Understanding Representation Dynamics of Diffusion Models via Low-Dimensional Modeling

Xiao Li[1]*, Zekai Zhang[1] Xiang Li[1] Siyi Chen[1] Zhihui Zhu[2] Peng Wang[1] Qing Qu[1]
[1]University of Michigan, [2]Ohio State University
xlxiao@umich.edu, zzekai@umich.edu, forkobe@umich.edu, siyich@umich.edu,
zhu.3440@osu.edu, pengwa@umich.edu, qingqu@umich.edu

This work addresses the critical question of why and when diffusion models, despite being designed for generative tasks, can excel at learning high-quality representations in a self-supervised manner. To address this, we develop a mathematical framework based on a low-dimensional data model and posterior estimation, revealing a fundamental trade-off between generation and representation quality near the final stage of image generation. Our analysis explains the unimodal representation dynamics across noise scales, mainly driven by the interplay between data denoising and class specification. Building on these insights, we propose an ensemble method that aggregates features across noise levels, significantly improving both clean performance and robustness under label noise. Extensive experiments on both synthetic and real-world datasets validate our findings.

1. Introduction

Diffusion models, a new class of likelihood-based generative models, have achieved great empirical success in various tasks such as image and video generation, speech and audio synthesis, and solving inverse problems [1–14]. These models, consisting of forward and backward processes, learn data distributions by simulating the non-equilibrium thermodynamic diffusion process [2, 15, 16]. The forward process progressively adds Gaussian noise to training samples until the data is fully destroyed, while the backward process involves training the score to generate samples from the noisy inputs [16, 17].

In addition to their impressive generative capabilities, recent studies [18–23] have found the exceptional representation power of diffusion models. They showed that the encoder in the learned denoisers can act as a powerful self-supervised representation learner, enabling strong performance on downstream tasks such as classification [19, 20], semantic segmentation [18], and image alignment [22]. Notably, these learned features often match or even surpass existing methods specialized for self-supervised learning. These findings indicate the potential of diffusion models to serve as a unified foundation model for both generative and recognition vision tasks—paralleling the role of large language models like the GPT family in the NLP domain [24, 25].

However, it remains unclear whether the representation capabilities of diffusion models stem from the diffusion process or the denoising mechanism [26]. Moreover, despite recent endeavors [21, 23], our understanding of the representation power of diffusion models across noise levels remains limited. As shown in Baranchuk et al. [18], Xiang et al. [19], Tang et al. [22], while these models exhibit a coarse-to-fine progression in generation quality from noise to image, their representation quality follows an unimodal curve—indicating a trade-off between generation and representation quality near the final stage of image generation. Understanding this unimodal representation curve is key to uncovering the mechanisms underlying representation learning in diffusion models. These insights can inform the development of more principled approaches for downstream representation learning, including improved feature selection and ensemble or distillation methods. Furthermore, understanding and balancing the trade-off between representation capacity and generative perfor-

*The first two authors contributed equally to the work.

Preprint. Work in progress.

mance is essential for developing diffusion models as unified foundation models for both generative and recognition tasks.

Summary of contributions. We introduce a mathematical framework for studying representation learning in diffusion models using low-dimensional data models that capture the intrinsic structure of image datasets [27, 28]. Building on the denoising auto-encoder (DAE) formulation [29–31], we derive insights into unimodal representation quality across noise levels and the representation–generation trade-off by analyzing how diffusion models learn low-dimensional distributions. We further demonstrate how these theoretical findings enhance representation learning in practice. Our main contributions are:

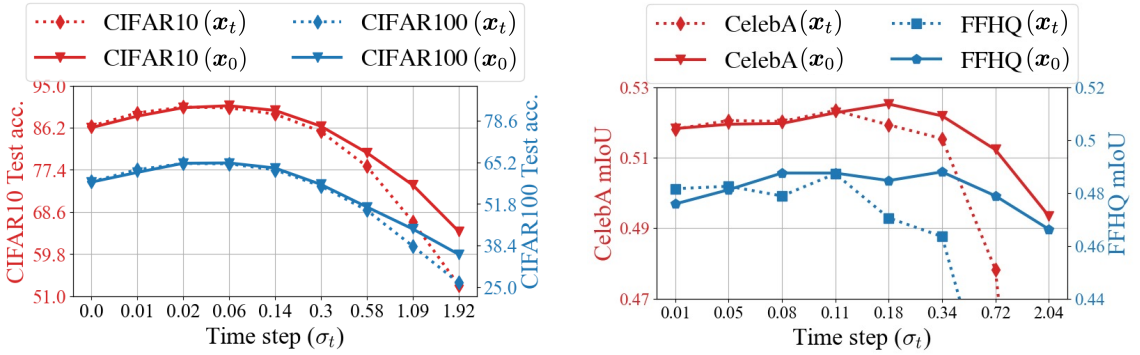
- **Mathematical framework for studying representation learning in diffusion models.** We introduce the Class-Specific Signal-to-Noise Ratio (CSNR) to quantify representation quality and analyze representation dynamics through the lens of how diffusion models learn a noisy mixture of low-rank Gaussians (MoLRG) distribution [32]. Our analysis reveals that the denoising objective is the primary driver of representation learning, while the diffusion process itself has a minimal impact. Based on these insights, we recommend using clean images as inputs for feature extraction in diffusion-based representation learning.
- **Theoretical characterization of unimodal curve and representation–generation trade-off.** Building on our mathematical framework, we theoretically characterize the unimodal behavior of representation quality across noise levels in diffusion models. By linking representation quality to the CSNR of optimal posterior estimation, we show that unimodality arises from the interplay between denoising strength and class confidence across varying noise scales. This analysis provides fundamental insights into the trade-off between representation quality and generative performance.
- **Empirical insights into diffusion-based representation learning.** Our findings offer practical guidance for optimizing diffusion models in representation learning. Specifically, we introduce a soft-voting ensemble method that aggregates features across noise levels, leading to significant improvements in classification performance and robustness to label noise. Additionally, we uncover an implicit weight-sharing mechanism in diffusion models, which explains their superior performance and more stable representations compared to traditional DAEs.

Relationship to prior results. As discussed in Appendix A.1, empirical developments of leveraging diffusion models for downstream representation learning have gained significant attention. However, a theoretical understanding of how diffusion models learn representations across different noise levels remains largely unexplored. A recent study by [23] takes initial steps in this direction by analyzing the optimization dynamics of a two-layer CNN trained with diffusion loss on binary class data. Since their framework does not explicitly distinguish between timesteps, their conclusions remain general across different noise levels. In contrast, our work characterizes and compares representations learned at different timesteps, provides a deeper understanding of diffusion-based representation learning and also extends to multi-class settings. A recent study by [33] also investigates the influence of timesteps in diffusion-based representation learning, but with a methodological focus. Unlike our work, it does not provide a theoretical explanation or practical metrics for the emergence of unimodal representation dynamics or the trade-off between representation quality and generative performance.

2. Problem Setup

2.1. Preliminaries on Denoising Diffusion Models

Basics of diffusion models. Diffusion models are a class of probabilistic generative models that aim to reverse a progressive noising process by mapping an underlying data distribution, p_{data}, to a Gaussian distribution.



(a) Classification (b) Segmentation

Figure 1: **Unimodal representation dynamic: using clean images as inputs improves representation quality.** We train DDPM/EDM-based diffusion models on various datasets and evaluate downstream performance using clean images ($\boldsymbol{x}_0$) versus noisy images ($\boldsymbol{x}_t$). Both scenarios show a unimodal performance trend, peaking at intermediate noise levels. While clean images perform similarly to noisy inputs at low noise, they outperform as noise increases.

- *The forward diffusion process.* Starting from clean data $\boldsymbol{x}_0$, noise is progressively added following a schedule based on time step t until the data becomes pure Gaussian noise. At each step t, the noised data is expressed as $\boldsymbol{x}_t = s_t\boldsymbol{x}_0 + s_t\sigma_t\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ is Gaussian noise, and s_t, $s_t\sigma_t$ scale the signal and noise, respectively.
- *The reverse diffusion process.* We can run a reverse SDE [34] to sample from $\boldsymbol{x}_1$ as:

$$\mathrm{d}\boldsymbol{x}_t = \left(f(t)\boldsymbol{x}_t - g^2(t)\nabla \log p_t(\boldsymbol{x}_t)\right)\mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}_t,$$

  where $\{\bar{\boldsymbol{w}}_t\}_{t\in[0,1]}$ is the standard Wiener process running backward in time from $t = 1$ to $t = 0$ and the functions $f(t), g(t) : \mathbb{R} \to \mathbb{R}$ respectively denote the drift and diffusion coefficients.

Training with denoising auto-encoder (DAE) based objective. Since the score function $\nabla \log p_t(\boldsymbol{x}_t)$ depends on the unknown data distribution p_{data} and satisfies

$$s_t\,\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}_t\right] = \boldsymbol{x}_t + s_t^2\sigma_t^2\nabla \log p_t(\boldsymbol{x}_t), \tag{1}$$

we can estimate $\nabla \log p_t(\boldsymbol{x}_t)$ by training a network $\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ to approximate the posterior mean $\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]$ [19, 21, 35]. This is achieved by minimizing the loss $\mathcal{L}(\boldsymbol{\theta})$ via

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \int_0^1 \lambda_t\,\mathbb{E}_{\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{x}_t^{(i)}, t) - \boldsymbol{x}_0^{(i)}\right\|^2\right]\mathrm{d}t, \tag{2}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_n)$, λ_t represents the weight associated with each noise level, and N denotes the dataset size. Moreover, $\boldsymbol{x}_i^{(0)} \overset{i.i.d.}{\sim} p_{\text{data}}$ is the training sample and the corresponding noisy sample is given by $\boldsymbol{x}_t^{(i)} = s_t\boldsymbol{x}_0^{(i)} + s_t\sigma_t\boldsymbol{\epsilon}$ for each $i \in [N]$. To simplify the analysis, we assume throughout the paper that $s_t = 1$ and λ_t remain constant across all noise levels, with the noise level denoted as σ_t. It is worth noting if we only minimize the error at one specific timestep, we are exactly training a single step DAE proposed in [31], and in Section 4.2 we discuss the superiority of diffusion models over DAEs.

2.2. Representation Learning via Diffusion Models

Once the diffusion model $\boldsymbol{x}_{\boldsymbol{\theta}}$ is trained via (2), we extract and evaluate the representation from data $\boldsymbol{x}_0$ as follows:

- **Using clean images as network inputs.** We propose to use clean images, $\boldsymbol{x}_0$, as inputs to the network $\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)$ for extracting representation across different noise levels t. This is different from existing approaches [18, 19, 22] that use noisy images $\boldsymbol{x}_t$ for representation extraction.

- **Layer selection for representations.** Following the protocol in [19], we freeze the entire model and extract representations from the layer of the diffusion model $\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)$ that yields the best downstream performance.[2] Typically, we select a layer near the bottleneck layer of U-Net and the exact midpoint layer of DiT to balance between data compression and performance.
- **Evaluation of representation.** Once the model is trained, we freeze the model and assess its representation quality based on downstream performance metrics, such as accuracy for classification and mean intersection over union (mIoU) for segmentation tasks.

Remarks on network inputs. Beyond simplifying the analysis, our choice of using *clean* images as network inputs $\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)$ (rather than $\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$) for representation extraction is driven by two primary considerations.

- *Empirical performance gains.* As demonstrated in Figure 1, using clean inputs outperforms using noisy inputs $\boldsymbol{x}_t$ on both classificaiton and segmentation tasks, a result further supported by our studies on the posterior estimation; see Figure 10 in the Appendix. These findings imply that the denoising objective is the primary driver of representation capabilities in diffusion models, whereas the progressive denoising procedure has a relatively minor impact on representation quality.
- *Alignment with existing learning paradigms.* Moreover, our approach is also consistent with standard supervised and self-supervised learning, where data augmentations (e.g., cropping [36], color jittering, masking [37]) are applied during training to improve robustness, but clean, unaugmented images are typically used at inference. In diffusion models, similarly, additive Gaussian noise serves as a form of data augmentation during training [21], while clean images are used for inference.

3. Study of Representation Dynamics

With the setup in Section 2, this section theoretically investigates the representation dynamics of diffusion models across the noise levels, providing new insights for understanding the representation-generation tradeoff. Moreover, our theoretical studies are corroborated by experimental results on real datasets.

3.1. Assumptions of Low-Dimensional Data Distribution

In this work, we assume that the input data follows a noisy version of the mixture of low-rank Gaussians (MoLRG) distribution [32, 38, 39], defined as follows.

Assumption 1 (K-Class Noisy MoLRG Distribution)**.** For any sample $\boldsymbol{x}_0$ drawn from the noisy MoLRG distribution with K subspaces, the following holds:

$$\boldsymbol{x}_0 = \boldsymbol{U}_k \boldsymbol{a} + \delta \boldsymbol{U}_k^{\perp} \boldsymbol{e}, \ \textit{with prob. } \pi_k \geq 0, \ k \in [K]. \quad (3)$$

Here, k represents the class of $\boldsymbol{x}_0$ and follows a multinomial distribution $k \sim \text{Mult}(K, \pi_k)$, $\boldsymbol{U}_k \in \mathcal{O}^{n \times d_k}$ denotes an orthonormal basis for the k-th subspace with its complement $\boldsymbol{U}_k^{\perp} \in \mathcal{O}^{n \times (n-d_k)}$, d_k is the subspace dimension with $d_k \ll n$, and the coefficient $\boldsymbol{a} \overset{i.i.d.}{\sim} \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{d_k})$ is drawn from the normal distribution. The level of the noise $\boldsymbol{e} \overset{i.i.d.}{\sim} \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{n-d_k})$ is controlled by the scalar $\delta < 1$.

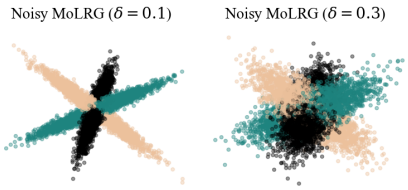


Figure 2: **An illustration of MoLRG with different noise levels.** We visualize samples drawn from noisy MoLRG with noise levels $\delta = 0.1,\ 0.3$ and $K = 3$.

[2] After feature extraction, we apply a global average pooling to the features. For instance, given a feature map of dimension $256 \times 4 \times 4$, we pool the last two dimensions, resulting in a 256-dimensional vector.

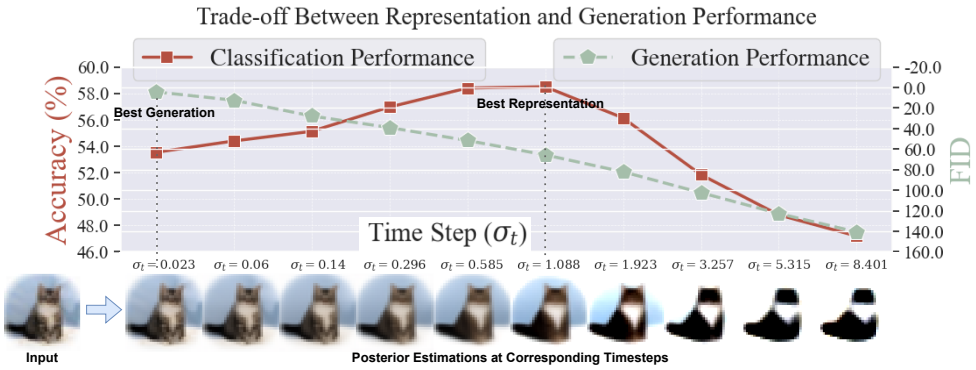


Figure 3: **Trade-offs between representation quality and generation quality.** The curve with pentagon markers demonstrates the transition from fine to coarse granularity in posterior estimation as noise levels increase, corresponding to the monotonic rise in FID. In contrast, the curve with square markers reveals an unimodal trend in posterior classification accuracy, achieving peak performance at intermediate noise levels. This occurs when high-level details are filtered out while essential low-level semantic information is preserved, as illustrated by the posterior estimations according to different noise levels shown at the bottom of the figure.

As shown in Figure 2, data from MoLRG resides on a union of low-dimensional subspaces, each following a Gaussian distribution with a low-rank covariance matrix representing its basis. The study of Noisy MoLRG distributions is further motivated by the fact that

- *MoLRG captures the intrinsic low-dimensionality of image data.* Although real-world image datasets are high-dimensional in terms of pixel count and data volume, extensive empirical studies [27, 28, 40] demonstrated that their intrinsic dimensionality is considerably lower. Additionally, recent work [41, 42] has leveraged the intrinsic low-dimensional structure of real-world data to analyze the convergence guarantees of diffusion model sampling. The MoLRG distribution, which models data in a low-dimensional space with rank $d_k \ll n$, effectively captures this property.
- *The latent space of latent diffusion models is approximately Gaussian.* State-of-the-art large-scale diffusion models [43, 44] typically employ autoencoders [45] to project images into a low-dimensional latent space, where a KL penalty encourages the learned latent distribution to approximate standard Gaussians [3]. Furthermore, recent studies [21, 46] show that diffusion models can be trained to leverage the intrinsic subspace structure of real-world data.
- *Modeling the complexity of real-world image datasets.* The noise term $\delta \boldsymbol{U}_k^{\perp} \boldsymbol{e}_i$ captures perturbations outside the k-th subspace via the orthogonal complement $\boldsymbol{U}_k^{\perp}$, analogous to insignificant attributes of real-world images, such as the background of an image. While this additional noise term may be less significant for representation learning, it plays a crucial role in enhancing the fidelity of generated samples.

Moreover, the noisy MoLRG is analytically tractable. For simplicity, we assume equal subspace dimensions ($d_1 = \cdots = d_K = d$), orthogonal bases ($\boldsymbol{U}_k^T \boldsymbol{U}_l = \boldsymbol{0}$ for $k \neq l$), uniform mixing weights ($\pi_1 = \cdots = \pi_K = 1/K$), and define the noise space as $\boldsymbol{U}_{\perp} = \bigcap_{k=1}^{K} \boldsymbol{U}_k^{\perp} \in \mathcal{O}^{n \times (n-Kd)}$. Then, we can derive the ground truth posterior mean $\mathbb{E}[\boldsymbol{x}_0 | \boldsymbol{x}_t]$ for the noisy MoLRG distribution as:

Proposition 1. *Suppose the data $\boldsymbol{x}_0$ is drawn from a noisy MoLRG data distribution with K-class and noise level δ. Let $\zeta_t = \frac{1}{1+\sigma_t^2}$ and $\xi_t = \frac{\delta^2}{\delta^2+\sigma_t^2}$, where σ_t is the noise scaling in (1). Then for each time $t > 0$, the optimal posterior estimator $\mathbb{E}[\boldsymbol{x}_0 | \boldsymbol{x}_t]$ has the analytical form:*

$$\mathbb{E}[\boldsymbol{x}_0 | \boldsymbol{x}_t] = \sum_{l=1}^{K} w_l^{\star}(\boldsymbol{x}_t, t) \left(\zeta_t \boldsymbol{U}_l \boldsymbol{U}_l^T + \xi_t \boldsymbol{U}_l^{\perp} \boldsymbol{U}_l^{\perp T} \right) \boldsymbol{x}_t.$$

where $w_l^{\star}(\boldsymbol{x}_t, t) = \frac{\exp(g_l(\boldsymbol{x}_t, t))}{\sum_{l=1}^{K} \exp(g_l(\boldsymbol{x}_t, t))}$ is a soft-max operator for $g_l(\boldsymbol{x}, t) = \frac{1}{2\sigma_t^2} \zeta_t \|\boldsymbol{U}_l^T \boldsymbol{x}\|^2 + \frac{\delta^2}{2\sigma_t^2} \xi_t \|\boldsymbol{U}_l^{\perp T} \boldsymbol{x}\|^2$.

The proof can be found in Appendix A.4.1 and it is an extension of the result in [32]. For $\boldsymbol{x}_0$ following noisy MoLRG, note that the optimal solution $\hat{\boldsymbol{x}}^\star_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ of the training loss (2) would exactly be $\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]$. As such, as illustrated in Figure 3, the analytical form of the posterior estimation facilitates the study of generation-representation tradeoff across timesteps:

- *The generation quality.* The generation quality of posterior estimation cam be measured by $||\hat{\boldsymbol{x}}^\star_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0||^2$. As shown in Proposition 1, this error is minimized at $t = 0$ with $\sigma_t = 0$, where the true class weight satisfies $w^\star_k(\boldsymbol{x}_t) = 1$, yielding $\hat{\boldsymbol{x}}^\star_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \boldsymbol{x}_0$. As t increases, higher noise levels σ_t decrease $w^\star_k(\boldsymbol{x}_t)$, causing a monotonic increase in FID, as seen in Figure 3.
- *The representation quality.* The representation quality follows a unimodal trend across timesteps [19, 22], which can be measured through the posterior estimator $\hat{\boldsymbol{x}}^\star_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ (see Section 3.2). As shown in Figure 3, this unimodal behavior creates a trade-off between generation and representation quality, particularly at smaller t when closer to the original image.

3.2. Measuring Posterior Representation Quality

For understanding diffusion-based representation learning, we introduce a metric termed Class-specific Signal-to-Noise Ratio (CSNR) to quantify the posterior representation quality as follows.

Definition 1 (Class-specific Signal-to-Noise Ratio)**.** Suppose the data $\boldsymbol{x}_0$ follows the noisy MoLRG introduced in Assumption 1. Without loss of generality, let k denote the true class of $\boldsymbol{x}_0$. For its associated posterior estimator $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}$,

$$\mathrm{CSNR}(\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}, t) := \mathbb{E}_k \left[\frac{\mathbb{E}_{\boldsymbol{x}_0}[\|\boldsymbol{U}_k \boldsymbol{U}_k^T \hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)\|^2 \mid k]}{\mathbb{E}_{\boldsymbol{x}_0}[\sum_{l \neq k} \|\boldsymbol{U}_l \boldsymbol{U}_l^T \hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)\|^2 \mid k]} \right]$$

Here, $\boldsymbol{U}_k$ represents the basis of the subspace corresponding to the true class to which $\boldsymbol{x}_0$ belongs and the $\boldsymbol{U}_l$s with $l \neq k$ denotes the bases of the subspaces for other classes.

Intuitively, successful prediction of the class for $\boldsymbol{x}_0$ is achieved when the projection onto the correct class subspace, $\|\boldsymbol{U}_k \boldsymbol{U}_k^T \hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)\|$, preserves larger energy than the projections onto subspaces of any other class, $\|\boldsymbol{U}_l \boldsymbol{U}_l^T \hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)\|$. Thus, CSNR measures the ratio of the true class signal to irrelevant noise from other classes at a given noise level t, serving as a practical metric for evaluating classification performance and hence the representation quality. In this work, we use posterior representation quality as a proxy for studying the representation dynamics of diffusion models for the following reasons:

- *Posterior quality reflects feature quality.* Diffusion models $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}$ are trained to perform posterior estimation at a given time step t using corrupted inputs, with the intermediate features emerging as a byproduct of this process. Thus, a more class-representative posterior estimation inherently implies more class-representative intermediate features.
- *Model-agnostic analysis.* Our goal is to provide a general analysis independent of specific network architectures and feature extraction protocols. Posterior representation quality offers a unified metric that avoids assumptions tied to particular architectures, making the analysis broadly applicable.

3.3. Main Theoretical Results

Based upon the setup in Section 3.1 and Section 3.2, we obtain the following results.

Theorem 1. (*Informal*) *Suppose the data $\boldsymbol{x}_0$ follows the noisy MoLRG introduced in Assumption 1 with K classes and noise level δ, then the* CSNR *of the optimal denoiser $\hat{\boldsymbol{x}}^\star_\theta$ takes the following form:*

$$\mathrm{CSNR}(\hat{\boldsymbol{x}}^\star_\theta, t) = \frac{1}{(K-1)\delta^2} \cdot \left(\frac{1 + \frac{\sigma_t^2}{\delta^2} h(\hat{w}_t^+, \delta)}{1 + \frac{\sigma_t^2}{\delta^2} h(\hat{w}_t^-, \delta)} \right)^2. \tag{4}$$

Here, $h(w, \delta) := (1 - \delta^2)w + \delta^2$ is a monotonically increasing function with respect to w. Additionally, $h(\hat{w}_t^+, \delta)$ and $h(\hat{w}_t^-, \delta)$ denote positive and negative class confidence rates with

$$\begin{cases} \hat{w}_t^+(\sigma_t, \delta) &= \mathbb{E}_k[\mathbb{E}_{\boldsymbol{x}_0}[w_k(\boldsymbol{x}_0, t) \mid k]], \\ \hat{w}_t^-(\sigma_t, \delta) &= \mathbb{E}_k[\mathbb{E}_{\boldsymbol{x}_0}[w_l(\boldsymbol{x}_0, t) \mid k \neq l]], \end{cases}$$

whose analytical forms can be found in Appendix A.4.2.

We defer the formal statement of Theorem 1 and its proof to Appendix A.4.2. In the following, we discuss the implications of our result.

The unimodal curve of CSNR across noise levels. Intuitively, our theorem shows that unimodal curve is mainly induced by the the interplay between the "denoising rate" σ_t^2/δ^2 and the positive class confidence rate $h(\hat{w}_t^+, \delta)$ as noise level σ_t increases. As observed in Figure 4, the "denoising rate" (σ_t^2/δ^2) increases monotonically with σ_t while the class confidence rate $h(\hat{w}_t^+, \delta)$ monotonically declines. Initially, when σ_t is small, the class confidence rate remains relatively stable due to its flat slope, and an increasing "denoising rate" improves the CSNR, resulting in improved posterior estimation. However, as indicated by Proposition 1, when σ_t becomes too large, $h(\hat{w}_t^+, \delta)$ approaches $h(\hat{w}_t^-, \delta)$, leading to a drop in CSNR, which limits the ability of the model to project $\boldsymbol{x}_0$ onto the correct signal subspace and ultimately hurts posterior estimation.

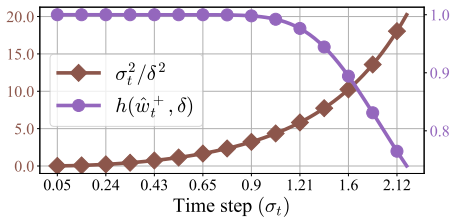


Figure 4: Illustration of the interplay between the denoising rate and the class confidence rate.

Alignment of CSNR with posterior representation quality. Although our theory is derived from the noisy `MoLRG` distribution, it effectively captures real-world phenomena. As shown in Figures 5 and 6, we conduct experiments on both synthetic (i.e., noisy `MoLRG`) and real-world datasets (i.e., CIFAR and ImageNet) to measure $\mathrm{CSNR}(\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}, t)$ as well as the posterior probing accuracy. For posterior probing, we use posterior estimations at different timesteps as inputs for classification. The results consistently show that $\mathrm{CSNR}(\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}, t)$ follows a unimodal pattern across all cases, mirroring the trend observed in posterior probing accuracy as the noise scale increases. This alignment provides a formal justification for previous empirical findings [18, 19, 22], which have reported a unimodal trajectory in the representation dynamics of diffusion models with increasing noise levels. Detailed experimental setups are provided in Appendix A.3.

Explanation of generation and representation trade-off. Our theoretical findings reveal the underlying rationale behind the generation and representation trade-off: the proportion of data associated with δ represents class-irrelevant attributes. The unimodal representation learning dynamic thus captures a "fine-to-coarse" shift [47, 48], where these class-irrelevant attributes are progressively stripped away. During this process, peak representation performance is achieved at a balance point where class-irrelevant attributes are eliminated, while class-essential information is preserved. In contrast, high-fidelity image generation requires capturing the entire data distribution—from coarse structures to fine details—leading to optimal performance at the lowest noise level σ_t, where class-irrelevant attributes encoded in the δ-term are maximally retained. Thus, our insights explain the trade-off between generation and representation quality. As visualized in Figure 3 and Figure 15, representation quality peaks at an intermediate noise level where irrelevant details are stripped away, while generation quality peaks at the lowest noise level, where all details are preserved.

4. Practical Insights

We examine the practical implications of our findings in Section 4.1, leveraging feature information at different levels of granularity to enhance robustness. Additionally, we discuss the advantages of diffusion models over traditional single-step DAEs in Section 4.2.

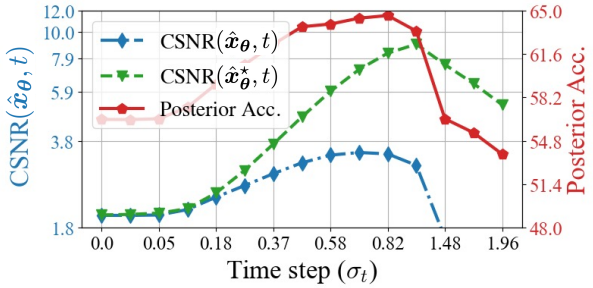
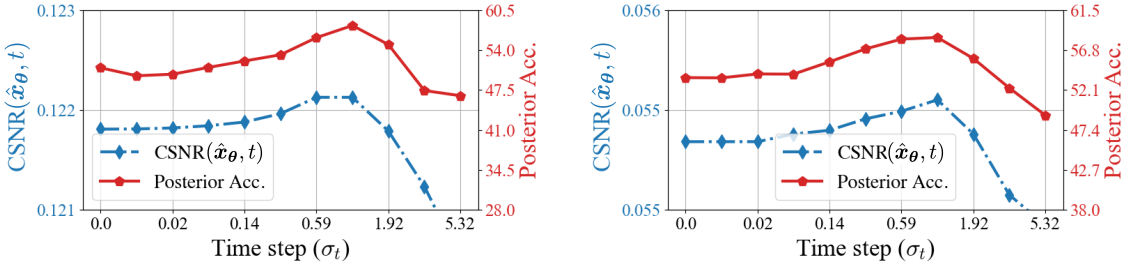


Figure 5: **Posterior probing accuracy and associated** CSNR **dynamics in** `MoLRG` **data.** We plot the posterior probing accuracy and CSNR with the posterior estimations obtained from a learned estimator $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}$. both of which exhibit a consistent unimodal pattern. Additionally, we include the optimal CSNR , calculated from the ground truth posterior function $\hat{\boldsymbol{x}}^\star_{\boldsymbol{\theta}}$ defined in Proposition 1, as a reference. The estimator is trained on a 3-class `MoLRG` dataset with data dimension $n = 50$, subspace dimension $d = 15$, and noise scale $\delta = 0.5$.



(a) CIFAR10 (b) MiniImageNet

Figure 6: **Dynamics of posterior probing accuracy and associated** CSNR **on CIFAR10 and MiniImageNet.** Posterior probing accuracy is plotted alongside CSNR($\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}, t$). Probing accuracy is evaluated on the test set, while the empirical CSNR is computed from the training set. Both exhibit an aligning unimodal pattern. We use released EDM models [49] trained on the CIFAR-10 [50] and ImageNet [51] datasets, evaluating them on CIFAR-10 and MiniImageNet [52], respectively. To compute CSNR , we apply PCA on the original CIFAR-10/MiniImageNet images to extract the basis $\boldsymbol{U}_k$s. Further details can be found in Appendix A.3.

4.1. Feature Ensembling Across Timesteps Improves Representation Robustness

Our theoretical insights imply that features extracted at different timesteps capture varying levels of granularity. Given the high linear separability of intermediate features, we propose a simple ensembling approach across multiple timesteps to construct a more holistic representation of the input. Specifically, in addition to the optimal timestep, we extract feature representations at four additional timesteps—two from the coarse (larger σ_t) and two from the fine-grained (smaller σ_t) end of the spectrum. We then train linear probing classifiers for each set and, during inference, apply a soft-voting ensemble by averaging the predicted logits before making a final decision.(experiment details in Appendix A.3)

We evaluate this ensemble method against results obtained from the best individual timestep, as well as a self-supervised method MAE [37], on both the pre-training dataset and a transfer learning setup. The results, reported in Table 1 and Table 2, demonstrate that ensembling significantly enhances performance for both EDM [49] and DiT [43], consistently outperforming their vanilla diffusion model counterparts and often surpassing MAE. Moreover, ensembling substantially improves the robustness of diffusion models for classification under label noise.

Method	*MiniImageNet** Test Acc. %				
Label Noise	Clean	20%	40%	60%	80%
MAE	73.7	70.3	67.4	62.8	51.5
EDM	67.2	62.9	59.2	53.2	40.1
EDM (Ensemble)	72.0	67.8	64.7	60.0	48.2
DiT	77.6	72.4	68.4	62.0	47.3
DiT (Ensemble)	**78.4**	**75.1**	**71.9**	**66.7**	**56.3**

Table 1: **Comparison of test performance across different methods under varying label noise levels.** All compared models are publicly available and pre-trained on ImageNet-1K [51], evaluated using MiniImageNet classes. Bold font highlights the best result in each scenario.

Method	Transfer Test Acc. %														
	CIFAR100					**DTD**					**Flowers102**				
Label Noise	Clean	20%	40%	60%	80%	Clean	20%	40%	60%	80%	Clean	20%	40%	60%	80%
MAE	63.0	58.8	54.7	50.1	38.4	61.4	54.3	49.9	40.5	24.1	68.9	55.2	40.3	27.6	9.6
EDM	62.7	58.5	53.8	48.0	35.6	54.0	49.1	45.1	36.4	21.2	62.8	48.2	37.2	24.1	9.7
EDM (Ensemble)	**67.5**	**64.2**	**60.4**	**55.4**	**43.9**	55.7	49.5	45.2	37.1	22.0	67.8	53.9	41.5	25.0	10.4
DiT	64.2	58.7	53.5	46.4	32.6	65.2	59.7	53.0	43.8	27.0	78.9	65.2	52.4	34.7	13.3
DiT (Ensemble)	66.4	61.8	57.6	51.3	39.2	**65.3**	**60.6**	**56.1**	**46.3**	**30.6**	**79.7**	**67.0**	**54.6**	**36.6**	**14.7**

Table 2: **Comparison of transfer learning performance across different methods under varying label noise levels.** All compared models are publicly available and pre-trained on ImageNet-1K [51], evaluated on different downstream datasets. Bold font highlights the best result in each scenario.

4.2. Weight Sharing in Diffusion Models Facilitates Representation Learning

Second, we reveal why diffusion models, despite sharing the same denoising objective with classical DAEs, achieve superior representation learning due to their inherent weight-sharing mechanism. By minimizing loss across all noise levels (2), diffusion models enable parameter sharing and interaction among denoising subcomponents, creating an implicit "ensemble" effect. This improves feature consistency and robustness across noise scales compared to DAEs [21], as illustrated in Figure 7.

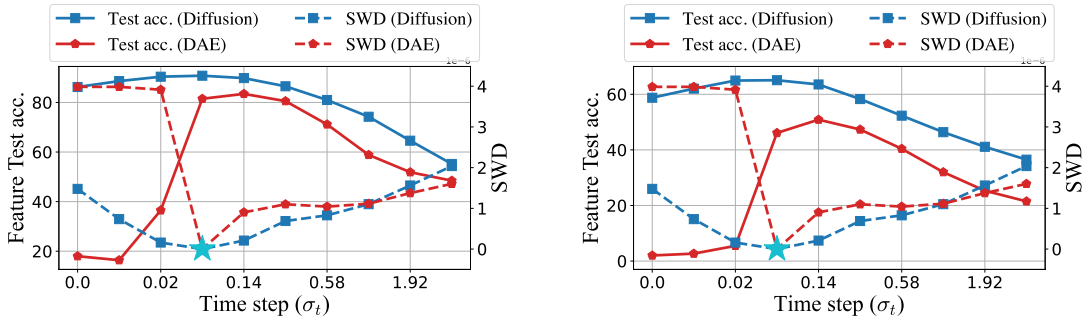
To test this, we trained 10 DAEs, each specialized for a single noise level, alongside a DDPM-based diffusion model on CIFAR10 and CIFAR100. We compared feature quality using linear probing accuracy and feature similarity relative to the optimal features at $\sigma_t = 0.06$ (where accuracy peaks) via sliced Wasserstein distance (SWD) [53].

The results in Figure 7 confirm the advantage of diffusion models over DAEs. Diffusion models consistently outperform DAEs, particularly in low-noise regimes where DAEs collapse into trivial identity mappings. In contrast, diffusion models leverage weight-sharing to preserve high-quality features, ensuring smoother transitions and higher accuracy as noise increases. This advantage is further supported by the SWD curve, which reveals an inverse correlation between feature accuracy and feature differences. Notably, diffusion model features remain significantly closer to their optimal state across all noise levels, demonstrating superior representational capacity.

Our finding also aligns with prior results that sequentially training DAEs across multiple noise levels improves representation quality [54–56]. Our ablation study further confirms that multi-scale training is essential for improving DAE performance on classification tasks in low-noise settings (details in Appendix A.2, Table 3).

5. Discussion

In this work, we develop a mathematical framework for analyzing the representation dynamics of diffusion models. By introducing the concept of CSNR and leveraging a low-dimensional mixture of low-rank Gaussians, we characterize the trade-off between generative quality and representation



(a) CIFAR10 (b) CIFAR100

Figure 7: **Diffusion models exhibit higher and smoother feature accuracy and similarity compared to individual DAEs.** We train DDPM-based diffusion models and individual DAEs on the CIFAR datasets and evaluate their representation learning performance. Feature accuracy, and feature differences from the optimal features (indicated by $\star$) are plotted against increasing noise levels. The results reveal an inverse correlation between feature accuracy and feature differences, with diffusion models achieving both higher/smoother accuracy and smaller/smoother feature differences compared to DAEs.

quality. Our theoretical analysis explains how the unimodal representation learning dynamics observed across noise scales emerge from the interplay between data denoising and class specification.

Beyond theoretical insights, we propose an ensemble method inspired by our findings that enhances classification performance in diffusion models, both with and without label noise. Additionally, we empirically uncover an inherent weight-sharing mechanism in diffusion models, which accounts for their superior representation quality compared to traditional DAEs. Experiments on both synthetic and real-world datasets validate our findings. Additionally, our findings also open up new avenues for future research that we discuss in the following.

- **Principled diffusion-based representation learning.** While diffusion models have shown strong performance in various representation learning tasks, their application often relies on trial-and-error methods and heuristics. For example, determining the optimal layer and noise scale for feature extraction frequently involves grid searches. Our work provides a theoretical framework to understand representation dynamics across noise scales. A promising future direction is to extend this analysis to include layer-wise dynamics. Combining these insights could pave the way for more principled and efficient approaches to diffusion-based representation learning.

- **Representation alignment for better image generation.** Recent work REPA Yu et al. [57] has demonstrated that aligning diffusion model features with features from pre-trained self-supervised foundation models can enhance training efficiency and improve generation quality. By providing a deeper understanding of the representation dynamics in diffusion models, our findings could further advance such representation alignment techniques, facilitating the development of diffusion models with superior training and generation performance.

References

[1] Ismail Alkhouri, Shijun Liang, Rongrong Wang, Qing Qu, and Saiprasad Ravishankar. Diffusion-based adversarial purification for robust deep mri reconstruction. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12841–12845. IEEE, 2024.

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[3] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[4] Huijie Zhang, Yifu Lu, Ismail Alkhouri, Saiprasad Ravishankar, Dogyoon Song, and Qing Qu. Improving training efficiency of diffusion models via multi-stage framework and tailored multi-decoder architectures. In *Conference on Computer Vision and Pattern Recognition 2024*, 2024. URL `https://openreview.net/forum?id=YtptmpZQOg`.

[5] Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, et al. Lumiere: A space-time diffusion model for video generation. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024.

[6] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

[7] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in neural information processing systems*, 33:17022–17033, 2020.

[8] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. DIFFWAVE: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021.

[9] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Transactions on Graphics (TOG)*, 42(1):1–13, 2022.

[10] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22500–22510, 2023.

[11] Siyi Chen, Huijie Zhang, Minzhe Guo, Yifu Lu, Peng Wang, and Qing Qu. Exploring low-dimensional subspace in diffusion models for controllable image editing. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=50aOEfb2km`.

[12] Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. *Advances in Neural Information Processing Systems*, 35:25683–25696, 2022.

[13] Bowen Song, Soo Min Kwon, Zecheng Zhang, Xinyu Hu, Qing Qu, and Liyue Shen. Solving inverse problems with latent diffusion models via hard data consistency. In *The Twelfth International Conference on Learning Representations*, 2024.

[14] Xiang Li, Soo Min Kwon, Ismail R Alkhouri, Saiprasad Ravishanka, and Qing Qu. Decoupled data consistency with diffusion purification for image restoration. *arXiv preprint arXiv:2403.06054*, 2024.

[15] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[16] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *International Conference on Learning Representations*, 2021.

[17] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[18] Dmitry Baranchuk, Andrey Voynov, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=SlxSY2UZQT`.

[19] Weilai Xiang, Hongyu Yang, Di Huang, and Yunhong Wang. Denoising diffusion autoencoders are unified self-supervised learners. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15802–15812, 2023.

[20] Soumik Mukhopadhyay, Matthew Gwilliam, Vatsal Agarwal, Namitha Padmanabhan, Archana Swaminathan, Srinidhi Hegde, Tianyi Zhou, and Abhinav Shrivastava. Diffusion models beat gans on image classification. *arXiv preprint arXiv:2307.08702*, 2023.

[21] Xinlei Chen, Zhuang Liu, Saining Xie, and Kaiming He. Deconstructing denoising diffusion models for self-supervised learning. *arXiv preprint arXiv:2401.14404*, 2024.

[22] Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. *Advances in Neural Information Processing Systems*, 36:1363–1389, 2023.

[23] Andi Han, Wei Huang, Yuan Cao, and Difan Zou. On the feature learning in diffusion models. *arXiv preprint arXiv:2412.01021*, 2024.

[24] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[25] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[26] Michael Fuest, Pingchuan Ma, Ming Gui, Johannes S Fischer, Vincent Tao Hu, and Bjorn Ommer. Diffusion models and representation learning: A survey. *arXiv preprint arXiv:2407.00783*, 2024.

[27] Phil Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=XJk19XzGq2J`.

[28] Jan Stanczuk, Georgios Batzolis, Teo Deveney, and Carola-Bibiane Schönlieb. Your diffusion model secretly knows the dimension of the data manifold. *arXiv preprint arXiv:2212.12611*, 2022.

[29] Pascal Vincent, H. Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning*, 2008.

[30] Pascal Vincent, H. Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, 2010.

[31] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[32] Peng Wang, Huijie Zhang, Zekai Zhang, Siyi Chen, Yi Ma, and Qing Qu. Diffusion models learn low-dimensional distributions via subspace clustering. *arXiv preprint arXiv:2409.02426*, 2024.

[33] Zhongqi Yue, Jiankun Wang, Qianru Sun, Lei Ji, Eric I-Chao Chang, and Hanwang Zhang. Exploring diffusion time-steps for unsupervised representation learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=bWzxhtl1HP`.

[34] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[35] Zahra Kadkhodaie, Florentin Guth, Eero P Simoncelli, and Stéphane Mallat. Generalization in diffusion models arises from geometry-adaptive harmonic representations. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=ANvmVS2Yr0`.

[36] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[37] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[38] Ehsan Elhamifar and René Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2765–2781, 2013.

[39] Peng Wang, Huikang Liu, Anthony Man-Cho So, and Laura Balzano. Convergence and recovery guarantees of the k-subspaces method for subspace clustering. In *International Conference on Machine Learning*, pages 22884–22918. PMLR, 2022.

[40] Sixue Gong, Vishnu Naresh Boddeti, and Anil K Jain. On the intrinsic dimensionality of image representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3987–3996, 2019.

[41] Zhihan Huang, Yuting Wei, and Yuxin Chen. Denoising diffusion probabilistic models are optimally adaptive to unknown low dimensionality. *arXiv preprint arXiv:2410.18784*, 2024.

[42] Jiadong Liang, Zhihan Huang, and Yuxin Chen. Low-dimensional adaptation of diffusion models: Convergence in total variation. *arXiv preprint arXiv:2501.12982*, 2025.

[43] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[44] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024.

[45] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[46] Bowen Jing, Gabriele Corso, Renato Berlinghieri, and Tommi Jaakkola. Subspace diffusion generative models. In *European Conference on Computer Vision*, pages 274–289. Springer, 2022.

[47] Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11472–11481, 2022.

[48] Binxu Wang and John J Vastola. Diffusion models generate images like painters: an analytical theory of outline first, details later. *arXiv preprint arXiv:2303.02490*, 2023.

[49] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. NeurIPS*, 2022.

[50] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[51] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[52] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[53] Anh-Dzung Doan, Bach Long Nguyen, Surabhi Gupta, Ian Reid, Markus Wagner, and Tat-Jun Chin. Assessing domain gap for continual domain adaptation in object detection. *Computer Vision and Image Understanding*, 238:103885, 2024.

[54] B. Chandra and Rajesh Kumar Sharma. Adaptive noise schedule for denoising autoencoder. In *International Conference on Neural Information Processing*, 2014.

[55] Krzysztof Geras and Charles Sutton. Scheduled denoising autoencoders. In *International Conference on Learning Representations (ICLR) 2015*, 2015.

[56] Qianjun Zhang and Lei Zhang. Convolutional adaptive denoising autoencoders for hierarchical feature extraction. *Frontiers of Computer Science*, 12:1140 – 1148, 2018.

[57] Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. *arXiv preprint arXiv:2410.06940*, 2024.

[58] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[59] Arnu Pretorius, Steve Kroon, and Herman Kamper. Learning dynamics of linear denoising autoencoders. In *International Conference on Machine Learning*, pages 4141–4150. PMLR, 2018.

[60] Harald Steck. Autoencoders that don't overfit towards the identity. In *Neural Information Processing Systems*, 2020.

[61] Daniel Kunin, Jonathan Bloom, Aleksandrina Goeva, and Cotton Seed. Loss landscapes of regularized linear autoencoders. In *International conference on machine learning*, pages 3560–3569. PMLR, 2019.

[62] Kamil Deja, Tomasz Trzciński, and Jakub M Tomczak. Learning data representations with joint diffusion models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 543–559. Springer, 2023.

[63] Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Hanshu Yan, Wenqing Zhang, Vincent YF Tan, and Song Bai. Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8839–8849, 2024.

[64] Xingyi Yang and Xinchao Wang. Diffusion model as representation learner. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18938–18949, 2023.

[65] Daiqing Li, Huan Ling, Amlan Kar, David Acuna, Seung Wook Kim, Karsten Kreis, Antonio Torralba, and Sanja Fidler. Dreamteacher: Pretraining image backbones with deep generative models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16698–16708, 2023.

[66] Nick Stracke, Stefan Andreas Baumann, Kolja Bauer, Frank Fundel, and Björn Ommer. Cleandift: Diffusion features without noise. *arXiv preprint arXiv:2412.03439*, 2024.

[67] Grace Luo, Lisa Dunlap, Dong Huk Park, Aleksander Holynski, and Trevor Darrell. Diffusion hyperfeatures: Searching through time and space for semantic correspondence. *Advances in Neural Information Processing Systems*, 36, 2024.

[68] Korbinian Abstreiter, Sarthak Mittal, Stefan Bauer, Bernhard Schölkopf, and Arash Mehrjou. Diffusion-based representation learning. *arXiv preprint arXiv:2105.14257*, 2021.

[69] Yingheng Wang, Yair Schiff, Aaron Gokaslan, Weishen Pan, Fei Wang, Christopher De Sa, and Volodymyr Kuleshov. Infodiffusion: Representation learning using information maximizing diffusion models. In *International Conference on Machine Learning*, pages 36336–36354. PMLR, 2023.

[70] Drew A Hudson, Daniel Zoran, Mateusz Malinowski, Andrew K Lampinen, Andrew Jaegle, James L McClelland, Loic Matthey, Felix Hill, and Alexander Lerchner. Soda: Bottleneck diffusion models for representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23115–23127, 2024.

[71] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10619–10629, 2022.

[72] Huijie Zhang, Jinfan Zhou, Yifu Lu, Minzhe Guo, Peng Wang, Liyue Shen, and Qing Qu. The emergence of reproducibility and consistency in diffusion models. In *Forty-first International Conference on Machine Learning*, 2023.

[73] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729, 2008.

[74] Mingi Kwon, Jaeseok Jeong, and Youngjung Uh. Diffusion models already have a semantic latent space. *arXiv preprint arXiv:2210.10960*, 2022.

[75] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Hk99zCeAb`.

[76] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 43(12):4217–4228, 2021. ISSN 1939-3539. doi: 10.1109/TPAMI.2020.2970919. URL `https://doi.ieeecomputersociety.org/10.1109/TPAMI.2020.2970919`.

[77] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium–supplementary material. *Advances in Neural Information Processing Systems*, 2017.

[78] tanelp. tiny-diffusion. `https://github.com/tanelp/tiny-diffusion`, 2022.

[79] Diederik P Kingma. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[80] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

A. Appendix

The Appendix is organized as follows: in Appendix A.1, we discuss related works; in Appendix A.2, we provide complementary experiments; in Appendix A.3, we present the detailed experimental setups for the empirical results in the paper. Lastly, in Appendix A.4, we provide proof details for Section 3.

A.1. Related Works

Denoising auto-encoders. Denoising autoencoders (DAEs) are trained to reconstruct corrupted images to extract semantically meaningful information, which can be applied to various vision [29, 30] and language downstream tasks [58]. Related to our analysis of the weight-sharing mechanism, several studies have shown that training with a noise scheduler can enhance downstream performance [54–56]. On the theoretical side, prior works have studied the learning dynamics [59, 60] and optimization landscape [61] through the simplified linear DAE models.

Diffusion-based representation learning. Diffusion-based representation learning [26] has demonstrated significant success in various downstream tasks, including image classification [19, 20, 62], segmentation [18], correspondence [22], and image editing [63]. To further enhance the utility of diffusion features, knowledge distillation [64–67] methods have been proposed, aiming to bypass the computationally expensive grid search for the optimal t in feature extraction and improving downstream performance. Beyond directly using intermediate features from pre-trained diffusion models, research efforts has also explored novel loss functions [68, 69] and network modifications [70, 71] to develop more unified generative and representation learning capabilities within diffusion models. Unlike the aforementioned efforts, our work focuses more on understanding the representation learning capabilities of diffusion models.

A.2. Additional Experiments

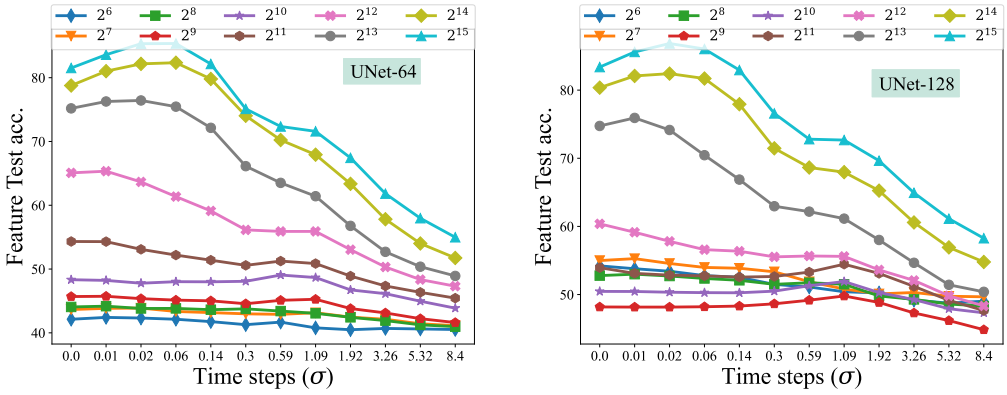
Influence of data complexity in diffusion representation learning Our analyses in the main body of the paper are based on the assumption that the training dataset contains sufficient samples for the diffusion model to learn the underlying distribution. Interestingly, if this assumption is violated by training the model on insufficient data, the unimodal representation learning dynamic disappears and the probing accuracy also drops severely.

As illustrated in Figure 8, we train 2 different UNets following the EDM [49] configuration with training dataset size ranging from 2^5 to 2^{15}. The unimodal curve emerges only when the dataset size exceeds 2^{12}, whereas smaller datasets produce flat curves.

The underlying reason for this observation is that, when training data is limited, diffusion models memorize all individual data points rather than learn the true underlying data structure [32]. In this scenario, the model memorizes an empirical distribution that lacks meaningful low-dimensional structures and thus deviates from the setting in our theory, leading to the loss of the unimodal representation dynamic. To confirm this, we calculated the generalization score, which measures the percentage of generated data that does not belong to the training dataset, as defined in [72]. As shown in Figure 9, representation learning only achieves strong accuracy and displays the unimodal dynamic when the generalization score approaches 1, aligning with our theoretical assumptions.

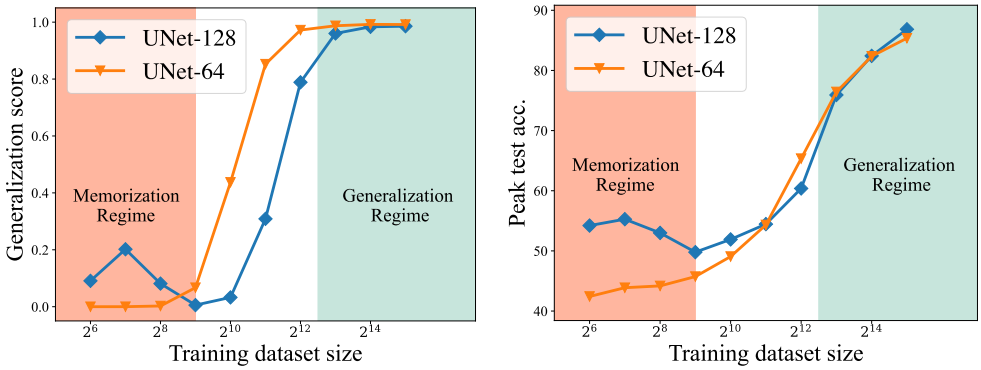
Posterior quality decides feature quality Diffusion models $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}$ are trained to perform posterior estimation at a given time step t using corrupted inputs, with the features for representation learning emerging as an intermediate byproduct of this process. This leads to a natural conjecture: *changes in feature quality should directly correspond to changes in posterior estimation quality.*

To test this hypothesis, we visualize the posterior estimation results for clean inputs ($\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)$) and noisy inputs ($\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$) across varying noise scales σ_t in Figure 10. The results reveal that, similar to the findings for feature representation, clean inputs yield superior posterior estimation compared to corrupted inputs, with the performance gap widening as the noise scale increases. Furthermore, as



(a) UNet-64 accuracy trend (b) UNet-128 accuracy trend

Figure 8: **The influence of data complexity in diffusion-based representation learning.** With the same model trained in Figure 9, we plot the representation learning dynamics for each trained model as a function of changing noise levels.



(a) Phase transition in generalization score (b) Phase transition in representation learning

Figure 9: **Better representations are learned in the generalization regime.** We train EDM-based [49] diffusion models on the CIFAR-10 dataset using different training dataset sizes, ranging from 2^6 to 2^{15}. (a) The change in the generalization score [72] as the dataset size increases, where regions with a generalization score close to 0 are labeled as the memorization regime, and those close to 1 are labeled as the generalization regime. (b) The peak representation learning feature accuracy achieved as a function of dataset size.

illustrated in the Figure 3, if we consider posterior estimation as the last-layer features and directly use it for classification, the accuracy curve reveals a unimodal trend as noise level progresses, similar to the behavior observed in feature classification accuracy. These findings strongly validate the conjecture.

Building on this insight, we use posterior estimation as a proxy to analyze the dynamics of diffusion-based representation learning in Section 3. Moreover, since the unimodal representation dynamic persists across different network architectures and feature extraction layers, analyzing posterior estimation also enables us to study the problem without relying on specific architectural or layer-dependent assumptions.

Additional representation learning experiments on DDPM. Apart from EDM and DDPM* models pre-trained using the framework proposed by [49], we also experiment with the features extracted by classic DDPM models [2] to make sure the observations do not depend on the specific training framework. We use the same groups of noise levels and also test using clean or noisy images as input to extract features at the bottleneck layer, and then conduct the linear probe. The DDPM models we use are trained on the Flowers-102 [73] and the CIFAR10 dataset accordingly. Different

(a) $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$: Posterior estimation using **noise image** as inputs.

(b) $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)$: Posterior estimation using **clean image** as inputs.

Figure 10: **Using clean images as inputs improves posterior estimation quality.** We use a pre-trained DDPM diffusion model on CIFAR10 to visualize posterior estimation for clean inputs and noisy inputs across varying noise scales σ_t. Clean inputs produce smooth and descriptive estimations even at high noise levels, whereas noisy inputs result in blurry and lossy estimations at large σ_t, making it difficult to extract meaningful representations.

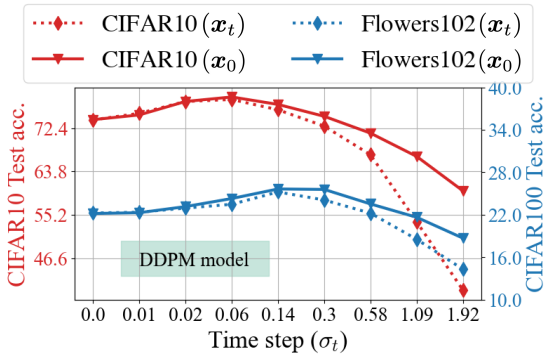


Figure 11: **Performance comparison: clean vs. noisy inputs.** We use pre-trained DDPM/EDM model on the CIFAR10/CIFAR100 datasets and Flowers-102 [73]. The feature probing accuracy is plotted to compare the performance when using clean versus noisy inputs.

from the framework proposed by [49], the input to the classic DDPM model is the same as the input to the UNet inside. Therefore, we calculate the scaling factor $\sqrt{\bar{\alpha}_t} = 1/\sqrt{\sigma^2(t)+1}$, and use $\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0$ as the clean image input. Besides, for noisy input, we set $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}(\boldsymbol{x}_0 + \boldsymbol{n})$, with $\boldsymbol{n} \sim \mathcal{N}\left(\mathbf{0}, \sigma(t)^2\mathbf{I}\right)$. The linear probe results are presented in Figure 11, where we consistently see an unimodal curve, as well as compatible or even superior representation learning performance of clean input $\boldsymbol{x}_0$.

Extend CSNR **on feature representations.** In the main body of the paper, we define CSNR with respect to the posterior estimation function. Given that the intermediate representations of diffusion models exhibit near-linear properties [74], we extend the definition of CSNR to feature extraction functions:

$$\text{CSNR}(t, f_{\boldsymbol{\theta}}) := \mathbb{E}_k \left[\frac{\mathbb{E}_{\boldsymbol{x}_0}[\|\hat{\boldsymbol{U}}_k \hat{\boldsymbol{U}}_k^T f_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)\|^2 \mid k]}{\mathbb{E}_{\boldsymbol{x}_0}[\sum_{l \neq k} \|\hat{\boldsymbol{U}}_l \hat{\boldsymbol{U}}_l^T f_{\boldsymbol{\theta}}(\boldsymbol{x}_0, t)\|^2 \mid k]} \right]$$

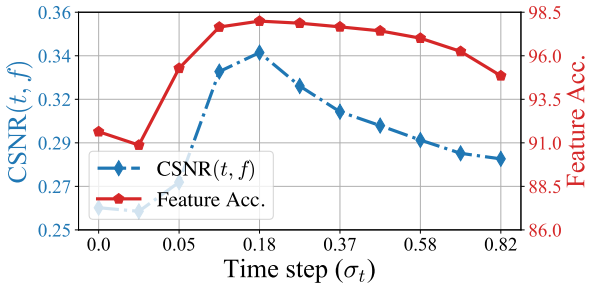
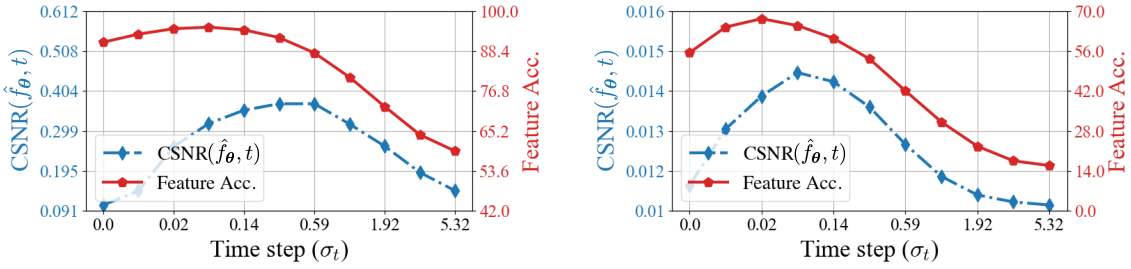


Figure 12: **Dynamics of feature probing accuracy and associated** CSNR **on** `MoLRG` **data** Feature probing accuracy is plotted alongside CSNR$(\hat{f}_{\boldsymbol{\theta}}, t)$. Probing accuracy is evaluated on the test set, while the empirical CSNR is computed from the training set. Both exhibit an aligning unimodal pattern.



(a) CIFAR10 (b) MiniImageNet

Figure 13: **Dynamics of feature probing accuracy and associated** CSNR **on CIFAR10 and MiniImageNet.** Feature probing accuracy is plotted alongside CSNR$(\hat{f}_{\boldsymbol{\theta}}, t)$. Probing accuracy is evaluated on the test set, while the empirical CSNR is computed from the training set. Both exhibit an aligning unimodal pattern.

Here, $f_{\boldsymbol{\theta}}(\cdot, t)$ represents a diffusion feature extraction function that includes all layers up to the feature extraction layer of a diffusion model. The matrix $\hat{\boldsymbol{U}}_k$ denotes the extracted basis corresponding to the correct class of the features, while $\hat{\boldsymbol{U}}_l (l \neq k)$ represents the bases of incorrect classes.

We validate this extension of CSNR as a measure of feature representation quality. Using the same models from Figure 5 and Figure 6, we extract intermediate features at each time step and evaluate classification performance on the test set via linear probing. For CSNR calculation, we compute the basis using features extracted at each time step and subsequently calculate CSNR for the extracted features, denoted as CSNR$(\hat{f}_{\boldsymbol{\theta}}, t)$. The results are presented in Figure 12 and Figure 13 for synthetic and real datasets, respectively. As shown in the plots, CSNR$(f_{\boldsymbol{\theta}}, t)$ consistently follows a unimodal pattern, mirroring the trend of feature probing performance as the noise scale increases.

Validation of $\hat{\boldsymbol{x}}^\star_{approx}$ approximation in Appendix A.4.2. In Theorem 2, we approximate the optimal posterior estimation function $\hat{\boldsymbol{x}}^\star_{\boldsymbol{\theta}}$ using $\hat{\boldsymbol{x}}^\star_{approx}$ by taking the expectation inside the softmax with respect to $\boldsymbol{x}_0$. To validate this approximation, we compare the CSNR calculated from $\hat{\boldsymbol{x}}^\star_{\boldsymbol{\theta}}$ and from $\hat{\boldsymbol{x}}^\star_{approx}$ using the definition in Proposition 1 and (5) in Appendix A.4.2, respectively. We use a fixed dataset size of 2400 and set the default parameters to $n = 50$, $d = 5$, $K = 3$, and $\delta = 0.1$ to generate `MoLRG` data. We then vary one parameter at a time while keeping the others constant, and present the computed CSNR in Figure 14. As shown, the approximated CSNR score consistently aligns with the actual score.

Visualization of the `MoLRG` **posterior estimation and** CSNR **across noise scales.** In Figure 5, we show that both the posterior classification accuracy and CSNR exhibit a unimodal trend for the `MoLRG` data. To further illustrate this behavior, we provide a visualization of the posterior estima-

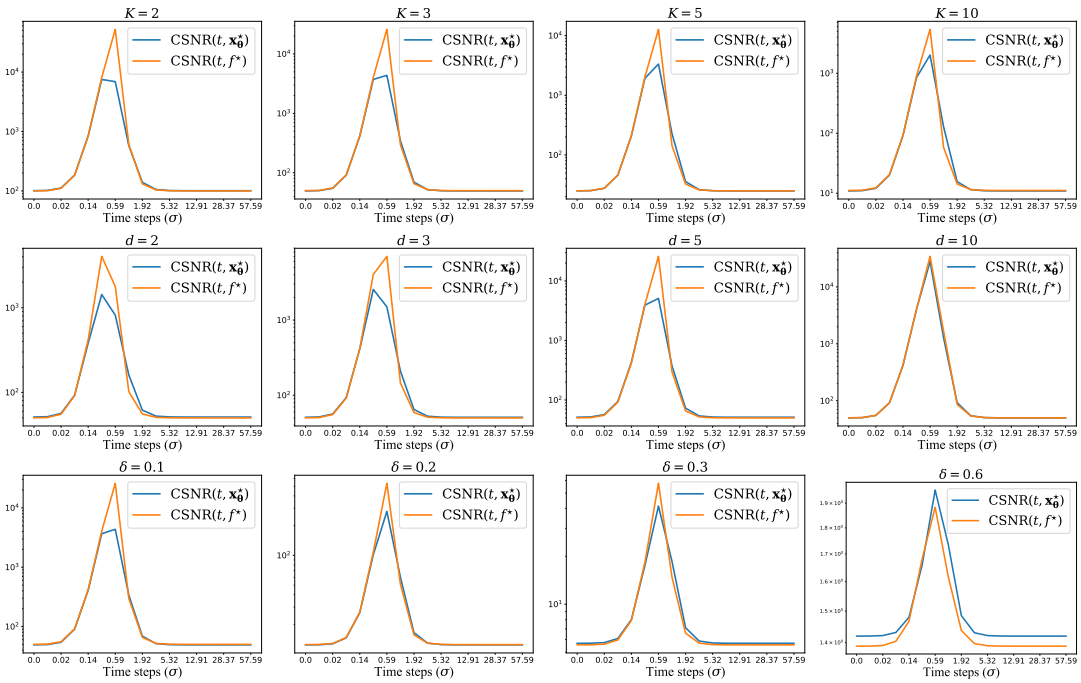


Figure 14: **Comparison between** CSNR **calculated using the optimal model $\hat{x}_{\theta}^{\star}$ and the** CSNR **calculated with our approximation in Theorem 1.** We generate `MoLRG` data and calculate CSNR using both the corresponding optimal posterior function $\hat{x}_{\theta}^{\star}$ and our approximation $\hat{x}_{approx}^{\star}$ from Theorem 1. Default parameters are set as $n = 50$, $d = 5$, $K = 3$, and $\delta = 0.1$. In each row, we vary one parameter while keeping the others fixed, comparing the actual and approximated CSNR.

tion and CSNR at different noise scales in Figure 15. In the plot, each class is represented by a colored straight line, while deviations from these lines correspond to the δ-related noise term. Initially, increasing the noise scale effectively cancels out the δ-related data noise, resulting in a cleaner posterior estimation and improved probing accuracy. However, as the noise continues to increase, the class confidence rate drops, leading to an overlap between classes, which ultimately degrades the feature quality and probing performance.

Mitigating the performance gap between DAE and diffusion models. Throughout the empirical results presented in this paper, we consistently observe a performance gap between individual DAEs and diffusion models, especially in low-noise regions. Here, we use a DAE trained on the CIFAR10 dataset with a single noise level $\sigma = 0.002$, using the NCSN++ architecture [49]. In the default setting, the DAE achieves a test accuracy of 32.3. We then explore three methods to improve the test performance: (a) adding dropout, as noise regularization and dropout have been effective in preventing autoencoders from learning identity functions [60]; (b) adopting EDM-based preconditioning during training, including input/output scaling, loss weighting, etc.; and (c) multi-level noise training, in which the DAE is trained simultaneously on three noise levels $[0.002, 0.012, 0.102]$. Each modification is applied independently, and the results are reported in Table 3. As shown, dropout helps improve performance, but even with a dropout rate of 0.95, the improvement is minor. EDM-based preconditioning achieves moderate improvement, while multi-level noise training yields the most promising results, demonstrating the benefit of incorporating the diffusion process in DAE training.

A.3. Experimental Details

In this section, we provide technical details for all the experiments in the main body of the paper.

Experimental details for Figure 1.

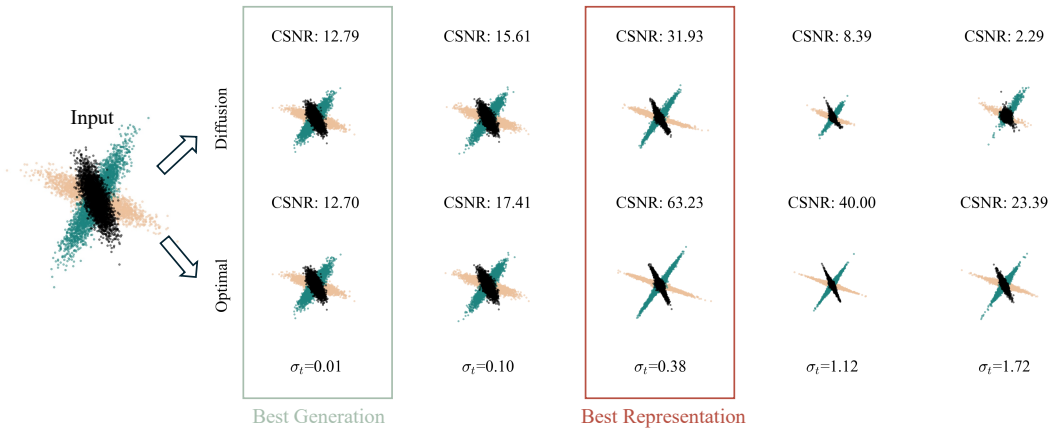


Figure 15: **Visualization of posterior estimation for a clean input, higher** CSNR **correspondings to higher classification accuracy.** The **same** MoLRG data is fed into the models; each row represents a different denoising model, and each column corresponds to a different time step with noise scale (σ_t). The teal box indicates the best generation quality and The red box indicates the best representation quality.

Table 3: **Improve DAE representation performance at low noise region.** A vanilla DAE trained on the CIFAR-10 dataset with a single noise level of $\sigma = 0.002$ serves as the baseline. We evaluate the performance improvement of dropout regularization, EDM-based preconditioning, and multi-level noise training ($\sigma = \{0.002, 0.012, 0.102\}$). Each technique is applied independently to assess its contribution to performance enhancement.

Modifications	Test acc.
Vanilla DAE	32.3
+Dropout (0.5)	35.3
+Dropout (0.9)	36.4
+Dropout (0.95)	38.1
+EDM preconditioning	49.2
+Multi-level noise training	**58.6**

- *Experimental details for Figure 1(a).* We train diffusion models based on the unified framework proposed by [49]. Specifically, we use the DDPM+ network, and use VP configuration for Figure 1(a). [49] has shown equivalence between VP configuration and the traditional DDPM setting, thus we call the models in Figure 3(a) as DDPM* models. We train two models on CIFAR10 and CIFAR100, respectively. After training, we conduct linear probe on CIFAR10 and CIFAR100. At a specific noise level $\sigma(t)$, we either use clean image $\boldsymbol{x}_0$ or noisy image $\boldsymbol{x}_t = \boldsymbol{x}_0 + \boldsymbol{n}$ as input to the DDPM* models for extracting features after the '8x8_block3' layer. Here, $\boldsymbol{n}$ represents random noise and $\boldsymbol{n} \sim \mathcal{N}\left(\mathbf{0}, \sigma(t)^2\mathbf{I}\right)$. We train a logistic regression on features in the train split and report the classification accuracy on the test split of the dataset. We perform the linear probe for each of the following noise levels: [0.002, 0.008, 0.023, 0.060, 0.140, 0.296, 0.585, 1.088, 1.923, 3.257].

- *Experimental details for Figure 1(b).* We exactly follow the protocol in [18], using the same datasets which are subsets of CelebA [75] and FFHQ [76], the same training procedure, and the same segmentation networks (MLPs). The only difference is that we use a newer latent diffusion model [3] pretrained on CelebAHQ from Hugging Face and the noise are added to the latent space. For feature extraction we concatenate the feature from the first layer of each resolution in the UNet's decoder (after upsampling them to the same resolution as the input). We perform segmentation for each of the following noise levels:[0.015, 0.045, 0.079, 0.112, 0.176, 0.342, 0.724, 2.041].

Experimental details for Figure 3. We use a pre-trained EDM CIFAR10 model from the official GitHub repository [49] and extract 10 sets of posterior estimations corresponding to σ_t values ranging from 0.002 to 8.401. MLP probing is applied to these posterior estimations to evaluate posterior accuracy, and FID [77] is computed relative to the original CIFAR10 dataset. For the posterior visualizations in the bottom figure, we randomly select a sample and display its posterior estimations according to the same σ_t schedule.

Experimental details for Figure 5 and Figure 15. For the `MoLRG` experiments, we train a 3-layer MLP with ReLU activation and a hidden dimension of 1024, following the setup provided in an open-source repository [78]. The MLP is trained for 200 epochs using DDPM scheduling with $T = 500$, employing the Adam optimizer with a learning rate of 5×10^{-4}. For feature extraction, we use the activations of the second layer of the MLP (dimension 1024) as intermediate features for linear probing. For CSNR computation, we follow the definition in Section 3.2 since we have access to the ground-truth basis for the `MoLRG` data, i.e., $\boldsymbol{U}_1, \boldsymbol{U}_2$, and $\boldsymbol{U}_3 \in \mathbb{R}^{50\times 15}$. For probing we simply train a linear probe on the posterior and estimations, noting that we take the absolute value of the posterior estimations before feeding them to the probe.

For both panels in Figure 5, we train our probe the same training set used for diffusion and test on five different `MoLRG` datasets generated with five different random seeds, reporting the average accuracy and CSNR at time steps [1, 5, 10, 20, 40, 60, 80, 100, 120, 140, 160, 180, 240, 260, 280]. In Figure 15, we switch to a 3-class `MoLRG` with $d = 1, n = 10, \delta = 0.3$ and visualize the posterior estimations at time steps [1, 20, 80, 200, 260] by projecting them onto the union of $\boldsymbol{U}_1, \boldsymbol{U}_2$, and $\boldsymbol{U}_3$ (a 3D space), then further projecting onto the 2D plane by a random 3×2 matrix with orthonormal columns. The subtitles of each visualization show the corresponding CSNR calculated as explained above.

Experimental details for Figure 6. We use pre-trained EDM models [49] for CIFAR10 and ImageNet, extracting feature representations from the best-performing layer and posterior estimations as the network outputs at each timestep . For the ImageNet model, features are extracted using images and classes from MiniImageNet [52]. Feature accuracy is evaluated via linear probing, while posterior accuracy is assessed using a two-layer MLP, where posterior estimations pass through a linear layer and ReLU activation before the final linear classifier. The bases for the CSNR metric on features are computed via singular value decomposition (SVD) on feature representations at each timestep for each class, followed by CSNR calculation using its definition in Section 3.2. For posterior estimation, we directly use bases $\boldsymbol{U}_k$ derived from raw dataset images. In all cases, the first 5 right singular vectors of each class are used to extract $\boldsymbol{U}_k$.

Experimental details for Figure 7. We train individual DAEs using the DDPM++ network and VP configuration outlined in Karras et al. [49] at the following noise scales:

$$[0.002, 0.008, 0.023, 0.06, 0.14, 0.296, 0.585, 1.088, 1.923, 3.257].$$

Each model is trained for 500 epochs using the Adam optimizer [79] with a fixed learning rate of 1×10^{-4}. For the diffusion models, we reuse the model from Figure $3(\mathrm{d})$. The sliced Wasserstein distance is computed according to the implementation described in Doan et al. [53].

Experimental details for Figure 8 and Figure 9. We use the DDPM++ network and VP configuration to train diffusion models[49] on the CIFAR10 dataset, using two network configurations: UNet-64 and UNet-128, by varying the embedding dimension of the UNet. Training dataset sizes range exponentially from 2^6 to 2^{15}. For each dataset size, both UNet-64 and UNet-128 are trained on the same subset of the training data. All models are trained with a duration of 50K images following the EDM training setup. After training, we calculate the generalization score as described in Zhang et al. [72], using 10K generated images and the full training subset to compute the score.

Experimental details for Table 1 and Table 2 For EDM, we use the official pre-trained checkpoints on ImageNet 64×64 from [49], and for DiT, we use the released DiT-XL/2 model pre-trained on ImageNet 256×256 from [43]. As a baseline, we include the Hugging Face pre-trained MAE encoder (ViT-B/16) [37].

For diffusion models, features are extracted from the layer and timestep that achieve the highest probing accuracy, following [19]. After feature extraction, we adopt the probing protocol from [21], passing the extracted features through a probe consisting of a BatchNorm1d layer followed by a linear layer. To ensure fair comparisons, all input images are cropped or resized to 224×224, matching the resolution used for MAE training.

For ensembling, we extract features from two additional timesteps on either side of the optimal timestep. Independent probes are trained on these timesteps, yielding five probes in total. At test time, we apply a soft-voting ensemble by averaging the output logits from all five probes for the final prediction. Specifically, let $\boldsymbol{W}_t \in \mathbb{R}^{K \times d}$ be the linear classifier trained on features from timestep t, and let $\boldsymbol{h}_t \in \mathbb{R}^d$ denote the feature representation of a sample at timestep t. Considering neighboring timesteps $t-2, t-1, t+1$, and $t+2$, our ensemble prediction is computed as: $\hat{y} = \arg\max \left(\frac{1}{5} \sum_{t=t-2}^{t+2} \boldsymbol{W}_t \boldsymbol{h}_t \right)$.

We evaluate each method under varying levels of label noise, ranging from 0% to 80%, by randomly mislabeling the specified percentage of training labels before applying linear probing. Performance is assessed on both the pre-training dataset and downstream transfer learning tasks. For pre-training evaluation, we use the images and classes from MiniImageNet [52] to reduce computational cost. For transfer learning, we evaluate on CIFAR100 [50], DTD [80], and Flowers102 [73].

A.4. Proofs

A.4.1. Proof of Proposition 1

Proof. We follow the same proof steps as in [32] Lemma 1 with a change of variable. Let $\boldsymbol{c}_k = \begin{bmatrix} \boldsymbol{a}_k \\ \boldsymbol{e}_k \end{bmatrix}$ and $\widetilde{\boldsymbol{U}_k} = \begin{bmatrix} \boldsymbol{U}_k & \delta \boldsymbol{U}_k^{\perp} \end{bmatrix}$, we first compute

$$
\begin{aligned}
& p_t(\boldsymbol{x}|Y=k) \\
&= \int p_t\left(\boldsymbol{x}|Y=k, \boldsymbol{c}_k\right) \mathcal{N}(\boldsymbol{c}_k; \boldsymbol{0}, \boldsymbol{I}_{d+D}) \, \mathrm{d}\boldsymbol{c}_k \\
&= \int p_t(\boldsymbol{x}|\boldsymbol{x}_0 = \widetilde{\boldsymbol{U}_k}\boldsymbol{c}_k) \mathcal{N}\left(\boldsymbol{c}_k; \boldsymbol{0}, \boldsymbol{I}_{d+D}\right) \mathrm{d}\boldsymbol{c}_k \\
&= \int \mathcal{N}(\boldsymbol{x}; s_t\widetilde{\boldsymbol{U}_k}\boldsymbol{c}_k, \gamma_t^2 \boldsymbol{I}_n) \mathcal{N}\left(\boldsymbol{c}_k; \boldsymbol{0}, \boldsymbol{I}_{d+D}\right) \mathrm{d}\boldsymbol{c}_k \\
&= \frac{1}{(2\pi)^{n/2}(2\pi)^{(d+D)/2}\gamma_t^n} \int \exp\left(-\frac{1}{2\gamma_t^2}\|\boldsymbol{x} - s_t\widetilde{\boldsymbol{U}_k}\boldsymbol{c}_k\|^2\right) \exp\left(-\frac{1}{2}\|\boldsymbol{c}_k\|^2\right) \mathrm{d}\boldsymbol{c}_k \\
&= \frac{1}{(2\pi)^{n/2}(2\pi)^{(d+D)/2}\gamma_t^n} \int \exp\left(-\frac{1}{2\gamma_t^2}\left(\boldsymbol{x}^T\boldsymbol{x} - 2s_t\boldsymbol{x}^T\widetilde{\boldsymbol{U}_k}\boldsymbol{c}_k + s_t^2\boldsymbol{c}_k^T\widetilde{\boldsymbol{U}_k}^T\widetilde{\boldsymbol{U}_k}\boldsymbol{c}_k + \gamma_t^2\boldsymbol{c}_k^T\boldsymbol{c}_k\right)\right) \mathrm{d}\boldsymbol{c}_k \\
&= \frac{1}{(2\pi)^{n/2}\gamma_t^n} \left(\frac{s_t^2+\gamma_t^2}{\gamma_t^2}\right)^{-d/2} \left(\frac{s_t^2\delta^2+\gamma_t^2}{\gamma_t^2}\right)^{-D/2} \exp\left(-\frac{1}{2\gamma_t^2}\boldsymbol{x}^T\left(\boldsymbol{I}_n - \frac{s_t^2}{s_t^2+\gamma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T - \frac{s_t^2\delta^2}{s_t^2\delta^2+\gamma_t^2}\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T}\right)\boldsymbol{x}\right) \\
&\int \frac{1}{(2\pi)^{d/2}} \left(\frac{\gamma_t^2}{s_t^2+\gamma_t^2}\right)^{-d/2} \exp\left(-\frac{s_t^2+\gamma_t^2}{2\gamma_t^2}\left\|\boldsymbol{a}_k - \frac{s_t}{s_t^2+\gamma_t^2}\boldsymbol{U}_k^T\boldsymbol{x}\right\|^2\right) \mathrm{d}\boldsymbol{a}_k \\
&\int \frac{1}{(2\pi)^{D/2}} \left(\frac{\gamma_t^2}{s_t^2\delta^2+\gamma_t^2}\right)^{-D/2} \exp\left(-\frac{s_t^2\delta^2+\gamma_t^2}{2\gamma_t^2}\left\|\boldsymbol{e}_k - \frac{s_t\delta}{s_t^2\delta^2+\gamma_t^2}\boldsymbol{U}_k^{\perp T}\boldsymbol{x}\right\|^2\right) \mathrm{d}\boldsymbol{e}_k \\
&= \frac{1}{(2\pi)^{n/2}} \frac{1}{(s_t^2+\gamma_t^2)^{d/2}(s_t^2\delta^2+\gamma_t^2)^{D/2}} \exp\left(-\frac{1}{2\gamma_t^2}\boldsymbol{x}^T\left(\boldsymbol{I}_n - \frac{s_t^2}{s_t^2+\gamma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T - \frac{s_t^2\delta^2}{s_t^2\delta^2+\gamma_t^2}\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T}\right)\boldsymbol{x}\right) \\
&= \frac{1}{(2\pi)^{n/2}\det{}^{1/2}(s_t^2\boldsymbol{U}_k\boldsymbol{U}_k^T + s_t^2\delta^2\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T} + \gamma_t^2\boldsymbol{I}_n)} \\
&\quad \exp\left(-\frac{1}{2}\boldsymbol{x}^T\left(s_t^2\boldsymbol{U}_k\boldsymbol{U}_k^T + s_t^2\delta^2\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T} + \gamma_t^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{x}\right) \\
&= \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2\boldsymbol{U}_k\boldsymbol{U}_k^T + s_t^2\delta^2\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T} + \gamma_t^2\boldsymbol{I}_n),
\end{aligned}
$$

where we repeatedly apply the pdf of multi-variate Gaussian and the second last equality uses $\det(s_t^2\boldsymbol{U}_k\boldsymbol{U}_k^T + s_t^2\delta^2\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T} + \gamma_t^2\boldsymbol{I}_n) = (s_t^2+\gamma_t^2)^d(s_t^2\delta^2+\gamma_t^2)^D$ and $(s_t^2\boldsymbol{U}_k\boldsymbol{U}_k^T + s_t^2\delta^2\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T} + \gamma_t^2\boldsymbol{I}_n)^{-1} = \left(\boldsymbol{I}_n - s_t^2/(s_t^2+\gamma_t^2)\boldsymbol{U}_k\boldsymbol{U}_k^T - s_t^2\delta^2/(s_t^2\delta^2+\gamma_t^2)\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T}\right)/\gamma_t^2$ because of the Woodbury matrix inversion lemma. Hence, with $\mathbb{P}\left(Y=k\right) = \pi_k$ for each $k \in [K]$, we have

$$
p_t(\boldsymbol{x}) = \sum_{k=1}^{K} p_t(\boldsymbol{x}|Y=k)\mathbb{P}(Y=k) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2\boldsymbol{U}_k\boldsymbol{U}_k^T + s_t^2\delta^2\boldsymbol{U}_k^{\perp}\boldsymbol{U}_k^{\perp T} + \gamma_t^2\boldsymbol{I}_n).
$$

Now we can compute the score function

$$\nabla \log p_t(\boldsymbol{x}) = \frac{\nabla p_t(\boldsymbol{x})}{p_t(\boldsymbol{x})} = \frac{\sum_{k=1}^K \pi_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2 \boldsymbol{U}_k \boldsymbol{U}_k^T + s_t^2\delta^2 \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} + \gamma_t^2 \boldsymbol{I}_n) \left(-\frac{1}{\gamma_t^2}\boldsymbol{x} + \frac{s_t^2}{\gamma_t^2(s_t^2+\gamma_t^2)}\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{x} + \frac{s_t^2\delta^2}{\gamma_t^2(s_t^2\delta^2+\gamma_t^2)}\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\boldsymbol{x}\right)}{\sum_{k=1}^K \pi_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2 \boldsymbol{U}_k \boldsymbol{U}_k^T + s_t^2\delta^2 \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} + \gamma_t^2 \boldsymbol{I}_n)}$$

$$= -\frac{1}{\gamma_t^2}\left(\boldsymbol{x} - \frac{\sum_{k=1}^K \pi_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2 \boldsymbol{U}_k \boldsymbol{U}_k^T + s_t^2\delta^2 \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} + \gamma_t^2 \boldsymbol{I}_n) \left(\frac{s_t^2}{s_t^2+\gamma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{x}) + \frac{s_t^2\delta^2}{s_t^2\delta^2+\gamma_t^2}\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\boldsymbol{x})\right)}{\sum_{k=1}^K \pi_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2 \boldsymbol{U}_k \boldsymbol{U}_k^T + s_t^2\delta^2 \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} + \gamma_t^2 \boldsymbol{I}_n)}\right).$$

According to Tweedie's formula, we have

$$\begin{aligned}
\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}_t\right] &= \frac{\boldsymbol{x}_t + \gamma_t^2 \nabla \log p_t(\boldsymbol{x}_t)}{s_t} \\
&= \frac{s_t}{s_t^2+\gamma_t^2}\frac{\sum_{k=1}^K \pi_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2 \boldsymbol{U}_k \boldsymbol{U}_k^T + s_t^2\delta^2 \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} + \gamma_t^2 \boldsymbol{I}_n)\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{x}}{\mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2 \boldsymbol{U}_k \boldsymbol{U}_k^T + s_t^2\delta^2 \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} + \gamma_t^2 \boldsymbol{I}_n)} \\
&\quad + \frac{s_t\delta^2}{s_t^2\delta^2+\gamma_t^2}\frac{\sum_{k=1}^K \pi_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2 \boldsymbol{U}_k \boldsymbol{U}_k^T + s_t^2\delta^2 \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} + \gamma_t^2 \boldsymbol{I}_n)\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\boldsymbol{x}}{\mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, s_t^2 \boldsymbol{U}_k \boldsymbol{U}_k^T + s_t^2\delta^2 \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} + \gamma_t^2 \boldsymbol{I}_n)} \\
&= \frac{s_t}{s_t^2+\gamma_t^2}\frac{\sum_{k=1}^K \pi_k \exp\left(\phi_t\|\boldsymbol{U}_k^T\boldsymbol{x}_t\|^2\right)\exp\left(\psi_t\|\boldsymbol{U}_k^{\perp T}\boldsymbol{x}_t\|^2\right)\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{x}_t}{\sum_{k=1}^K \pi_k \exp\left(\phi_t\|\boldsymbol{U}_k^T\boldsymbol{x}_t\|^2\right)\exp\left(\psi_t\|\boldsymbol{U}_k^{\perp T}\boldsymbol{x}_t\|^2\right)} \\
&\quad + \frac{s_t\delta^2}{s_t^2\delta^2+\gamma_t^2}\frac{\sum_{k=1}^K \pi_k \exp\left(\phi_t\|\boldsymbol{U}_k^T\boldsymbol{x}_t\|^2\right)\exp\left(\psi_t\|\boldsymbol{U}_k^{\perp T}\boldsymbol{x}_t\|^2\right)\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\boldsymbol{x}_t}{\sum_{k=1}^K \pi_k \exp\left(\phi_t\|\boldsymbol{U}_k^T\boldsymbol{x}_t\|^2\right)\exp\left(\psi_t\|\boldsymbol{U}_k^{\perp T}\boldsymbol{x}_t\|^2\right)},
\end{aligned}$$

with $\phi_t = s_t^2/(2\gamma_t^2(s_t^2+\gamma_t^2))$ and $\psi_t = s_t^2\delta^2/(2\gamma_t^2(s_t^2\delta^2+\gamma_t^2))$. The final equality uses the pdf of multi-variant Gaussian and the matrix inversion lemma discussed earlier.

Now since π_k is consistent for all k and $s_t = 1$, we have

$$\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}_t\right] = \sum_{k=1}^K w_k^\star(\boldsymbol{x}_t)\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T + \frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\right)\boldsymbol{x}_t$$

$$\text{where } w_k^\star(\boldsymbol{x}_t) := \frac{\exp\left(\frac{1}{2\sigma_t^2(1+\sigma_t^2)}\|\boldsymbol{U}_k^T\boldsymbol{x}_t\|^2 + \frac{\delta^2}{2\sigma_t^2(\delta^2+\sigma_t^2)}\|\boldsymbol{U}_k^{\perp T}\boldsymbol{x}_t\|^2\right)}{\sum_{k=1}^K \exp\left(\frac{1}{2\sigma_t^2(1+\sigma_t^2)}\|\boldsymbol{U}_k^T\boldsymbol{x}_t\|^2 + \frac{\delta^2}{2\sigma_t^2(\delta^2+\sigma_t^2)}\|\boldsymbol{U}_k^{\perp T}\boldsymbol{x}_t\|^2\right)}.$$

□

A.4.2. Proof of Theorem 1

We first state the formal version of Theorem 1.

To simplify the calculation of CSNR as introduced in Section 3.2 on posterior estimation, which involves the expectation over the softmax term $w_k^\star$, we approximate $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}^\star$ as follows:

$$\begin{aligned}
\hat{\boldsymbol{x}}_{approx}^\star(\boldsymbol{x}, t) &= \sum_{k=1}^K \hat{w}_k\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T + \frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\right)\boldsymbol{x}, \\
&\text{where } \hat{w}_k := \frac{\exp\left(\mathbb{E}_{\boldsymbol{x}_0}[g_k(\boldsymbol{x}_0, t)]\right)}{\sum_{k=1}^K \exp\left(\mathbb{E}_{\boldsymbol{x}_0}[g_k(\boldsymbol{x}_0, t)]\right)}.
\end{aligned} \tag{5}$$

In other words, we use $\hat{w}_k$ in (5) to approximate $w_k^\star(\boldsymbol{x}_0)$ in Proposition 1 by taking expectation inside the softmax with respect to $\boldsymbol{x}_0$. This allows us to treat $\hat{w}_k$ as a constant when calculating CSNR, making the analysis more tractable while maintaining $\mathbb{E}[\|\boldsymbol{U}_l\boldsymbol{U}_l^T\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}^\star(\boldsymbol{x}_0, t)\|^2] \approx \mathbb{E}[\|\boldsymbol{U}_l\boldsymbol{U}_l^T\hat{\boldsymbol{x}}_{approx}^\star(\boldsymbol{x}_0, t)\|^2]$

for all $l \in [K]$. We verify the tightness of this approximation at Appendix A.2 (Figure 14). With this approximation, we state the theorem as follows:

Theorem 2. *Let data $\boldsymbol{x}_0$ be any arbitrary data point drawn from the `MoLRG` distribution defined in Assumption 1 and let k denote the true class $\boldsymbol{x}_0$ belongs to. Then* CSNR *introduced in Section 3.2 depends on the noise level σ_t in the following form:*

$$\mathrm{CSNR}(\hat{\boldsymbol{x}}^\star_{approx}, t) = \frac{1}{(K-1)\delta^2} \cdot \left(\frac{1 + \frac{\sigma_t^2}{\delta^2} h(\hat{w}_k, \delta)}{1 + \frac{\sigma_t^2}{\delta^2} h(\hat{w}_l, \delta)} \right)^2 \tag{6}$$

where $h(w, \delta) := (1-\delta^2)w + \delta^2$. Since δ is fixed, $h(w, \delta)$ is a monotonically increasing function with respect to w. Note that here δ represents the magnitude of the fixed intrinsic noise in the data where σ_t denotes the level of additive Gaussian noise introduced during the diffusion training process.

Proof. Following the definition of CSNR as defined in Section 3.2, Lemma 1 and the fact that $k \sim \mathrm{Mult}(K, \pi_k)$ with $\pi_1 = \cdots = \pi_K = 1/K$, we can write

$$\begin{aligned}
\mathrm{CSNR}(\hat{\boldsymbol{x}}^\star_{approx}, t) &= \frac{\mathbb{E}_{\boldsymbol{x}_0}[\|\boldsymbol{U}_k \boldsymbol{U}_k^T \hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2]}{\mathbb{E}_{\boldsymbol{x}_0}[\sum_{l \neq k} \|\boldsymbol{U}_l \boldsymbol{U}_l^T \hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2]} = \frac{\mathbb{E}_{\boldsymbol{x}_0}[\|\boldsymbol{U}_k \boldsymbol{U}_k^T \hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2]}{\sum_{l \neq k} \mathbb{E}_{\boldsymbol{x}_0}[\|\boldsymbol{U}_l \boldsymbol{U}_l^T \hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2]} \\
&= \frac{\left(\frac{\hat{w}_k}{1+\sigma_t^2} + \frac{(K-1)\delta^2 \hat{w}_l}{\delta^2 + \sigma_t^2} \right)^2 d}{(K-1)\left(\frac{\hat{w}_l}{1+\sigma_t^2} + \frac{\delta^2(\hat{w}_k + (K-2)\hat{w}_l)}{\delta^2 + \sigma_t^2} \right)^2 \delta^2 d} \\
&= \frac{1}{(K-1)\delta^2} \cdot \left(\frac{\hat{w}_k \delta^2 + \hat{w}_k \sigma_t^2 + (K-1)\delta^2 \hat{w}_l + (K-1)\delta^2 \hat{w}_l \sigma_t^2}{\hat{w}_l \delta^2 + \hat{w}_l \sigma_t^2 + \delta^2 \hat{w}_k + (K-2)\delta^2 \hat{w}_l + \delta^2 \hat{w}_k \sigma_t^2 + (K-2)\delta^2 \hat{w}_l \sigma_t^2} \right)^2 \\
&= \frac{1}{(K-1)\delta^2} \cdot \left(\frac{\delta^2 + \sigma_t^2 \left(\hat{w}_k + (K-1)\delta^2 \hat{w}_l \right)}{\delta^2 + \sigma_t^2 \left(\hat{w}_l + \delta^2 \hat{w}_k + (K-2)\delta^2 \hat{w}_l \right)} \right)^2 \\
&= \frac{1}{(K-1)\delta^2} \cdot \left(\frac{1 + \frac{\sigma_t^2}{\delta^2} \left((1-\delta^2)\hat{w}_k + \delta^2(\hat{w}_k + (K-1)\hat{w}_l) \right)}{1 + \frac{\sigma_t^2}{\delta^2} \left((1-\delta^2)\hat{w}_l + \delta^2(\hat{w}_l + \hat{w}_k + (K-2)\hat{w}_l) \right)} \right)^2 \\
&= \frac{1}{(K-1)\delta^2} \cdot \left(\frac{1 + \frac{\sigma_t^2}{\delta^2} \left((1-\delta^2)\hat{w}_k + \delta^2 \right)}{1 + \frac{\sigma_t^2}{\delta^2} \left((1-\delta^2)\hat{w}_l + \delta^2 \right)} \right)^2 \\
&= \frac{1}{(K-1)\delta^2} \cdot \left(\frac{1 + \frac{\sigma_t^2}{\delta^2} h(\hat{w}_k, \delta)}{1 + \frac{\sigma_t^2}{\delta^2} h(\hat{w}_l, \delta)} \right)^2
\end{aligned}$$

where $h(w, \delta) := (1 - \delta^2)w + \delta^2$. □

Lemma 1. *With the set up of a K-class `MoLRG` data distribution as defined in* (3)*, and define the noise space as $\boldsymbol{U}_\perp = \bigcap_{k=1}^K \boldsymbol{U}_k^\perp \in \mathcal{O}^{n \times (n-Kd)}$ (i.e., mutual noise for all classes). Consider the following the function:*

$$\hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}, t) = \sum_{k=1}^K \hat{w}_k(\boldsymbol{x}) \left(\frac{1}{1+\sigma_t^2} \boldsymbol{U}_k \boldsymbol{U}_k^T + \frac{\delta^2}{\delta^2 + \sigma_t^2} \boldsymbol{U}_k^\perp \boldsymbol{U}_k^{\perp T} \right) \boldsymbol{x}, \tag{7}$$

$$\textit{where } \hat{w}_k(\boldsymbol{x}) := \frac{\exp\left(\mathbb{E}_{\boldsymbol{x}}[g_k(\boldsymbol{x}, t)]\right)}{\sum_{k=1}^K \exp\left(\mathbb{E}_{\boldsymbol{x}}[g_k(\boldsymbol{x}, t)]\right)}, \tag{8}$$

$$\textit{and } g_k(\boldsymbol{x}) = \frac{1}{2\sigma_t^2(1+\sigma_t^2)} \|\boldsymbol{U}_k^T \boldsymbol{x}\|^2 + \frac{\delta^2}{2\sigma_t^2(\delta^2 + \sigma_t^2)} \|\boldsymbol{U}_k^{\perp T} \boldsymbol{x}\|^2. \tag{9}$$

I.e., we consider a simplified version of the expected posterior mean as in Proposition 1 by taking expectation of $g_k(\boldsymbol{x})$ prior to the softmax operation. Under this setting, for any clean $\boldsymbol{x}_0$ from class k (i.e., $\boldsymbol{x}_0 = \boldsymbol{U}_k \boldsymbol{a}_i + b\boldsymbol{U}_k^{\perp}\boldsymbol{e}_i$), we have:

$$\mathbb{E}_{\boldsymbol{x}_0}[\|\boldsymbol{U}_k\boldsymbol{U}_k^T\hat{\boldsymbol{x}}^{\star}_{approx}(\boldsymbol{x}_0,t)\|^2] = \left(\frac{\hat{w}_k}{1+\sigma_t^2} + \frac{(K-1)\delta^2\hat{w}_l}{\delta^2+\sigma_t^2}\right)^2 d \tag{10}$$

$$\mathbb{E}_{\boldsymbol{x}_0}[\|\boldsymbol{U}_l\boldsymbol{U}_l^T\hat{\boldsymbol{x}}^{\star}_{approx}(\boldsymbol{x}_0,t)\|^2] = \left(\frac{\hat{w}_l}{1+\sigma_t^2} + \frac{\delta^2(\hat{w}_k+(K-2)\hat{w}_l)}{\delta^2+\sigma_t^2}\right)^2 \delta^2 d \tag{11}$$

$$\mathbb{E}_{\boldsymbol{x}_0}[\|\boldsymbol{U}_{\perp}\boldsymbol{U}_{\perp}^T\hat{\boldsymbol{x}}^{\star}_{approx}(\boldsymbol{x}_0,t)\|^2] = \frac{\delta^6(n-kd)}{(\delta^2+\sigma_t^2)^2} \tag{12}$$

$$\begin{aligned}\mathbb{E}[\|\hat{\boldsymbol{x}}^{\star}_{approx}(\boldsymbol{x}_0,t)\|^2] = &\underbrace{\left(\frac{\hat{w}_k}{1+\sigma_t^2} + \frac{(K-1)\delta^2\hat{w}_l}{\delta^2+\sigma_t^2}\right)^2 d}_{\mathbb{E}\,\|\boldsymbol{U}_k\boldsymbol{U}_k^T\hat{\boldsymbol{x}}^{\star}_{approx}(\boldsymbol{x}_0,t)\|^2]} \\ &+ \underbrace{(K-1)\left(\frac{\hat{w}_l}{1+\sigma_t^2} + \frac{\delta^2(\hat{w}_k+(K-2)\hat{w}_l)}{\delta^2+\sigma_t^2}\right)^2 \delta^2 d}_{\mathbb{E}[\sum_{l\neq k}^{K}\boldsymbol{U}_l\boldsymbol{U}_l^T\hat{\boldsymbol{x}}^{\star}_{approx}(\boldsymbol{x}_0,t)\|^2]} + \underbrace{\frac{\delta^6(n-Kd)}{(\delta^2+\sigma_t^2)^2}}_{\mathbb{E}[\|\boldsymbol{U}_{\perp}\boldsymbol{U}_{\perp}^T\hat{\boldsymbol{x}}^{\star}_{approx}(\boldsymbol{x}_0,t)\|^2]}\end{aligned} \tag{13}$$

and

$$\begin{aligned}\hat{w}_k := \hat{w}_k(\boldsymbol{x}_0) &= \frac{\exp\left(\frac{d}{2\sigma_t^2(1+\sigma_t^2)} + \frac{\delta^4 D}{2\sigma_t^2(\delta^2+\sigma_t^2)}\right)}{\exp\left(\frac{d}{2\sigma_t^2(1+\sigma_t^2)} + \frac{\delta^4 D}{2\sigma_t^2(\delta^2+\sigma_t^2)}\right) + (K-1)\exp\left(\frac{\delta^2 d}{2\sigma_t^2(1+\sigma_t^2)} + \frac{\delta^2 d+\delta^4(D-d)}{2\sigma_t^2(\delta^2+\sigma_t^2)}\right)}, \\ \hat{w}_l := \hat{w}_l(\boldsymbol{x}_0) &= \frac{\exp\left(\frac{\delta^2 d}{2\sigma_t^2(1+\sigma_t^2)} + \frac{\delta^2 d+\delta^4(D-d)}{2\sigma_t^2(\delta^2+\sigma_t^2)}\right)}{\exp\left(\frac{d}{2\sigma_t^2(1+\sigma_t^2)} + \frac{\delta^4 D}{2\sigma_t^2(\delta^2+\sigma_t^2)}\right) + (K-1)\exp\left(\frac{\delta^2 d}{2\sigma_t^2(1+\sigma_t^2)} + \frac{\delta^2 d+\delta^4(D-d)}{2\sigma_t^2(\delta^2+\sigma_t^2)}\right)}\end{aligned} \tag{14}$$

for all class index $l \neq k$.

Proof. Throughout the proof, we use the following notation for slices of vectors.

$$\boldsymbol{e}_i[a:b] \qquad \text{Slices of vector } \boldsymbol{e}_i \text{ from } a\text{th entry to } b\text{th entry.}$$

We begin with the softmax terms. Since each class has its unique disjoint subspace, it suffices to consider $g_k(\boldsymbol{x}_0,t)$ and $g_l(\boldsymbol{x}_0,t)$ for any $l \neq k$. Let $a_t = \frac{1}{2\sigma_t^2(1+\sigma_t^2)}$ and $c_t = \frac{\delta^2}{2\sigma_t^2(\delta^2+\sigma_t^2)}$, we have:

$$\begin{aligned}\mathbb{E}[g_k(\boldsymbol{x}_0,t)] &= \mathbb{E}[a_t\|\boldsymbol{U}_k^T\boldsymbol{x}_0\|^2 + c_t\|\boldsymbol{U}_k^{\perp T}\boldsymbol{x}_0\|^2] \\ &= \mathbb{E}[a_t\|\boldsymbol{U}_k^T(\boldsymbol{U}_k\boldsymbol{a}_i + b\boldsymbol{U}_k^{\perp}\boldsymbol{e}_i)\|^2] + \mathbb{E}[c_t\|\boldsymbol{U}_k^{\perp T}(\boldsymbol{U}_k\boldsymbol{a}_i + b\boldsymbol{U}_k^{\perp}\boldsymbol{e}_i)\|^2] \\ &= \mathbb{E}[a_t\|\boldsymbol{a}_i\|^2] + \mathbb{E}[c_t\|b\boldsymbol{e}_i\|^2] \\ &= a_t d + c_t\delta^2 D\end{aligned}$$

where the last equality follows from $\boldsymbol{a}_i \overset{i.i.d.}{\sim} \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ and $\boldsymbol{e}_i \overset{i.i.d.}{\sim} \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_D)$.

Without loss of generality, assume the $j = k+1$, we have:

$$\begin{aligned}\mathbb{E}[g_l(\boldsymbol{x}_0,t)] &= \mathbb{E}[a_t\|\boldsymbol{U}_l^T\boldsymbol{x}_0\|^2 + c_t\|\boldsymbol{U}_l^{\perp T}\boldsymbol{x}_0\|^2] \\ &= \mathbb{E}[a_t\|\boldsymbol{U}_l^T(\boldsymbol{U}_k\boldsymbol{a}_i + b\boldsymbol{U}_k^{\perp}\boldsymbol{e}_i)\|^2] + \mathbb{E}[c_t\|\boldsymbol{U}_l^{\perp T}(\boldsymbol{U}_k\boldsymbol{a}_i + b\boldsymbol{U}_k^{\perp}\boldsymbol{e}_i)\|^2] \\ &= \mathbb{E}[a_t\|b\boldsymbol{e}_i[1:d]\|^2] + \mathbb{E}\left[c_t\left\|\begin{bmatrix}\boldsymbol{a}_i \\ \boldsymbol{0}\in\mathbb{R}^{D-d}\end{bmatrix} + b\begin{bmatrix}\boldsymbol{0}\in\mathbb{R}^d \\ \boldsymbol{e}_i[d:D]]\end{bmatrix}\right\|^2\right] \\ &= a_t\delta^2 d + c_t(d+\delta^2(D-d))\end{aligned}$$

Plug a_t and b_t back with the exponentials, we get $\hat{w}_k$ and $\hat{w}_l$.

Now we prove (10):

$$\begin{aligned}
\boldsymbol{U}_k\boldsymbol{U}_k^T\hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0,t) &= \hat{w}_k\boldsymbol{U}_k\boldsymbol{U}_k^T\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T+\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\right)\boldsymbol{x}_0 \\
&\quad+\sum_{l\neq k}\hat{w}_l\boldsymbol{U}_k\boldsymbol{U}_k^T\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_l\boldsymbol{U}_l^T+\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_l^\perp\boldsymbol{U}_l^{\perp T}\right)\boldsymbol{x}_0 \\
&= \hat{w}_k\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{x}_0\right)+\sum_{l\neq k}\hat{w}_l\left(\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{x}_0\right) \\
&= \left(\frac{\hat{w}_k}{1+\sigma_t^2}+\frac{(K-1)\,\delta^2\hat{w}_l}{\delta^2+\sigma_t^2}\right)\boldsymbol{U}_k\boldsymbol{U}_k^T(\boldsymbol{U}_k\boldsymbol{a}_i+b\boldsymbol{U}_k^\perp\boldsymbol{e}_i) \\
&= \left(\frac{\hat{w}_k}{1+\sigma_t^2}+\frac{(K-1)\,\delta^2\hat{w}_l}{\delta^2+\sigma_t^2}\right)\boldsymbol{U}_k\boldsymbol{a}_i
\end{aligned}$$

Since $\boldsymbol{U}_k\in\mathcal{O}^{n\times d}$:

$$\mathbb{E}[\|\boldsymbol{U}_k\boldsymbol{U}_k^T\hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0,t)\|^2]=\left(\frac{\hat{w}_k}{1+\sigma_t^2}+\frac{(K-1)\,\delta^2\hat{w}_l}{\delta^2+\sigma_t^2}\right)^2 d$$

and similarly for (11):

$$\begin{aligned}
\boldsymbol{U}_l\boldsymbol{U}_l^T\hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0,t) &= \hat{w}_k\boldsymbol{U}_l\boldsymbol{U}_l^T\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T+\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\right)\boldsymbol{x}_0 \\
&\quad+\hat{w}_l\boldsymbol{U}_l\boldsymbol{U}_l^T\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_l\boldsymbol{U}_l^T+\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_l^\perp\boldsymbol{U}_l^{\perp T}\right)\boldsymbol{x}_0 \\
&\quad+\sum_{j\neq k,l}\hat{w}_j\boldsymbol{U}_l\boldsymbol{U}_l^T\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_j\boldsymbol{U}_j^T+\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_j^\perp\boldsymbol{U}_j^{\perp T}\right)\boldsymbol{x}_0 \\
&= \hat{w}_k\left(\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_l\boldsymbol{U}_l^T\boldsymbol{x}_0\right)+\hat{w}_l\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_l\boldsymbol{U}_l^T\boldsymbol{x}_0\right)+\sum_{j\neq k,l}\hat{w}_j\left(\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_l\boldsymbol{U}_l^T\boldsymbol{x}_0\right) \\
&= \left(\frac{\hat{w}_l}{1+\sigma_t^2}+\frac{\delta^2(\hat{w}_k+(K-2)\hat{w}_j)}{\delta^2+\sigma_t^2}\right)\boldsymbol{U}_l\boldsymbol{U}_l^T(\boldsymbol{U}_k\boldsymbol{a}_i+b\boldsymbol{U}_k^\perp\boldsymbol{e}_i) \\
&= \left(\frac{\hat{w}_l}{1+\sigma_t^2}+\frac{\delta^2(\hat{w}_k+(K-2)\hat{w}_l)}{\delta^2+\sigma_t^2}\right)b\boldsymbol{U}_l\boldsymbol{e}_i[1:d]
\end{aligned}$$

where the third equality follows since $\hat{w}_j=\hat{w}_l$ for all $j\neq k,l$. Further, we have:

$$\mathbb{E}[\|\boldsymbol{U}_l\boldsymbol{U}_l^T\hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0,t)\|^2]=\left(\frac{\hat{w}_l}{1+\sigma_t^2}+\frac{\delta^2(\hat{w}_k+(K-2)\hat{w}_l)}{\delta^2+\sigma_t^2}\right)^2\delta^2 d$$

Next, we consider (12):

$$\begin{aligned}
\boldsymbol{U}_\perp\boldsymbol{U}_\perp^T\hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0,t) &= \hat{w}_k\boldsymbol{U}_\perp\boldsymbol{U}_\perp^T\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_k\boldsymbol{U}_k^T+\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_k^\perp\boldsymbol{U}_k^{\perp T}\right)\boldsymbol{x}_0 \\
&\quad+\sum_{l\neq k}\hat{w}_l\boldsymbol{U}_\perp\boldsymbol{U}_\perp^T\left(\frac{1}{1+\sigma_t^2}\boldsymbol{U}_l\boldsymbol{U}_l^T+\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_l^\perp\boldsymbol{U}_l^{\perp T}\right)\boldsymbol{x}_0 \\
&= \hat{w}_k\left(\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_\perp\boldsymbol{U}_\perp^T\boldsymbol{x}_0\right)+\sum_{l\neq k}\hat{w}_l\left(\frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_\perp\boldsymbol{U}_\perp^T\boldsymbol{x}_0\right) \\
&= \frac{\delta^2}{\delta^2+\sigma_t^2}\boldsymbol{U}_\perp\boldsymbol{U}_\perp^T(\boldsymbol{U}_k\boldsymbol{a}_i+b\boldsymbol{U}_k^\perp\boldsymbol{e}_i) \\
&= \frac{\delta^3}{\delta^2+\sigma_t^2}\boldsymbol{U}_\perp\boldsymbol{e}_i[(K-1)d:D]
\end{aligned}$$

Hence:

$$\mathbb{E}[\|\boldsymbol{U}_\perp \boldsymbol{U}_\perp^T \hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2] = \frac{\delta^6(n - Kd)}{(\delta^2 + \sigma_t^2)^2}$$

Lastly, we prove (13). Given that the subspaces of all classes and the complement space are both orthonormal and mutually orthogonal, we can write:

$$\mathbb{E}[\|\hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2] = \mathbb{E}[\|\boldsymbol{U}_k \boldsymbol{U}_k^T \hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2] + \mathbb{E}[\sum_{l \neq k} \|\boldsymbol{U}_l \boldsymbol{U}_l^T \hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2] + \mathbb{E}[\|\boldsymbol{U}_\perp \boldsymbol{U}_\perp^T \hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2]$$

Combine terms, we get:

$$\begin{aligned}\mathbb{E}[\|\hat{\boldsymbol{x}}^\star_{approx}(\boldsymbol{x}_0, t)\|^2] = &\left(\frac{\hat{w}_k}{1 + \sigma_t^2} + \frac{(K-1)\delta^2 \hat{w}_l}{\delta^2 + \sigma_t^2}\right)^2 d \\ &+ (K-1)\left(\frac{\hat{w}_l}{1 + \sigma_t^2} + \frac{\delta^2(\hat{w}_k + (K-2)\hat{w}_l)}{\delta^2 + \sigma_t^2}\right)^2 \delta^2 d + \frac{\delta^6(n - Kd)}{(\delta^2 + \sigma_t^2)^2}.\end{aligned}$$

□