SIMPLE SHAPES setup, a sample is labeled as hallucinated if more than one shape of the same type occurs in the generated image. We generate 7500 images using a DDPM and study the separation between hallucinated and non-hallucinated images. We find that the reverse diffusion process of $T = 1000$ steps is rather long. Generally, the image stabilizes around $T = 700$ (as shown in Appendix 18). Therefore, we use the time range between $T = 850$ and $T = 700$ in the reverse diffusion process to compute the variance of the predicted sample value. Using this process, we can filter out 95% of the hallucinated samples while retaining 95% of the in-support samples. The histogram for the values is presented in Figure 7.

1D GAUSSIAN. In the 1D-Gaussian setup, we label any examples as a hallucination if they have negligible probability (for instance values greater than 6σ from the mean under normal) under the real data distribution (refer to Figure 2). We measure the variance of the last 15 steps of the $\hat{x_0}$ during the reverse diffusion process, and plot the histogram of values of the same in Figure 7. We can filter out 95 % of the hallucinated samples while retaining 98% of the in-support samples.

2D GAUSSIAN. Next, we discuss our investigation on synthetic datasets with experiments on the 2D GAUSSIAN dataset. Similar to the 1D GAUSSIAN setup, we once again measure the prediction variance of the last 20 steps of the reverse diffusion process. We compute the variance per dimension and then take the mean across dimensions to . With this metric, we can filter out 96% of the hallucinated samples while retaining 95% of the in-support samples.

HANDS. Finally, we conclude our investigation with experiments on the Hands dataset. To analyze the effectiveness of the proposed metric, we manually label 130 images from the generated samples as hallucinated vs. in-support. This includes 88 images with 5 fingers and 40 images with missing/ additional fingers i.e. hallucinated samples. The histogram (in Figure 5) shows that the proposed metric can indeed detect these hallucinations to a reasonable degree. In our experiments, we observe that we can eliminate 80% of the hallucinated samples while retaining 81% of the in-support samples. The trajectories of the hallucinated and in-support samples are shown in Figures 22 and 23, respectively. A higher variance in the trajectory of $\hat{x_0}$ is clearly observed in the hallucinated samples compared to the in-support samples. We note that the detection is a hard problem and the fact that the method transfers to the real world is proof of the relationship between mode interpolation and hallucination in real-world data.

6 Implications on Recursive Model Training

The internet is increasingly populated by more and more synthetic data (data synthesized from generative models). It is likely that future generative models will be exposed to large volumes of machine-generated data during their training [26, 27]. Recursive training on synthetic data leads to mode collapse [2, 9] and exacerbates data biases. In this section, we study the impact of hallucinations within the context of recursive generative model training. We adopt the standard synthetic-only setup similar to [2] where we only use synthetic data from the current generative model in training the next generation of generative models. The first generation of generative model is trained on real data and samples from this generative model is used to train the second generation (and so on).

Most of the previous works [3] studied the model collapse to a single mode. In this work, we emphasize that the interaction between modes and mode interpolation plays a massive role when training generative models on their own output.

2D GAUSSIAN. When we recursively train a DDPM on its own generated data using a square grid of 2D Gaussians (with $T = 500$), the hallucinated samples significantly influence the learning of the next generation's distribution (see Figure 9). The frequency of the interpolated samples increases as we further train on the learned distribution that consists of interpolated samples. Figure 9d shows samples from Generation 20, where it is evident that the modes have almost collapsed into a single mode, differing greatly from the original data distribution.

https://doi.org/10.52202/079017-4278

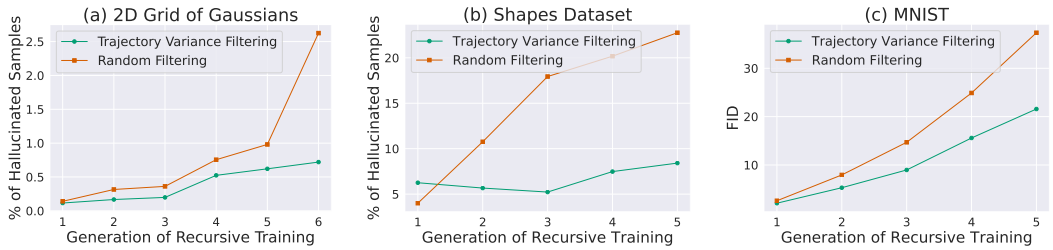

Figure 8: **Mitigating Hallucinations with Pre-emptive Detection**. We filter out hallucinated samples using the metric from § 5 before training on samples from the previous generation of the diffusion model. In the case of (a) 2D GAUSSIAN, (b) SIMPLE SHAPES, where we have clear definitions of hallucination (mode interpolation, and new shape combinations) we see the effectiveness of our variance-based filtering method in minimizing hallucinations across generations compared to random filtering. In the case of (c) MNIST dataset, we measure the FID of subsequent generations and notice that pre-emptive filtering of hallucinated samples makes the recursive model collapse slower.

SIMPLE SHAPES. We define a hallucinated sample as one that contains at least two shapes of the same type (which is never seen in the training distribution). We observe the presence of around 5% hallucinated samples when trained on the real data. We note that the ratio of hallucinated samples increases exponentially as the we iteratively train the diffusion model on its own data. This is expected as the diffusion model progressively learns from a distribution increasingly dominated by hallucinated images, compounding the effect in subsequent generations.

MNIST. We also run the recursive model training on the MNIST dataset [22]. At every generation, we generate 65k images and sample 60k images using the filtering mechanism. For each generation, we train a class conditional DDPM with Classifier-Free Guidance [16] with $T = 500$ for 50 epochs. To evaluate the quality of the generated images, we compute the FID [14] using a LeNet [22] trained on MNIST instead of Inception backbone as MNIST is not a natural image dataset. In Figure 8, we clearly see that the proposed metric based on the variance of the trajectory outperforms the random filtering method across all generations (lower FID is better). We also plot the Precision and Recall [36] curves (in the Appendix Figure 18) where we observe that our filtering mechanism selects high quality samples without much loss in diversity.

Mitigating the curse of recursion with pre-emptive detection of hallucinations. Based on the metric developed in § 5, we analyze the efficacy of the proposed metric in filtering out the hallucinated samples for the next generation of training. After training each generation of the generative model, we sample k images more than size of the training data and then filter out hallucinated samples based on the metric. Figure 8 shows the results on 2D Grid of Gaussians, SIMPLE SHAPES and MNIST dataset. We also compare with random filtering where we randomly sample points for the next generation. The variance-based filtering method easily outperforms the random sampling method in all the generations. We see the effectiveness of the proposed metric in minimizing the rate of hallucinations across generations and thus model collapse to a certain extent. This holds true for all the three datasets we have studied in this work.

7 Discussion

In this work, we performed an in-depth study to formulate and understand hallucination in diffusion models, focusing on the phenomenon of mode interpolation. We analyzed this phenomenon in four different settings: 1D Gaussian, 2D Grid of Gaussians, Shapes and Hands datasets, and saw how diffusion models learn smoothed approximations of disjoint score functions, leading to mode interpolation. Based on our analysis, we developed a metric to identify hallucinated samples effectively and explored the implications of hallucination in the context of recursive generative model training. This study is the first to propose mode interpolation as a potential hypothesis for explaining the generation of additional fingers in large-scale generative models. We hope that future research will build upon this hypothesis and develop methods to mitigate these issues in generative models. We hope our work inspires future research in understanding and mitigating hallucination in diffusion models.

Acknowledgements

PM is supported by funding from the DARPA GARD program. ZL gratefully acknowledges the NSF (FAI 2040929 and IIS2211955), UPMC, Highmark Health, Abridge, Ford Research, Mozilla, the PwC Center, Amazon AI, JP Morgan Chase, the Block Center, the Center for Machine Learning and Health, and the CMU Software Engineering Institute (SEI) via Department of Defense contract FA8702-15-D-0002, for their generous support of ACMI Lab's research. ZK gratefully acknowledges support from the Bosch Center for Artificial Intelligence to support work in his lab as a whole.

References

[1] M. Afifi. 11k hands: gender recognition and biometric identification using a large dataset of hand images. *Multimedia Tools and Applications*, 2019.

[2] S. Alemohammad, J. Casco-Rodriguez, L. Luzi, A. I. Humayun, H. Babaei, D. LeJeune, A. Siahkoohi, and R. G. Baraniuk. Self-consuming generative models go mad. *arXiv preprint arXiv:2307.01850*, 2023.

[3] Q. Bertrand, A. J. Bose, A. Duplessis, M. Jiralerspong, and G. Gidel. On the stability of iterative retraining of generative models on their own data. *arXiv preprint arXiv:2310.00429*, 2023.

[4] A. Borji. Qualitative failures of image generation models and their application in detecting deepfakes. *Image and Vision Computing*, 137:104771, 2023.

[5] M. Briesch, D. Sobania, and F. Rothlauf. Large language models suffer from their own output: An analysis of the self-consuming training loop. *arXiv preprint arXiv:2311.16822*, 2023.

[6] T. Brooks, B. Peebles, C. Holmes, W. DePue, Y. Guo, L. Jing, D. Schnurr, J. Taylor, T. Luhman, E. Luhman, C. Ng, R. Wang, and A. Ramesh. Video generation models as world simulators. 2024.

[7] N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramer, B. Balle, D. Ippolito, and E. Wallace. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 5253–5270, 2023.

[8] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[9] E. Dohmatob, Y. Feng, P. Yang, F. Charton, and J. Kempe. A tale of tails: Model collapse as a change of scaling laws. *arXiv preprint arXiv:2402.07043*, 2024.

[10] S. Fu, S. Zhang, Y. Wang, X. Tian, and D. Tao. Towards theoretical understandings of self-consuming generative models. *arXiv preprint arXiv:2402.11778*, 2024.

[11] S. Gao, X. Liu, B. Zeng, S. Xu, Y. Li, X. Luo, J. Liu, X. Zhen, and B. Zhang. Implicit diffusion models for continuous super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10021–10030, 2023.

[12] M. Gerstgrasser, R. Schaeffer, A. Dey, R. Rafailov, H. Sleight, J. Hughes, T. Korbak, R. Agrawal, D. Pai, A. Gromov, et al. Is model collapse inevitable? breaking the curse of recursion by accumulating real and synthetic data. *arXiv preprint arXiv:2404.01413*, 2024.

[13] S. Gunasekar, Y. Zhang, J. Aneja, C. C. T. Mendes, A. Del Giorno, S. Gopi, M. Javaheripi, P. Kauffmann, G. de Rosa, O. Saarikivi, et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

[14] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[15] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

 https://doi.org/10.52202/079017-4278

[16] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[17] Z. Huang, P. Zhou, S. Yan, and L. Lin. Scalelong: Towards more stable training of diffusion model via scaling network long skip connection. *Advances in Neural Information Processing Systems*, 36:70376–70401, 2023.

[18] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[19] V. Khrulkov, G. Ryzhakov, A. Chertkov, and I. Oseledets. Understanding ddpm latent codes through optimal transport. *arXiv preprint arXiv:2202.07477*, 2022.

[20] D. Kingma, T. Salimans, B. Poole, and J. Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

[21] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[23] Q. Liu, A. Kortylewski, Y. Bai, S. Bai, and A. Yuille. Intriguing properties of text-guided diffusion models. *arXiv preprint arXiv:2306.00974*, 2023.

[24] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11461–11471, 2022.

[25] C. Luo. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.

[26] G. Martínez, L. Watson, P. Reviriego, J. A. Hernández, M. Juarez, and R. Sarkar. Combining generative artificial intelligence (ai) and the internet: Heading towards evolution or degradation? *arXiv preprint arXiv:2303.01255*, 2023.

[27] G. Martínez, L. Watson, P. Reviriego, J. A. Hernández, M. Juarez, and R. Sarkar. Towards understanding the interplay of generative artificial intelligence and the internet. In *International Workshop on Epistemic Uncertainty in Artificial Intelligence*, pages 59–73. Springer, 2023.

[28] S. Narasimhaswamy, U. Bhattacharya, X. Chen, I. Dasgupta, S. Mitra, and M. Hoai. Handiffuser: Text-to-image generation with realistic hand appearances. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[29] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[30] M. Ning, E. Sangineto, A. Porrello, S. Calderara, and R. Cucchiara. Input perturbation reduces exposure bias in diffusion models. In *International Conference on Machine Learning*, pages 26245–26265. PMLR, 2023.

[31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[32] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, and Y. Shoham. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331, 2023.

[33] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[34] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[35] D. Samuel, R. Ben-Ari, S. Raviv, N. Darshan, and G. Chechik. Generating images of rare concepts using pre-trained diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 4695–4703, 2024.

[36] K. Shmelkov, C. Schmid, and K. Alahari. How good is my gan? In *Proceedings of the European conference on computer vision (ECCV)*, pages 213–229, 2018.

[37] I. Shumailov, Z. Shumaylov, Y. Zhao, Y. Gal, N. Papernot, and R. Anderson. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arXiv:2305.17493*, 2023.

[38] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[39] G. Somepalli, V. Singla, M. Goldblum, J. Geiping, and T. Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6048–6058, 2023.

[40] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[41] Y. Song, C. Durkan, I. Murray, and S. Ermon. Maximum likelihood training of score-based diffusion models. *Advances in neural information processing systems*, 34:1415–1428, 2021.

[42] Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[43] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[44] B. Trabucco, K. Doherty, M. Gurinas, and R. Salakhutdinov. Effective data augmentation with diffusion models. *arXiv preprint arXiv:2302.07944*, 2023.

[45] L. Weng. What are diffusion models? *lilianweng.github.io*, Jul 2021.

[46] H. Ye, T. Liu, A. Zhang, W. Hua, and W. Jia. Cognitive mirage: A review of hallucinations in large language models. *arXiv preprint arXiv:2309.06794*, 2023.

[47] Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen, et al. Siren's song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023.

[48] Z. Zhang, M. Li, and J. Yu. On the convergence and mode collapse of gan. In *SIGGRAPH Asia 2018 Technical Briefs*, pages 1–4. 2018.

 https://doi.org/10.52202/079017-4278

A Additional Experimental Details

A.1 Gaussian experiments

We run all our experiments for $10,000$ epochs with batch size of $10,000$. A linear noise schedule is used with starting noise $\beta_0 = 0.001$ and the final noise $\beta_1 = 0.2$. We use $T = 1000$ by default in our experiments (unless specified otherwise). The neural network (NN) is trained to predict the noise (similar to the original DDPM [15] implementation) and we use a Mean Squared Error loss to train the model. The input and output of the NN have the same shape (in this case, 1 for 1D Gaussian and 2 for the 2D Gaussian). The NN architecture starts with an initial fully connected layer, followed by three blocks and then output fully connected layer. Each block includes normalization, a LeakyReLU activation, and two fully connected layers. Finally, the output is normalized and transformed back to the input dimension with a fully connected layer. Adam [21] with learning rate of 0.001 is used as the optimizer. We build our codebase on top of [1] for the synthetic toy experiments.

Metric: We use $t = 0$ to $t = 15$ (last 15 steps in the reverse diffusion process) to compute the variance of the trajectory in the case of Gaussian 1D and $t = 0$ to $t = 8$ in the case of 2D Gaussian Grid.

A.2 Shapes

The generated images are grayscale images of size 64×64. A total of 5000 images is generated for training the diffusion model. We use a U-Net [34] architecture to model the reverse diffusion process. We use a cosine noise scheduler similar to ADM [29]. We derive our implementation based on [2] for training the DDPM. We train an unconditonal DDPM on the dataset with $T = 1000$ while training and 250 steps during sampling to reduce computation cost [40].

A.3 MNIST

MNIST [22] consists of 60,000 grayscale images of size (28, 28). We use classifier-free guidance [16] to train a conditional DDPM on MNIST with $T = 500$. For each generation, we train for a total of 50 epochs with a batch size of 512 shared across 4 GPUs. Adam [21] optimizer with learning rate of 1e-4 is used to train the network. We use a U-Net [34] with 256 feature dimension to model the reverse diffusion process. For the variance filtering mechanism in Section 6, we use 10 timesteps between $t = 100$ to $t = 150$ to compute the variance of the trajectory. In the case of MNIST, we do post-hoc filtering just using the samples. This means that we add t timesteps of noise, then compute $\hat{x_0}$ and then use this to compute variance.

Our implementations of the DDPM model is based on PyTorch [31].

Compute: We run all our experiments of Nvidia RTX 2080 Ti and Nvidia A6000 GPUs. The training and sampling for the Gaussian experiments takes less than 3 hours on single 2080Ti GPU. Sampling 100 million datapoints takes around 3-4 hours. Running DDPM on the shapes dataset takes around 6-7 hours with 4 2080Ti GPUs. The recursive generative training on MNIST takes about 16 hours with 4 A6000 GPUs for 5 generations.

A.4 HANDS

ADM refers to Ablated Diffusion model as defined in [8]. We trained for a total of 200k iterations with batch size 16 and a learning rate of 1e-4. The diffusion process was trained with 1000 timesteps ($T = 1000$) with a cosine noise schedule. The U-Net comprised 256 channels, with an attention mechanism incorporating 64 channels per head and 3 residual blocks. For sampling, we use 500 timesteps with respacing. We base our implementation and hyperparameters on the official DDPM-IP [30] repository.

[1] https://github.com/tqch/ddpm-torch

[2] https://github.com/VSehwag/minimal-diffusion

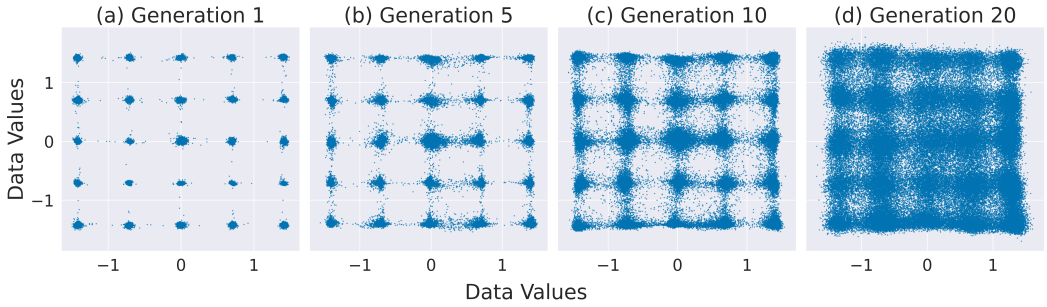

Figure 9: **Recursive Training on 2D GAUSSIAN**. We investigate the impact of recursively training a DDPM on its own generated data using a square grid of 2D Gaussians with $T = 500$ diffusion steps. In each generation, we sample 100k examples, and train the subsequent generation on these data points. As the training progresses through multiple generations, the hallucinated (interpolated) samples significantly influence the learning of the next generation's distribution.

B Limitations and Broader Impact

Hallucinations in LLMs have been studied extensively [46, 47] given the widespread use of these systems in various contexts. This work investigates hallucinations in diffusion models. In current generative models, these hallucinations could be used to more easily identify machine-generated images. Developing a metric to identify these hallucinations and remove them could make the detection of generated images much harder. However, we argue that understanding hallucinations in diffusion models is crucial as it can help shed light on their failure modes and thereby enable better control in practical applications.

In current text-to-image generative models, the poorly modeled "hands" are a clear giveaway in identification of AI generated images. The detection of such AI-generated content would be made much more difficult if these hallucinations were identified and removed from the generated images. While our work builds an understanding of hallucinations, and allows us to also detect them, we believe that future generations of models would have become more robust to such hallucinations by virtue of training on more data independent of this work.

Concerning the limitations of the proposed hallucination metric, the selection of the right timesteps is key to be able to detect hallucinations. More analysis on what region of trajectory leads to hallucinations would be useful across various schedules and sampling algorithms. We believe these are great areas for future work to explore. Additional explorations of mode interpolation and hallucinations in real-world datasets would be useful to the community.

C Additional Experiments and Figures

We also study Variational Diffusion Models (VDM) [20] to verify the generality of our findings. Our results show that the over-smoothed score function phenomenon persists in VDM, supporting the hypothesis that this issue is not specific to DDPM. We train a simple VDM on the 2D Gaussian with 10k samples. We follow the setup and hyperparameters in the official implementation [3]. We train both continuous and discrete variants of VDM on the 2D Gaussian dataset. The main observation is that VDM mitigates the hallucinations significantly especially with more training data but the phenomenon of mode interpolation still exists. In this figure, we also show the impact of the number of sampling steps on the count of hallucinations. We clearly see that increasing the number of sampling steps reduces the number of hallucinated samples. This can be clearly observed in Figure 11 (first two columns) where the count of hallucinations decreases mode interpolation.

The frequency of mode interpolation is inversely proportional to the number of training samples. We train the unconditional diffusion model with 25k, 50k, 100k and 500k samples from the true distribution.

[3] https://github.com/google-research/vdm

https://doi.org/10.52202/079017-4278

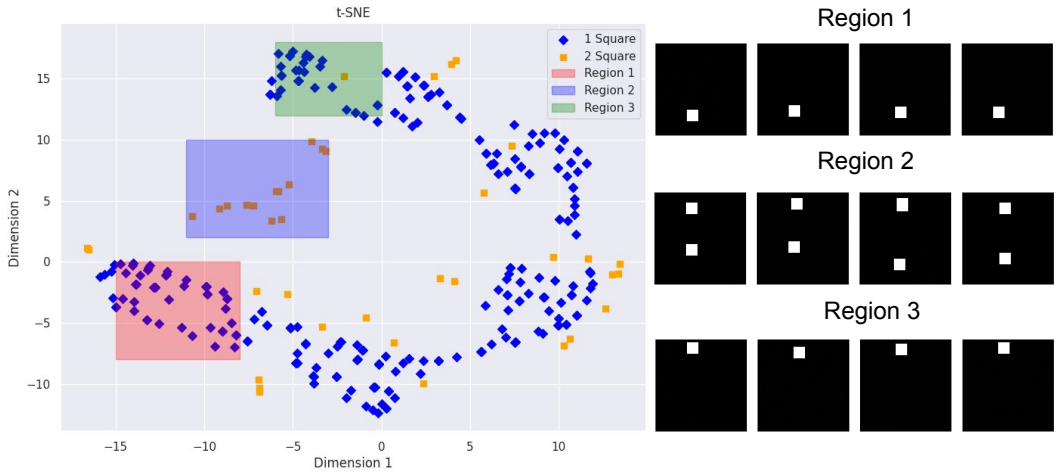

Figure 10: **Interpolation in Representation Space**. We analyze the bottleneck of the U-Net to demonstrate mode interpolation in the Shapes dataset. We clearly see that Region 2 (which consists of 2 squares) is interpolating between Region 1 (one square in the bottom half) and Region 3 (one square in the top half).

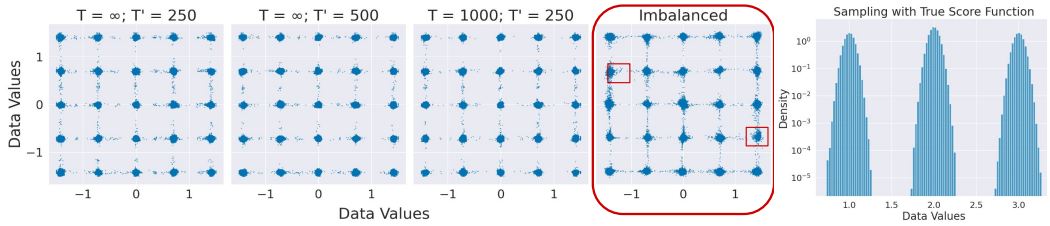

Figure 11: **Variational Diffusion Model**. We train a Variational Diffusion Model (VDM) on the 2D Gaussian Data with 10k samples (first three columns). T denotes the timesteps during training and T' denotes the sampling timesteps. $T = \infty$ refers to the continuous time variant. The fourth column shows a DDPM trained on a 2D Gaussian with imbalanced modes. The boxes indicate the modes with less data. The last column shows result of sampling from the true score function.

1. Figure 12 shows the histogram of samples generated by the diffusion model (with 10 million samples) when the model is trained on the distribution with $\mu_1 = 1, \mu_2 = 2, \mu_3 = 3$.
2. Figure 13 shows the histogram of samples generated by the diffusion model (with 10 million samples) when the model is trained on the distribution with $\mu_1 = 1, \mu_2 = 2, \mu_3 = 4$.
3. We also experiment with mixture of 2 Gaussians in Figure 14 and 4 Gaussians in Figure 15.
4. Figure 16 shows the FID, precision and recall curves for MNIST across generations.
5. Figure 17 shows additional examples of hallucinated images generated by the diffusion model.
6. Figure 18 shows the $\hat{x_0}$ across various timesteps for a hallucinated image. The number on top of the image indicates the timestep.
7. Figure 19 shows the $\hat{x_0}$ across various timesteps for a image in-support of the distribution. The number on top of the image indicates the timestep.

https://doi.org/10.52202/079017-4278

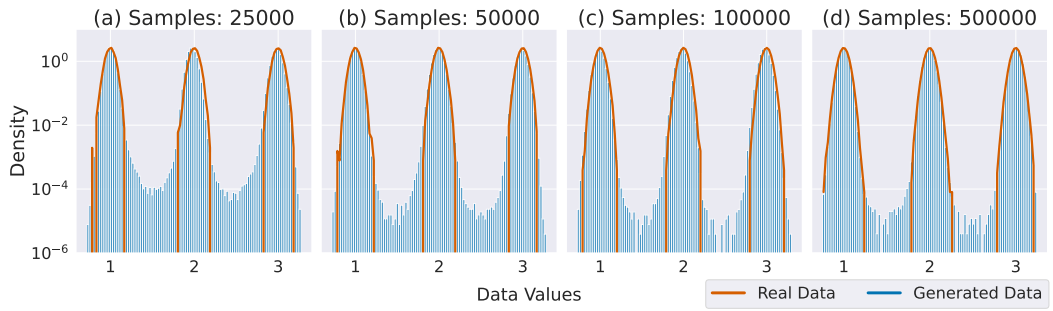

Figure 12: Mixture of 3 Gaussians with $\mu = [1, 2, 3]$. We vary the number of training samples and observe that mode interpolation decreases with increase in the size of training data

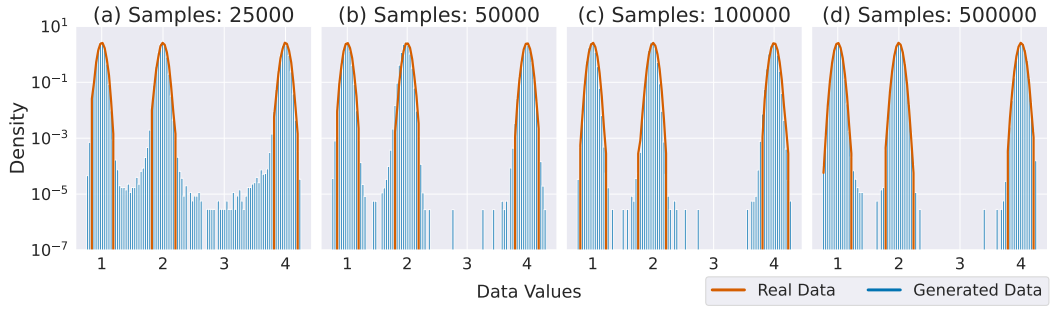

Figure 13: Mixture of 3 Gaussians with $\mu = [1, 2, 4]$. We vary the number of training samples and observe that mode interpolation decreases with increase in the size of training data.

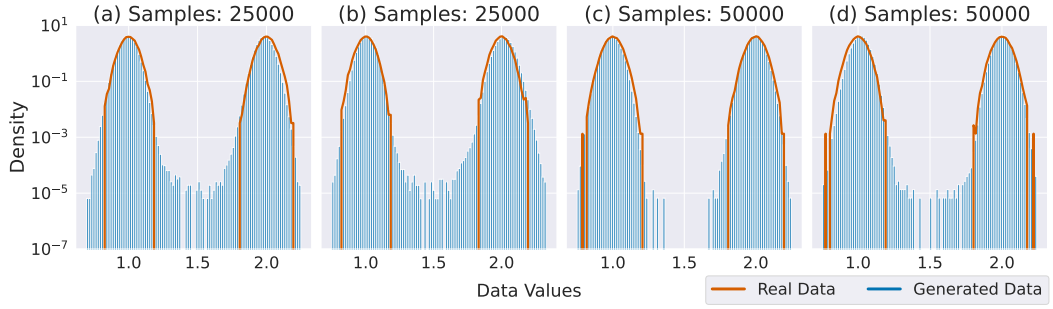

Figure 14: Mixture of 2 1D Gaussians with varying number of training samples. (a) and (b) have the same number of training samples but with two different seeds. Similarly for (c) and (d).

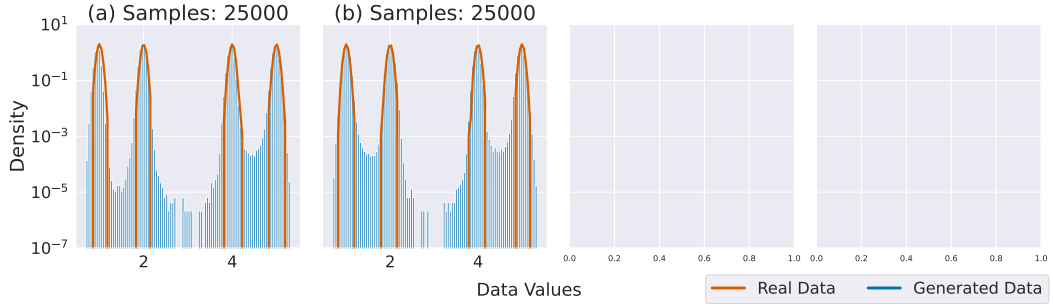

Figure 15: Mixture of 4 1D Gaussians ($\mu = [1, 2, 4, 5]$) with varying number of training samples. (a) and (b) have the same number of training samples but with two different seeds. We clearly see more samples in the region between modes μ_1 = 1 and $\mu_2 = 2$ compared to $\mu_2 = 2$ and $\mu_3 = 4$.

https://doi.org/10.52202/079017-4278

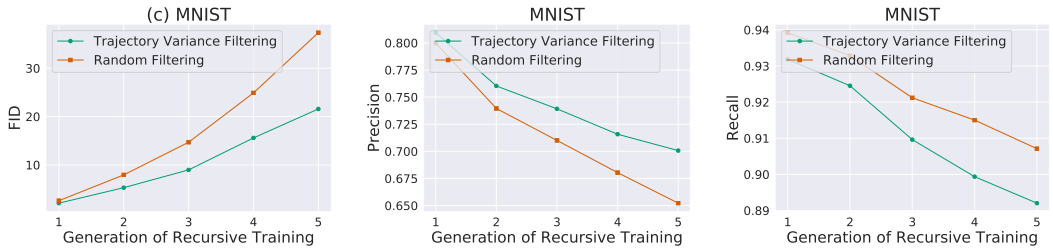

Figure 16: Recursive Generative Training on MNIST with Variance and Random Filtering. We observe that the proposed filtering mechanism can discard low quality samples while maintaining sufficient diversity.

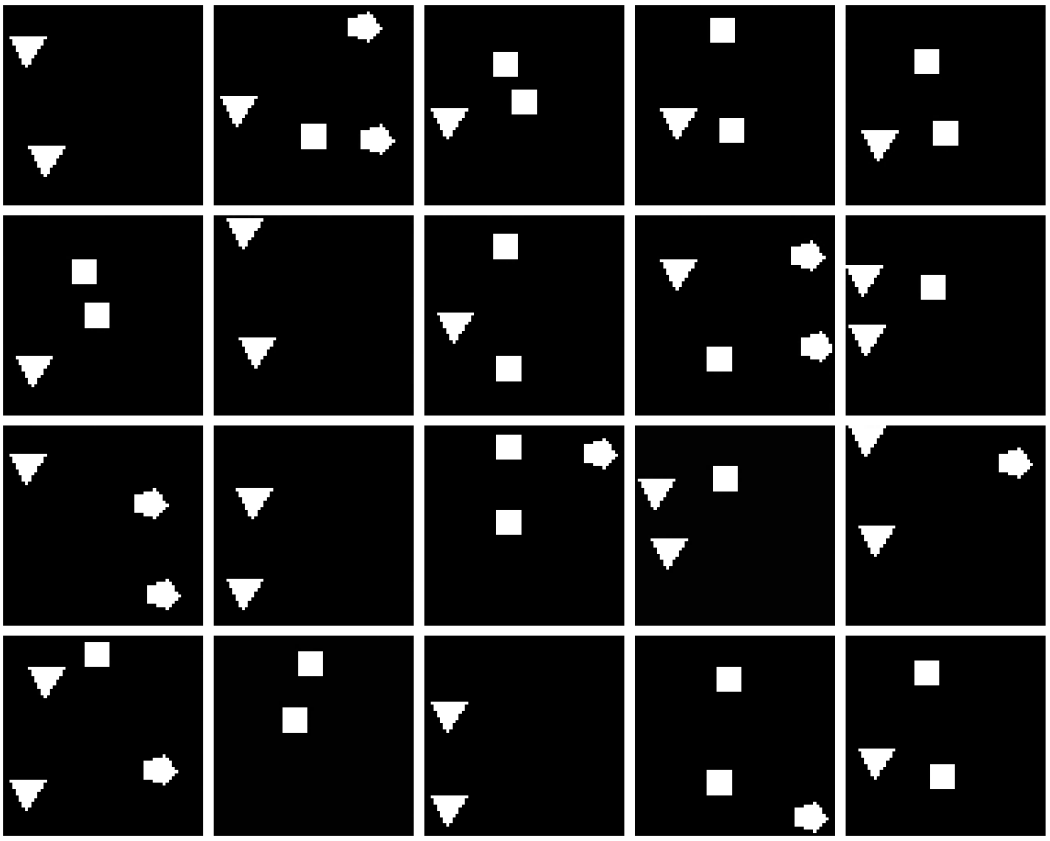

Figure 17: Example of Generated Hallucinated Images

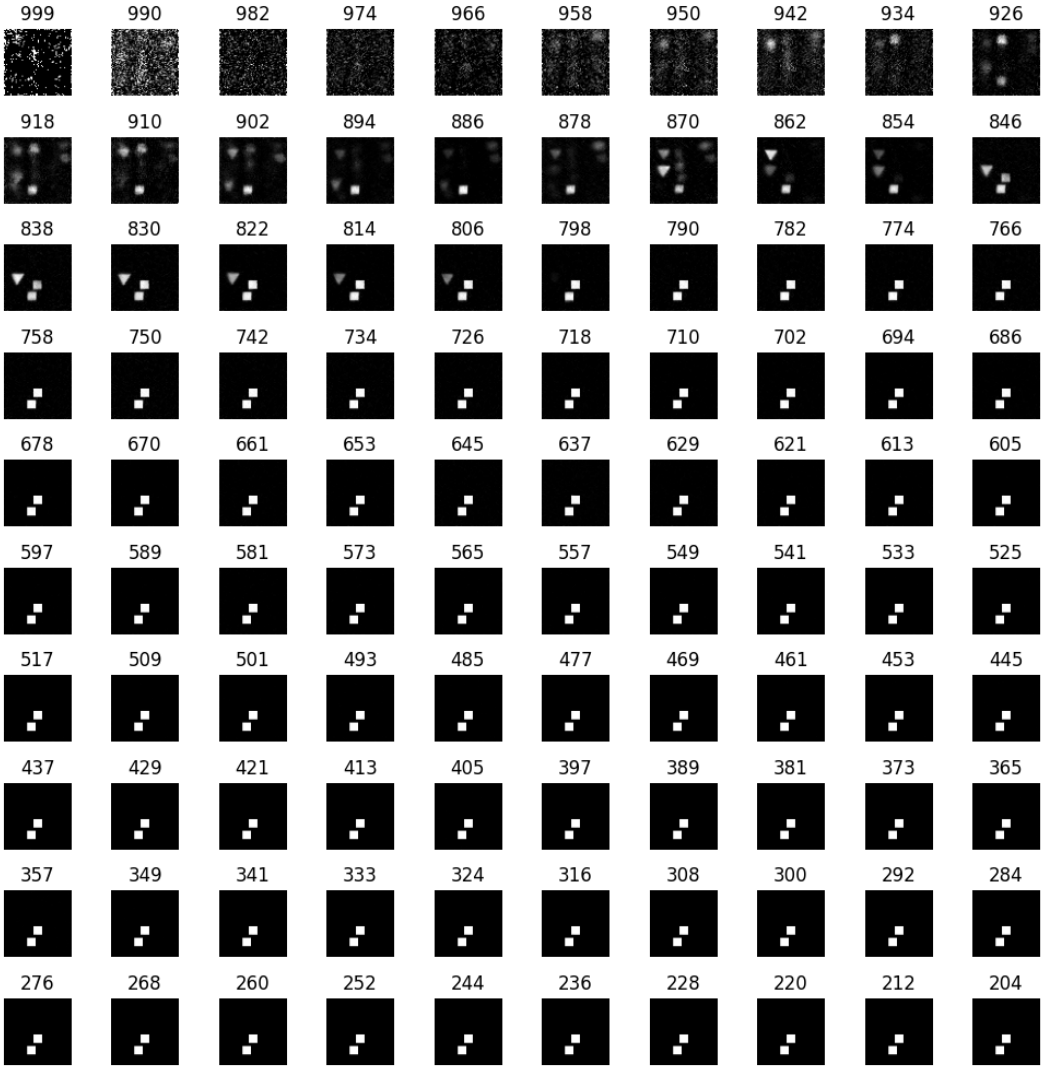

Figure 18: $\hat{x_0}$ for Hallucinated Sample. Here, we observe that the predicted x_0 has a lot of variance around $t = 700$ to $t = 850$. This clearly motivates our proposed metric.

https://doi.org/10.52202/079017-4278

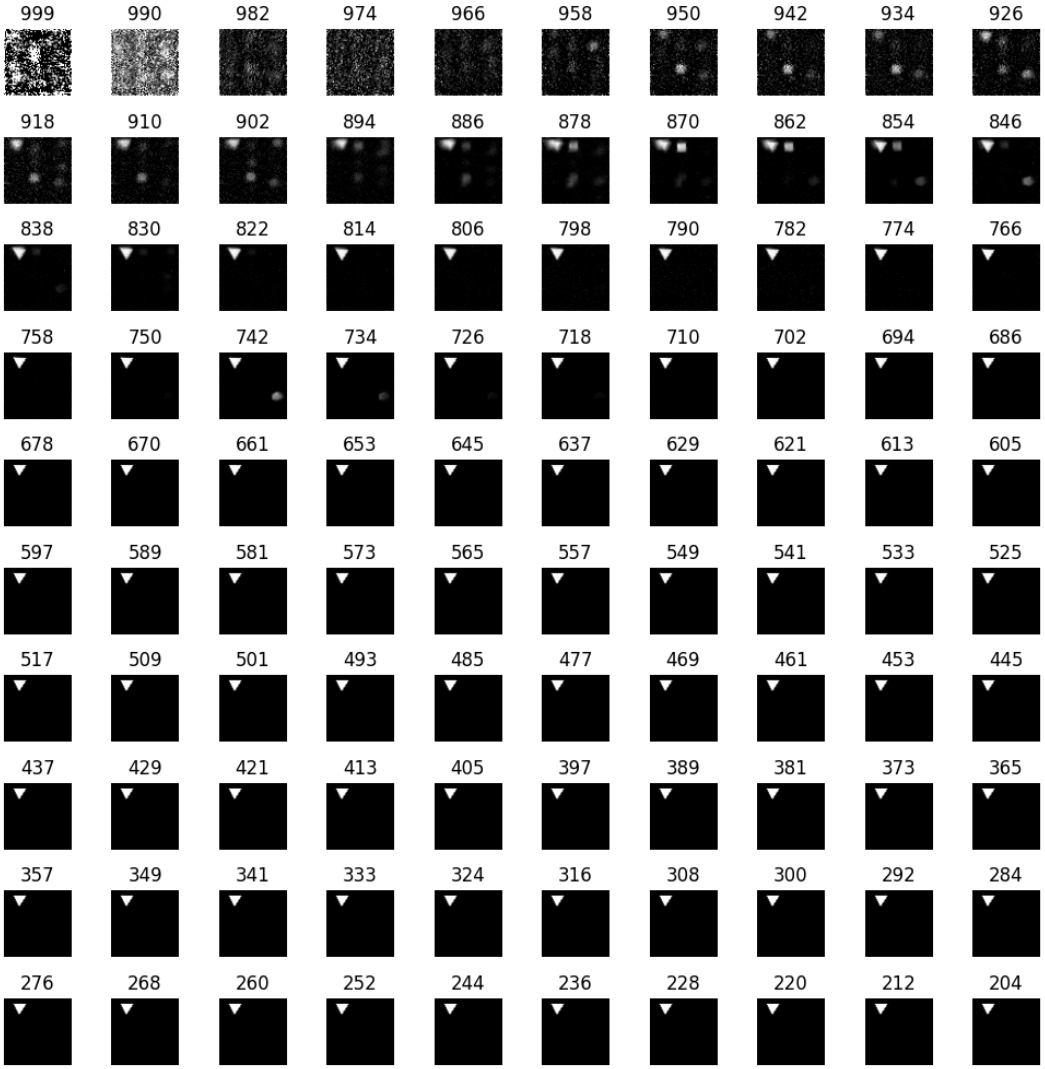

Figure 19: $\hat{x_0}$ for In-Support Sample. Here, we observe that the predicted x_0 is more consistent around $t = 700$ to $t = 850$.

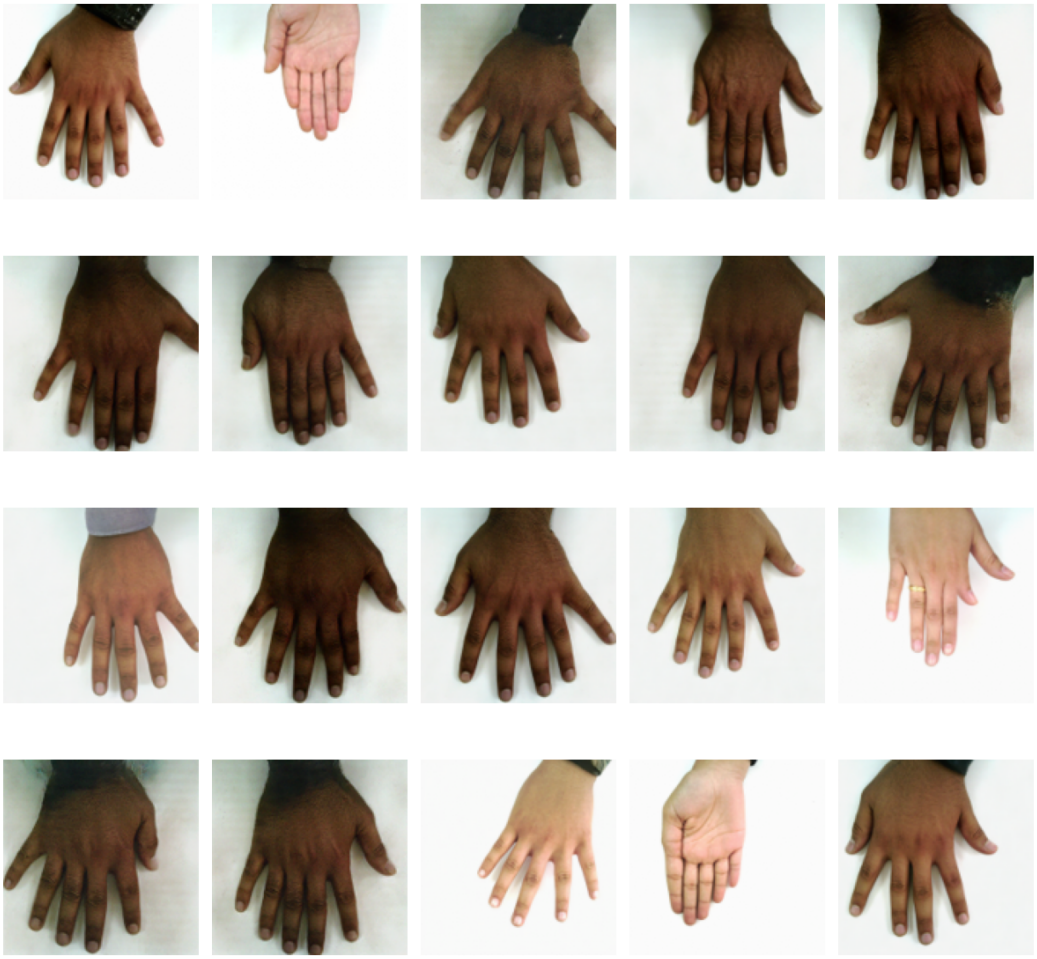

Figure 20: Hallucinated Images of Hands generated by the diffusion model.

https://doi.org/10.52202/079017-4278

Figure 21: In-Support Images of Hands generated by the diffusion model.

Figure 22: $\hat{x_0}$ trajectory for Hallucinated Sample (with 250 timesteps). We observe high variance/instability during the steps $t = 600$ to $t = 900$.

https://doi.org/10.52202/079017-4278

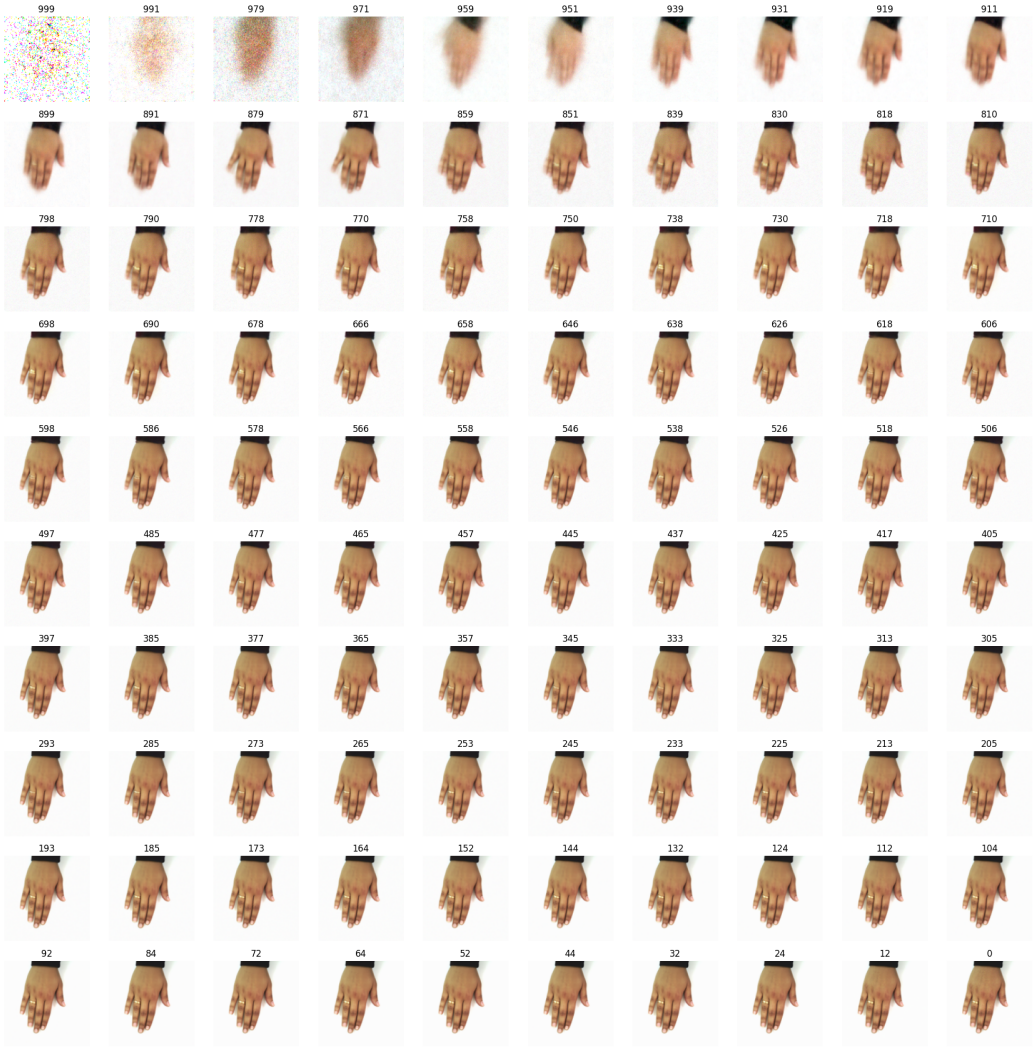

Figure 23: $\hat{x_0}$ trajectory for In-Support Sample (with 250 timesteps). We do not observe high variance/instability during the steps $t = 600$ to $t = 900$.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We provide experiments and analysis in Section 4, 5 and 6 as evidence to the claims made in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in Appendix B.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

https://doi.org/10.52202/079017-4278

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Appendix A contains the experimental details in suffient detail to reproduce the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We attach the code in supplementary material. The code to generate the synthetic dataset is also available in the supplementary material. We will release the code publicly upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We describe this in Appendix A and also provide code.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We train with 2 random seeds for the Gaussian experiments. The plots are available in the Appendix. For recursive training on MNIST, we do not report error bar as it is computationally expensive to run multiple seeds.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

 https://doi.org/10.52202/079017-4278

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We mention this in Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: All the research follows the NeurIPS code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: This is an investigative work analysing a particular failure mode of diffusion models. We discuss broader impacts in Appendix B.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We credit the authors of the codebase we base our implementation on in Appendix A.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

https://doi.org/10.52202/079017-4278

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: There are no new assets in this paper.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve any crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-4278