PDF Download
3711896.3737125.pdf
29 January 2026
Total Citations: 0
Total Downloads: 1339

Published: 03 August 2025

Citation in BibTeX format

KDD '25: The 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining
August 3 - 7, 2025
Toronto ON, Canada

Conference Sponsors:
SIGMOD
SIGKDD

RESEARCH-ARTICLE

SepDiff: Self-Encoding Parameter Diffusion for Learning Latent Semantics

ZHANGKAI WU, Macquarie University, Sydney, NSW, Australia

XUHUI FAN, Macquarie University, Sydney, NSW, Australia

JIN LI, University of Technology Sydney, Sydney, NSW, Australia

ZHILIN ZHAO, Sun Yat-Sen University, Guangzhou, Guangdong, China

HUI CHEN, Macquarie University, Sydney, NSW, Australia

LONGBING CAO, Macquarie University, Sydney, NSW, Australia

Open Access Support provided by:

Macquarie University

Sun Yat-Sen University

University of Technology Sydney

KDD '25: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (August 2025)
https://doi.org/10.1145/3711896.3737125
ISBN: 9798400714542

SepDiff: Self-Encoding Parameter Diffusion for Learning Latent Semantics

Zhangkai Wu
Macquarie University
University of Technology Sydney
Sydney, Australia
zhangkai.wu@mq.edu.au

Xuhui Fan
Macquarie University
Sydney, Australia
xuhui.fan@mq.edu.au

Jin Li
University of Technology Sydney
Sydney, Australia
jin.li-4@student.uts.edu.au

Zhilin Zhao
Sun Yat-sen University
Guangzhou, China
zhaozhlin@mail.sysu.edu.cn

Hui Chen
Macquarie University
Sydney, Australia
hui.chen2@students.mq.edu.au

Longbing Cao*
Macquarie University
Sydney, Australia
longbing.cao@mq.edu.au

Abstract

The recently proposed Bayesian Flow Networks (BFNs) show great potential in modeling parameter spaces via a diffusion process, offering a unified strategy for handling continuous, discrete data. However, these parameter diffusion models cannot learn high-level semantic representation from the parameter space since common encoders, which encode data into one static representation, cannot capture semantic changes in parameters. This motivates a new direction: learning semantic representations hidden in the parameter spaces to characterize noisy data. Accordingly, we propose a representation learning framework named SepDiff which operates in the parameter space to obtain parameter-wise latent semantics that exhibit progressive structures. Specifically, SepDiff proposes a *self*-encoder to learn latent semantics directly from parameters, rather than from observations. The encoder is then integrated into parameter diffusion model, enabling representation learning with various formats of observations. Mutual information terms further promote the disentanglement of latent semantics and capture meaningful semantics simultaneously. We illustrate seven representation learning tasks in SepDiff via expanding this parameter diffusion model, and extensive quantitative experimental results demonstrate the superior effectiveness of SepDiff in learning parameter representation.

CCS Concepts

• **Diffusion models, autoencoder, representation learning, hierarchical structures**;

Keywords

Diffusion; Representation Learning

[*] Corresponding author.

KDD '25, Toronto, ON, Canada

ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3737125

1 Introduction

Representation learning [5], which aims at learning low-dimensional latent semantics from high-dimensional observations, offers an unsupervised approach to discovering high-level semantics in observations. It has been widely applied in areas such as computer vision [12, 26, 63], and data analytics [33, 45, 51]. While most representation learning methods [9, 21, 31, 52] work on continuous-valued observations, different non-trivial methods are needed to discover semantics for discrete data [2, 10, 36, 48]. Consequently, these individual efforts might face issues such as inconsistent discoveries within the data [66] or repeated modelling efforts [24, 62].

On the other hand, Bayesian Flow Networks (BFNs) [15, 42, 56] have been recently proposed as promising Parameter Diffusion Models (**PDMs**). By operating in the parameter space, PDMs design a multi-step mechanism to approximate the ground-truth parameters of observation sequentially. As a result, a uniform strategy may be adopted to deal with continuous, discrete data while simultaneously maintaining fast sampling. Pilot studies of PDMs have shown promising results in modelling different data formats.

Leveraging PDMs , this paper introduces SepDiff , a novel parameter space representation learning framework that employs a unified strategy to extract meaningful high-level semantics from continuous and discrete data. Specifically, a *self*-encoder is designed to encode step-wise parameters into low dimensional semantic latents, capturing gradual semantic changes throughout the multi-step generation process. These semantic latents are then integrated into a neural network architecture to form the parameters for an output distribution that simulates observations. Furthermore, mutual information is introduced to enhance the disentanglement of latent semantics, promoting the capture of distinct and meaningful representations.

SepDiff is applied on benchmark datasets and verifies its effectiveness in obtaining meaningful high-level semantics for discrete and continuous-valued observations. Sampling and reverse-sampling procedures are developed here to complete conditional image reconstruction and generation tasks. In particular, our developed self-encoder discovers interesting progressive semantics along with the flow steps. That is, our SepDiff obtains meaningful,clearer disentangled representations while maintaining high sample quality.

The main contributions of this work can be summarized as follows: (1) A parameter space representation learning framework

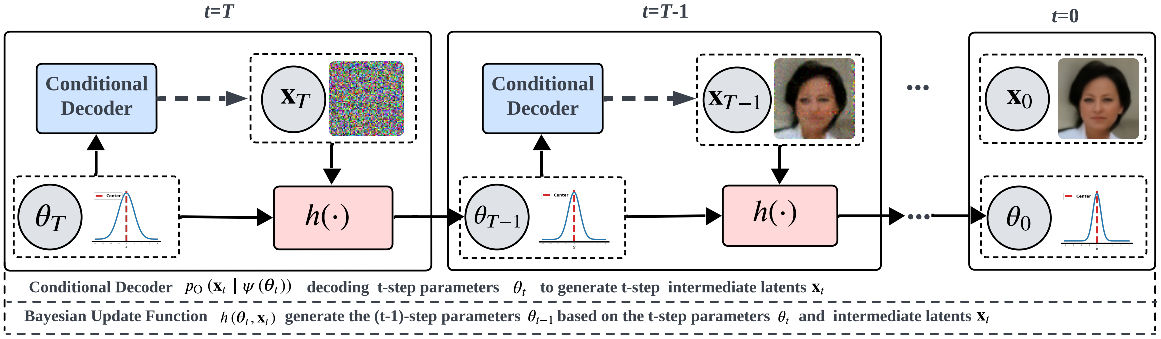

t=T
t=T-1
t=0
Conditional Decoder
Conditional Decoder
$\mathbf{x}_T$
$\mathbf{x}_{T-1}$
$\mathbf{x}_0$
θ_T
θ_{T-1}
θ_0
Center
$h(\cdot)$
$h(\cdot)$
Conditional Decoder $p_O(\mathbf{x}_t \mid \psi(\theta_t))$ decoding t-step parameters θ_t to generate t-step intermediate latents $\mathbf{x}_t$
Bayesian Update Function $h(\theta_t, \mathbf{x}_t)$ generate the (t-1)-step parameters θ_{t-1} based on the t-step parameters θ_t and intermediate latents $\mathbf{x}_t$

Figure 1: Our understanding of PDMs serves as an alternative notation for vanilla BFNs. Each step consists of a conditional decoder $p_O(\mathrm{x}_t|\psi(\theta_t)$ (in blue rectangle) and a Bayesian update function $h(\cdot)$ (in peach rectangle). In training PDMs , dashed arrows (between conditional decoder and $\{\mathrm{x}_t\}_{t=1}^T$) are non-existent as $\{\mathrm{x}_t\}_{t=1}^T$ refers to observations. The dashed arrows become solid for sample generation, representing the decoder generates x_t in sample generation.

Table 1: A comparative assessment of SepDiff and various sample based generative models focuses on high-quality generation and key representation learning capabilities, including low-dimensional (capturing compact and meaningful latent representations), smooth (ensuring small input variations lead to gradual output transitions), continuous (maintaining consistency in the latent space to prevent abrupt changes), and time-specific (preserving temporal correlations among data features). To systematically evaluate these properties, we design specific experiments: sample quality task and unconditional generation task can illustrate high-quality generation, latent classification task and attributes encoding task evaluate low-dimensional representation learning, latent space interpolation task measures smooth transitions, disentanglement task examines the continuity of the latent space, and time-varying generation task investigates time-specific semantics.

Modelling Space	Methods	Generation	Representation			
		High Quality	Low Dimensional	Smooth	Conti-nuous	Time-Specific
Sample	**AE** [30] (2014)	✗	✓	✗	✗	✗
	VAE [23] (2014)	✗	✓	✗	✗	✗
	GAN [14] (2014)	✗	✗	✗	✗	✗
	DDPM [20] (2020)	✓	✗	✗	✓	✗
	DDIM [41] (2021)	✓	✗	✓	✓	✗
	LDM [37] (2022)	✓	✓	✗	✓	✗
	DiffAE [35] (2022)	✓	✓	✓	✗	✗
	PDAE [61] (2022)	✓	✓	✓	✗	✗
	InfoDiff [49] (2023)	✓	✓	✓	✓	✗
	DisDiff [57] (2023)	✓	✓	✓	✗	✗
	DiTi [59] (2024)	✓	✓	✓	✗	✗
	HDAE [28] (2024)	✓	✓	✓	✗	✗
	DBAE [22] (2025)	✓	✓	✓	✗	✗
Parameter	**PDM** [15] (2024)	✓	✗	✗	✗	✗
	SepDiff (Ours)	✓	✓	✓	✓	✓

introduces a uniform strategy for modelling continuous and discrete observations; (2) A *self*-encoder encodes step-wise parameters into step-wise semantics to reveal a series of gradually changing latent semantics; (3) A mutual information term promotes latent semantics being disentangled and storing meaningful semantics simultaneously; (4) Sampling and reverse-sampling methods are developed, and generation and reconstruction tasks are completed in the parameter space. (5) We comprehensively evaluate SepDiff across seven representation learning tasks, including two newly designed tasks. Table 1 concludes the advance of SepDiff compared with SOTA generative models.

2 Understanding Parameter Diffusion Model - An Alternative View of Bayesian Flow Networks

Parameter Diffusion Models (PDMs), a.k.a Bayesian Flow Networks (BFNs) [15, 42, 56], serve as deep generative models with the primary objective of learning an output distribution for generating observations. The distribution's parameters are learned by a neural network, which takes the posterior parameters of observations of inputs. Here, we try to understand PDMs from an alternative parameter perspective since these (posterior) parameters play a key role in PDMs . PDMs involves concepts such as input distribution, sender distribution and receiver distribution, to introduce PDMs , making it less accessible to readers unfamiliar with PDMs . Interested readers may refer to Appendix A.1 and [15] for the original illustrations.

Figure 1 shows T steps of training and sample generation in PDMs , similar to diffusion models [20, 41]. To train PDMs , we minimize the divergence between the ground-truth data distribution and the evolving output distributions over T steps. At each step $t \in \{T, \ldots, 1\}$, an intermediate (posterior) parameter $\boldsymbol{\theta}_t$ is first updated using a Bayesian update function $h(\cdot)$ as $\boldsymbol{\theta}_t = h(\boldsymbol{\theta}_{t+1}, \mathbf{x}_{t+1})$, where $\mathbf{x}_{t+1}$ is the observation at step $t+1$. $\boldsymbol{\theta}_t$ is then fed into a neural network $\psi(\cdot)$ to form the parameters of output distribution, i.e., a decoder $p_O(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t)$, for model training. After training, these intermediate output distributions can be employed to simulate observations during the sample generation process, replacing the actual observations at each step t.

By working in the parameter space, PDMs can uniformly model continuous and discrete observations. For example, PDMs can use the mean of Gaussian distributions as parameter $\boldsymbol{\theta}$ to model continuous data or use the event probabilities of categorical distributions as $\boldsymbol{\theta}$ to study discrete data (see detailed settings for distributions in Appendix Table 3). However, PDMs cannot produce meaningful latent semantics capturing high-level concepts in the observations, such as hair colors in portrait images.

3 SepDiff : Parameter Space Representation Learning

Here, we explain the framework of SepDiff and specific design mechanisms.

3.1 The SepDiff Framework

The framework and workflow of SepDiff are in Figure 2. SepDiff leverages the parameter space for representation learning by extracting low-dimensional latent semantics from high-dimensional data. Different from PDMs in approximating data distribution $p(\mathbf{x}_0)$, SepDiff learns the joint distribution over observation $\mathbf{x}_0$ and a series of latent semantics $\{\mathbf{z}_t\}_{t=1}^{T}$, with $|\mathbf{z}_t| \ll |\mathbf{x}_0|, \forall t \in \{1, \ldots, T\}$. That is, SepDiff seeks to reconstruct $\mathbf{x}_0$ while obtaining meaningful low-dimensional latent semantics $\{\mathbf{z}_t\}_{t=1}^{T}$.

Building on PDMs , SepDiff consists of four main components:

(1) *A self-encoder*, conditioning on the intermediate (posterior) parameters $\boldsymbol{\theta}_t$ to generate progressive latent semantics $\mathbf{z}_t$, described in Section 3.2.
(2) *A conditional decoder*, using a neural network on latent semantics $\mathbf{z}_t$ and intermediate parameters $\boldsymbol{\theta}_t$ to form the output distribution for subsequent steps, detailed in Section 3.3.
(3) *A sampling and reverse-sampling process*, facilitating tasks such as image reconstruction and interpolation, outlined in Section 3.4.
(4) *A training and testing procedure*, as discussed in Section 3.5, optimizing latent semantics $\mathbf{z}_t$ and ensuring an effective model generalization.

Together, SepDiff forms a robust framework to capture and utilize latent semantics and to improve the performance of tasks including unconditional image generation and reconstruction.

3.2 Parameter Encoding through A Self-encoder

The *self-encoder*, denoted as $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$, progressively encodes intermediate parameters $\boldsymbol{\theta}_t$ into low-dimensional latent semantics $\mathbf{z}_t$, which facilitates representation learning from high-dimensional data at each step t. [4] has shown that upsampling layers from a U-Net in pretrained diffusion models [37] may capture meaningful semantic information. Inspiring from this discovery and in training SepDiff we adopt approaches similar to [29] to parameterize $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$. Through $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$, the intermediate parameter $\boldsymbol{\theta}_t$ effectively encodes itself into $\mathbf{z}_t$, together they form $\psi(\boldsymbol{\theta}_t, \mathbf{z}_t)$ for the output distribution.

Ideally, the latent semantics $\mathbf{z}_t$ should provide low-dimensional semantics distinct from the intermediate parameters $\boldsymbol{\theta}_t$ in PDMs but without compromising the data reconstruction process. To learn high-quality latent semantics, a smooth, learnable latent space is necessary, which is ensured by integrating the prior distribution $p(\mathbf{z}_t)$ into a robust probabilistic framework, allowing efficient sampling of $\mathbf{x}_0$. For simplicity and efficiency, we assume $p(\mathbf{z}_t)$ follows a Gaussian distribution.

Here, $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$ differs from traditional auto-encoders $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_0)$ in two key aspects:

- $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$ is conditioned on the intermediate parameter $\boldsymbol{\theta}_t$, rather than being conditioned on $\mathbf{x}_0$. This summarizes information from all previous steps to enable generating latent semantic $\mathbf{z}_t$ through all the T steps.
- The self-encoder generates a step-wise semantic $\mathbf{z}_t$, which is tailored to the dynamic behavior of variables over time t. This series of latent semantics $\{\mathbf{z}_t\}_{t=1}^{T}$ are expected to exhibit progressive semantic behaviors (such as gradual changes in age, smile, or skin color) throughout the generation process (as illustrated in the right panel of Figure 5).

When observations $\mathbf{x}_0$ are unavailable, e.g. sample generation tasks, it is also worth noting that directly using regular auto-encoders like $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_0)$ to generate latent semantics is infeasible. They may require an additional module to generate latent semantics [35], while training such modules would introduce computational overhead. However, in their case, not using auto-encoders $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_0)$ would lead to inefficient resource use.

3.3 Conditional Decoder

The conditional decoder refers to the output distribution $p_{\mathrm{O}}(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t, \mathbf{z}_t))$ which conditions on latent semantics $\mathbf{z}_t$ and intermediate parameter $\boldsymbol{\theta}_t$ to simulate $\mathbf{x}_t$. The condition $\psi(\boldsymbol{\theta}_t, \mathbf{z}_t)$ explicitly incorporates $\mathbf{z}_t$ as part of its conditioning mechanism. Following the settings in diffusion models [20, 41], we use the U-Net architecture with the Cross-Attention in each layer specified as:

$$\text{Cross-Attention}(\boldsymbol{\theta}_t, \mathbf{z}_t) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{d}})\mathbf{V}, \tag{1}$$

where $\mathbf{Q} = \mathbf{W}^{Q}\boldsymbol{\theta}_t, \mathbf{K} = \mathbf{W}^{K}\mathbf{z}_t, \mathbf{V} = \mathbf{W}^{V}\mathbf{z}_t$ and $\mathbf{W}^{Q}, \mathbf{W}^{K}, \mathbf{W}^{V}$ are the query, key and value weight matrix, respectively. See the detailed U-Net architecture.

Since $\mathbf{z}_t$ works together with the corresponding intermediate parameter $\boldsymbol{\theta}_t$, it is expected that $\mathbf{z}_t$ aligns well with the progressively structured parameter $\boldsymbol{\theta}_t$. Lower-level intermediate latent $\mathbf{x}_t$ (such as hair texture) is progressively incorporated. The proposed self-encoder works consistently with the conditional decoder here as both work on $\boldsymbol{\theta}_t$, see Figure 6 (b).

3.4 Sampling and Reverse-sampling Processes

After training SepDiff , the sampling and reverse-sampling processes play a crucial role in generating and reconstructing data, which is essential for tasks such as image generation and interpolation. Generating samples begins with an initial guess of the intermediate parameters $\boldsymbol{\theta}_{T+1}$. From $\boldsymbol{\theta}_{T+1}$, this sampling process sequentially generates $\mathbf{x}_T, \mathbf{x}_{T-1}, \ldots, \mathbf{x}_0$. Specifically, given the parameter $\boldsymbol{\theta}_t$ at each step t, we have:

$$\mathbf{z}_t \sim q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t),\ \mathbf{x}_t \sim p_{\mathrm{O}}(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t, \mathbf{z}_t)),\ \boldsymbol{\theta}_{t-1} = h(\boldsymbol{\theta}_t, \mathbf{x}_t). \tag{2}$$

We use the trained encoder $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$ to replace the prior $p(\mathbf{z}_t)$ of $\mathbf{z}_t$ for improving the sampling quality. After $\boldsymbol{\theta}_0$ is obtained, a sample can be generated as $\mathbf{z}_0 \sim q_{\boldsymbol{\phi}}(\mathbf{z}_0|\boldsymbol{\theta}_0, 0), \mathbf{x}_0 \sim p_{\mathrm{O}}(\mathbf{x}_0|\psi(\boldsymbol{\theta}_0, \mathbf{z}_0))$.

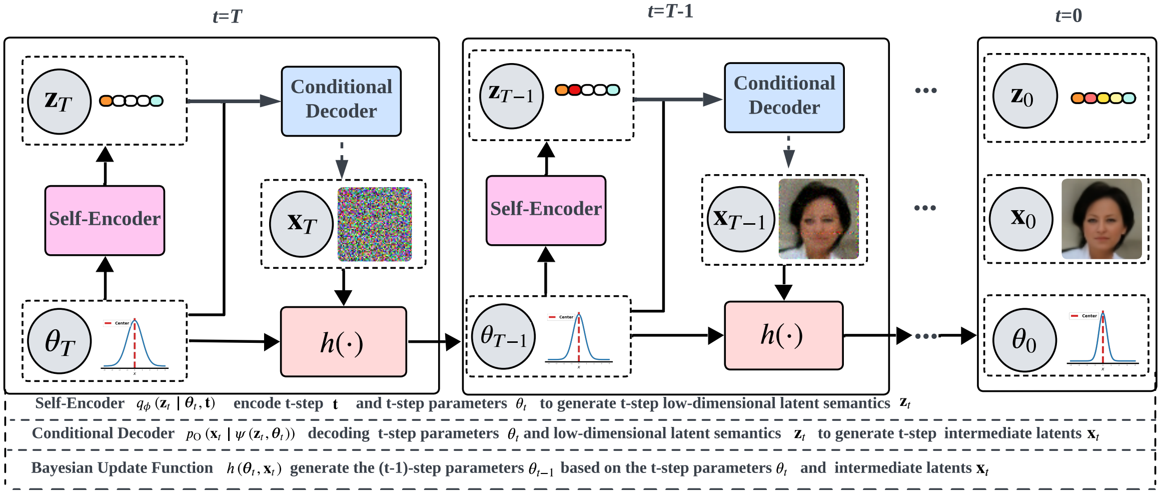

t=T
t=T-1
t=0
$\mathbf{z}_T$
Conditional Decoder
Self-Encoder
$\mathbf{x}_T$
θ_T
$h(\cdot)$
$\mathbf{z}_{T-1}$
$\mathbf{x}_{T-1}$
θ_{T-1}
$\mathbf{z}_0$
$\mathbf{x}_0$
θ_0
Center
Self-Encoder $q_\phi(\mathbf{z}_t \mid \theta_t, \mathbf{t})$ encode t-step t and t-step parameters θ_t to generate t-step low-dimensional latent semantics $\mathbf{z}_t$
Conditional Decoder $p_O(\mathbf{x}_t \mid \psi(\mathbf{z}_t, \theta_t))$ decoding t-step parameters θ_t and low-dimensional latent semantics $\mathbf{z}_t$ to generate t-step intermediate latents $\mathbf{x}_t$
Bayesian Update Function $h(\theta_t, \mathbf{x}_t)$ generate the (t-1)-step parameters θ_{t-1} based on the t-step parameters θ_t and intermediate latents $\mathbf{x}_t$

Figure 2: The framework of SepDiff . Each step consists of a self-encoder $q_\phi(\mathrm{z}_t|\boldsymbol{\theta}_t, t)$ (pink rectangle), a conditional decoder $p_O(\mathrm{x}_t|\psi(\mathrm{z}_t, \boldsymbol{\theta}_t))$ (blue rectangle), and Bayesian update $h(\cdot)$ (peach rectangle). During the reverse-sampling stage, the self-encoder q_ϕ encodes intermediate parameters $\boldsymbol{\theta}_t$ into a time-specific latent semantic z_t, and $p_O(\mathrm{x}_t|\psi(\mathrm{z}_t, \boldsymbol{\theta}_t))$ generates x_t.

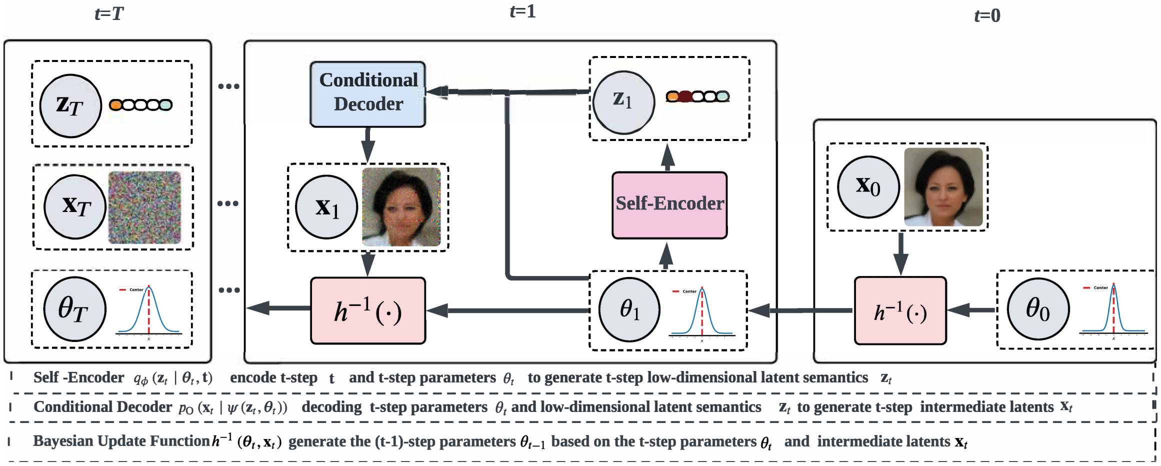

t=T
t=1
t=0
$\mathbf{z}_T$
$\mathbf{x}_T$
θ_T
Conditional Decoder
$\mathbf{z}_1$
Self-Encoder
$\mathbf{x}_1$
$h^{-1}(\cdot)$
θ_1
$\mathbf{x}_0$
$h^{-1}(\cdot)$
θ_0
Center
Self -Encoder $q_\phi(\mathbf{z}_t \mid \theta_t, \mathbf{t})$ encode t-step t and t-step parameters θ_t to generate t-step low-dimensional latent semantics $\mathbf{z}_t$
Conditional Decoder $p_O(\mathbf{x}_t \mid \psi(\mathbf{z}_t, \theta_t))$ decoding t-step parameters θ_t and low-dimensional latent semantics $\mathbf{z}_t$ to generate t-step intermediate latents $\mathbf{x}_t$
Bayesian Update Function $h^{-1}(\boldsymbol{\theta}_t, \mathbf{x}_t)$ generate the (t-1)-step parameters θ_{t-1} based on the t-step parameters $\boldsymbol{\theta}_t$ and intermediate latents $\mathbf{x}_t$

Figure 3: The reverse-sampling process in SepDiff .

However, the reverse-sampling process, which transits the observation $\mathbf{x}_0$ through the intermediate latents $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{T-1}$ until $\mathbf{x}_T$, is not as straightforward as the sampling procedure. Without a clearly defined reverse-sampling process, it would be challenging to perform tasks such as image reconstruction and interpolation. In fact, by taking the inverse of the Bayesian update function $h(\cdot)$ as $\boldsymbol{\theta}_t = h^{-1}(\boldsymbol{\theta}_{t-1}, \mathbf{x}_{t-1})$, the intermediate latent $\mathbf{x}_{t-1}$ can transit to $\mathbf{x}_t$ as:

$$\boldsymbol{\theta}_t = h^{-1}(\boldsymbol{\theta}_{t-1}, \mathbf{x}_{t-1}),\ \mathbf{z}_t \sim q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t),\ \mathbf{x}_t \sim p_O(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t, \mathbf{z}_t)). \tag{3}$$

Given the straightforward definition of Bayesian update function $h(\cdot)$, its inverse operation is generally easy to derive. Furthermore, this developed reverse-sampling process can be naturally extended to PDMs . Transiting $\mathbf{x}_{t-1}$ to $\mathbf{x}_t$ at time t can be performed as $\boldsymbol{\theta}_t = h^{-1}(\boldsymbol{\theta}_{t-1}, \mathbf{x}_{t-1})$, with $\mathbf{x}_t$ sampled as $\mathbf{x}_t \sim p_O(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t))$. With this approach, PDMs can effectively perform downstream tasks like image reconstruction and interpolation, which were difficult or even impossible by previous PDMs . Figure 3 shows the reverse-sampling process of SepDiff . The PDMs version is provided in Figure 7 in Appendix A.

3.5 Training and Test with SepDiff

Here, we outline the process of training and testing SepDiff by focusing on optimizing SepDiff to learn meaningful latent semantics while ensuring effective reconstruction of observations. The training process involves variational inference to approximate the joint distribution of latent variables, and a mutual information term is integrated into improving the quality of learned latent semantics by strengthening the relationship between intermediate parameters and latent semantics.

Variational Inference for Intractable Joint Distribution In SepDiff , the joint distribution over $\mathbf{x}_0$, intermediate latents $\{\mathbf{x}_t\}_{t=1}^{T}$ and latent semantics $\{\mathbf{z}_t\}_{t=1}^{T}$ can be defined as $p(\mathbf{x}_0, \{\mathbf{x}_t\}_{t=1}^{T}, \{\mathbf{z}_t\}_{t=1}^{T}|-) = p_{\mathrm{O}}(\mathbf{x}_0|\psi(\boldsymbol{\theta}_0, \mathbf{z}_0))\cdot\prod_{t=1}^{T}\left[p(\mathbf{z}_t)\mathbb{E}_{p_{\mathrm{O}}(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t,\mathbf{z}_t))}[p_{\mathrm{S}}(\mathbf{x}_{t-1}|\mathbf{x}_t)]\right]$, where the output distribution $p_{\mathrm{O}}(\mathbf{x}_0|\psi(\boldsymbol{\theta}_0, \mathbf{z}_0))$ at step 0 is used to model observation $\mathbf{x}_0$, and $\mathbb{E}_{p_{\mathrm{O}}(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t,\mathbf{z}_t))}[p_{\mathrm{S}}(\mathbf{x}_{t-1}|\mathbf{x}_t)]$ follows the definition of PDMs to model intermediate latent $\mathbf{x}_{t-1}$, and $p_{\mathrm{S}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is a noisy distribution of $\mathbf{x}_t$.

With $q_{\boldsymbol{\phi}}(\mathbf{z}_t \mid \boldsymbol{\theta}_t, t)$ defined as the encoder for $\mathbf{z}_t$ and $p_{\mathrm{S}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ defined as the variational distribution for $\mathbf{x}_{t-1}$, the evidence lower bound (ELBO) on the marginal log-likelihood of observation $\mathbf{x}_0$ is Eq. 4 (see the full derivation in Appendix B.1).

Maximizing ELBO is equivalent to performing amortized inference [23] through encoders $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$ and learning likelihood function through decoders [64]. When the encodable posterior $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$ is used to infer high-level semantics $\mathbf{z}_t$, those intermediate latents $\{\mathbf{x}_t\}_{t=1}^{T}$ contain low-level information in generating the observations. In SepDiff , the parameters of the output distribution are learned through iteratively proceeding the Bayesian updating functions and a learned noise model $\psi(\boldsymbol{\theta}, \mathbf{z})$ parameterized by neural networks ψ.

Mutual Information Regularization Ideally, during the training phase, we want to acquire the latent semantic $\mathbf{z}_t$ by the self-encoder $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$ and achieve high-quality reconstruction $\widehat{x_0}$ by the decoder (i.e., the output distribution $p_{\mathrm{O}}(\mathbf{x}_0|\psi(\boldsymbol{\theta}_0, \mathbf{z}_0))$). However, there exists a trade-off between inference and learning [38, 50] coherent in optimizing the ELBO in Eq. (4). In most cases, optimizing ELBO favours fitting likelihood rather than inference [64]. Based on the rate-distortion theory [1, 3], the rate, represented by the KL divergence term constrained by the encoders, compresses sufficient information to minimize the distortion, or reconstruction error, while simultaneously limiting the informativeness to promote a smooth latent space.

To remedy the insufficient representation learning during the inference stage, we want to increase the dependence between intermediate parameters $\boldsymbol{\theta}_t$ and latent semantics $\mathbf{z}_t$ by maximizing their mutual information $MI(\boldsymbol{\theta}_t, \mathbf{z}_t)$. We can rewrite the tractable learning object in SepDiff by adding the mutual information maximization term as

$$\text{ELBO}_{+} = \text{ELBO} + \frac{\gamma}{T}\sum_{t} MI_q(\boldsymbol{\theta}_t; \mathbf{z}_t) = \mathcal{L}^{\mathrm{D}} - \mathcal{L}^{\mathrm{R}} + \frac{\gamma}{T}\sum_{t} MI_q(\boldsymbol{\theta}_t; \mathbf{z}_t) \tag{5}$$

where γ is the trade-off parameter, $\mathcal{L}_{\mathbf{D}}$ is distoration term and $\mathcal{L}_{\mathbf{R}}$ is the rate term. Considering that we cannot optimize this object directly, we can rewrite it by factorizing the rate term into mutual information and total correlation (TC) to acquaire the final training object:

$$\begin{aligned}
&\mathcal{L}_{\texttt{SepDiff+}} \quad = \\
&-\sum_{t=1}^{T}\mathbb{E}_{p_{\mathrm{F}}(\boldsymbol{\theta}_t|-)}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_t)}\Big\{D_{\mathrm{KL}}[p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_0; \alpha_{T:t})\|p_{\mathrm{R}}(\mathbf{x}_{t-1}; \psi(\boldsymbol{\theta}_t, \mathbf{z}_t), \alpha_t)] \\
&\quad -\frac{1-\gamma}{T}D_{\mathrm{KL}}[q_{\boldsymbol{\phi}}(\mathbf{z}_t \mid \boldsymbol{\theta}_t)\|p(\mathbf{z})] - \frac{\gamma+\lambda-1}{T}D_{\mathrm{KL}}[q_{\boldsymbol{\phi}}(\mathbf{z}_t)\|p(\mathbf{z})]\Big\} \\
&\qquad\qquad\qquad + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_0,\boldsymbol{\theta}_0)}[\ln p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0, \mathbf{z}_0))],
\end{aligned} \tag{6}$$

where λ is the scale parameter of $D_{KL}[q_{\boldsymbol{\phi}}(z_t)\|p(z_t)]$. The full derivation is in Appendix B.

4 Experiments

We present two variants of SepDiff operating in different parameter spaces: SepDiff*d* with discrete input distributions for discrete datasets, and SepDiff*c* with continuous input distributions for continuous datasets, respectively. In addition, we evaluate the representation and generative capabilities of SepDiff in seven tasks, including **latent classification task** , **latent space interpolation task** , **disentanglement task** , **attributes encoding task** and **sample quality task** . Furthermore, we propose a novel **time-varying generation task** and demonstrate that SepDiff can perform **unconditional generation task** directly, where samples are generated *solely by the decoder* using a given prior. Specifically, to comprehensively evaluate the generative and representation learning capabilities of SepDiff , we devised seven tasks to address the following research questions:

- **RQ1:** What performance improvements does SepDiff achieve over state-of-the-art (SOTA) generative representation learning frameworks?
- **RQ2:** What novel property does SepDiff introduce to generative representation learning models, and how can it be empirically validated?
- **RQ3:** How does SepDiff enhance generative models by introducing new features?
- **RQ4:** How can we verify that the low-dimensional features learned by SepDiff capture meaningful semantic information?
- **RQ5:** How can we assess the probabilistic properties of the low-dimensional features learned by SepDiff, particularly their smoothness and continuity, as reflected in latents $\mathbf{z}_t \sim q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$?
- **RQ6:** How does SepDiff compare to existing generative representation learning frameworks in terms of time efficiency during training and inference?
- **RQ7:** What is the contribution of each component in SepDiff to its overall performance, as demonstrated through ablation studies?

4.1 Experimental Setup

The Choices of Baselines and Datasets We conduct a two-fold comparison to evaluate the performance of SepDiff variants. Firstly, we compare our parameter-based models (SepDiff*c* and SepDiff*d*) with established sample-based representation learning baselines, including AE and VAE-based models such as β-VAE [19], infoVAE [64], and diffusion-based models such as DiffAE [35] and InfoDiff [49]. These models represent key advancements in the field: β-VAE introduce disentanglement into VAE, infoVAE incorporates MMD for balancing generation and representation, while DiffAE and InfoDiff explore the integration of AEs and VAEs into diffusion models to learn encodable latents and disentangled representations, respectively. Secondly, we compare the performance of SepDiff*c* and SepDiff*d* across various input distributions for continuous and discrete data, respectively. The discrete datasets include binarized versions of MNIST (*b*MNIST) [11], FashionMNIST (*b*FashionMNIST

$$\log p(\mathbf{x}_0) \geq -\sum_{t=1}^{T} \mathbb{E}_{p_{\mathrm{F}}(\boldsymbol{\theta}_t|-)} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t,t)} \Big\{ D_{\mathrm{KL}}[p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_0) \| \mathbb{E}_{p_{\mathrm{O}}(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t,\mathbf{z}_t))}[p_{\mathrm{S}}(\mathbf{x}_{t-1}|\mathbf{x}_t)]] \\ - D_{\mathrm{KL}}[q_{\boldsymbol{\phi}}(\mathbf{z}_t \mid \boldsymbol{\theta}_t, t) \| p(\mathbf{z}_t)] \Big\} + \mathbb{E}_{p_F(\boldsymbol{\theta}_0|-) q_{\boldsymbol{\phi}}(\mathbf{z}_0|\boldsymbol{\theta}_0,0)} [\ln p_{\mathrm{O}}(\mathbf{x}_0|\psi(\boldsymbol{\theta}_0, \mathbf{z}_0))] := \textsc{elbo}. \quad (4)$$

) [54], while the continuous datasets include CelebA [27], CIFAR-10 [25], 3DShapes [6][1] and FFHQ-64. This comparison allows for a detailed examination of how different parameter space assumptions impact the representation learning of discrete and continuous data.

Metrics, Deep Structures and Hyperparameters. To ensure reproducibility and comprehensive understanding of the experimental setup. We detailed the metrics for evaluation in Appendix B.3.

4.2 RQ1: SOTA Performance

Downstream Classification for Representation Learning To evaluate the representation capability of our SepDiff, we design downstream classifier-based **latent classification task** and report the AUROC to measure the quality of the learned latent $\mathbf{z}_0$. From Figure 4 (a) for the discrete datasets and Table 2 the continuous datasets, we can conclude that both SepDiff*c* and SepDiff*d* can achieve a higher AUROC, suggesting that the learned latent $\mathbf{z}_0$ contain more low-dimensional semantics about the data, which is *general* and *transferable* [13]. More about AUROC and experimental details can be seen in the Appendix B.3. In Figure 4 (a), we can see that the SepDiff*d* with discrete assumption achieves the best performance in two datastes. Additionally, we report SepDiff*c* results on continuous datasets in Table 2. We can see that for the continuous data, the SepDiff*c*s with Delta distribution can achieve the highest AUROC, capturing the most informative semantics for classification in three datasets.

Generation Ability In addition to evaluating the classification based representation ability, we also conduct **sample quality task** against baselines. For discrete data, we report the FID in Figure 4 (a). For continuous data, we report the FID in Table 2. We can conclude that the SepDiff*c* with Delta distribution can achieve the lowest FID value for three datasets. The description and configuration details for the FID metric used are provided in the Appendix B.3.

4.3 RQ2: New Time-Varying Task

We extend the representation learning scope on existing frameworks for **attributes encoding task** and propose a *new* **time-varying generation task** , to evaluate the effectiveness of the progressive latent semantics learned by the self-encoder.

Attributes Encoding in Representation Learning Figure 5 (a) demonstrates that attributes are captured by the learned latent semantics $\{\mathbf{z}_t\}_{t=1}^T$ in **attributes encoding task** . This is illustrated by a set of latent-sample pairs $< \{\mathbf{z}_t^i\}_{t=1}^T, \mathbf{x}_T^{i,j} >$, where $\{\mathbf{z}_t^i\}_{t=1}^T$ are obtained by reverse-sampling from the i-th input image through the trained SepDiff , and $\mathbf{x}_T^{i,j}$ is the j-th sample from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ corresponding to the i-th input image. Concurrently, the inherent attributes of samples, such as local attributes in images (e.g., `Narrow_Eyes`, `Mouth_Slightly_Open`, `Blond_Hair`), are characterized by $\mathbf{x}_T^{i,j}$.

New Time-Varying Task We illustrate the learned time-varying semantics by **time-varying generation task** on Figure 4 (b), and Figure 5 (b). Specifically, a latent sample pair $< \{\mathbf{z}_t^{\text{fixed}}\}_{t=1}^T, \mathbf{x}_T^{\text{fixed}} >$ is first obtained by applying the reverse sampling process in trained SepDiff on an image. Then, we use the latent semantics at step t^* to replace other steps' ones and "reconstruct" the image as $\mathbf{x}_t \sim p_{\mathrm{O}}(\mathbf{x}_t|\psi(\boldsymbol{\theta}_t, \mathbf{z}_{t^*}^{\text{fixed}})), \boldsymbol{\theta}_{t-1} = h(\boldsymbol{\theta}_t, \mathbf{x}_t), \forall t = T, \ldots, 1$. In that case, the attributes vary due to the semantics evolution encoded by time-specific latent.

4.4 RQ3: New Paradigm for Unconditional Generation

We introduce new paradigm for **unconditional generation task** without relying on training additional deep modules. Refer to Algorithm 1 in Appendix for more information. We can conclude that VAE-based models still produce blurry reconstructions, while diffusion-based and parameter-based models can build near-exact reconstructions.

4.5 RQ4: Time-dependent semantics guided interpolation

latent space interpolation task [14, 19] is commonly used to validate the smoothness, continuity, and semantic coherence of the learned latent semantics in generative models. Typically, two samples are embedded into the latent space, and interpolating between the latent variables generates interpolated representations. The reconstructed outputs produced by the sampling process reveal the semantic richness of the latent space. Demonstration of the image interpolation is detailed in Appendix B.4.

SepDiff achieves near-exact reconstruction, in contrast to the downgraded performance of VAE variants such as (a) vanilla VAE, and (b) β-VAE. Compared with diffusion models (c) DiffAE and (d) InfoDiff, SepDiff characterizes a smoother and more consistent latent space.

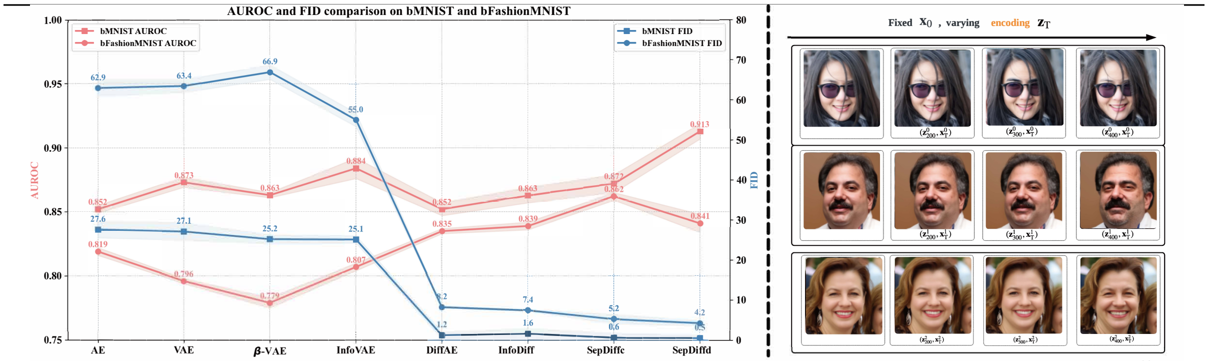
4.6 RQ5: Time-dependent semantic encoding for disentanglement

We perform latent traversals on the FFHQ-64 and CelebA datasets to evaluate the **disentanglement task** of our trained SepDiff . In this process, we modify one dimension of the learned latent semantics $\{\mathbf{z}_t\}_{t=1}^T$ each step, and replace it with M evenly distributed numbers within a standardized range (e.g., −3 to +3), while keeping the other dimensions fixed. After decoding these adjusted latent semantics, we evaluate the generated samples for changes in specific attributes. Successful disentanglement is verified when manipulating one single dimension alters only one distinguishable attribute, such as age, while leaving all other attributes unchanged. SepDiff effectively isolates and controls individual data attributes in both

[1] For the discrete version, continuous data (k-bit images) can be discretized into 2^k bins by dividing the data range $[-1, 1]$ into k intervals, each of length $2/k$.

Table 2: Comparison of representation learning algorithms on continuous data by disentanglement performance (mean ± std) and classification. The quantitative results for each algorithm are averaged over five trials. Notations: Modeling on data space $\mathcal{D}$, parameter space $\mathcal{P}$. Prior distributions: Gaussian g, Categorical c, Delta d. ↑: higher better, ↓: lower better. Color: Top-1, Top-2.

Prior on	Prior type	Methods	CelebA				3DShapes		CIFAR-10	
			$\mathcal{TAD}\uparrow$	$\mathcal{ATTRS}\uparrow$	$\mathcal{FID}\downarrow$	$\mathcal{AUROC}\uparrow$	$\mathcal{DCI}\uparrow$	$\mathcal{AUROC}\uparrow$	$\mathcal{FID}\downarrow$	$\mathcal{AUROC}\uparrow$
$\mathcal{D}$	-	**AE**	0.042 ±0.004	1.0 ±0.0	90.4±1.8	0.759 ±0.003	0.219 ±0.001	0.796±0.007	169.4±2.4	0.721±0.001
	g	**VAE** [23]	0.000 ±0.000	0.0 ±0.0	94.3±2.8	0.770 ±0.002	0.276 ±0.001	0.799±0.002	177.2±3.2	0.743±0.002
	g	**β-VAE** [7]	0.088 ±0.051	1.6 ±0.8	99.8±2.4	0.699 ±0.001	0.281 ±0.001	0.801±0.001	183.3±3.1	0.769±0.003
	g	**InfoVAE** [64]	0.000 ±0.000	0.0 ±0.0	77.8±1.6	0.757 ±0.003	0.134 ±0.001	0.829±0.003	160.7±2.5	0.814±0.006
	g	**DiffAE** [35]	0.155 ±0.010	2.0 ±0.0	22.7±2.1	0.799 ±0.002	0.196 ±0.001	0.899±0.001	32.1±1.1	0.859±0.002
	g	**InfoDiff** [49]	0.299 ±0.006	3.0 ±0.0	23.8±1.6	0.848 ±0.001	0.342 ±0.002	0.882±0.001	32.4±1.8	0.886±0.004
$\mathcal{P}$	c	**SepDiff** ($\gamma = 1, \lambda = 0.01$)	0.261 ±0.01	**5.0 ±0.0**	22.6±1.2	0.846 ±0.009	0.477 ±0.002	0.901±0.007	31.8±1.1	0.892±0.004
	d	**SepDiff** ($\gamma = 0.9, \lambda = 0.01$)	0.302 ±0.005	4.0 ±0.0	22.1±1.6	0.850 ±0.006	**0.567 ±0.005**	0.902±0.001	31.2±1.1	0.901±0.001
	d	**SepDiff** ($\gamma = 1, \lambda = 0.01$)	**0.368 ±0.005**	3.0 ±0.0	**21.6±1.1**	**0.865±0.004**	0.485 ±0.009	**0.931±0.001**	**31.1±1.1**	**0.911±0.002**



(a): Comparison on discrete data by classification accuracy and generation performance.

(b): Time-varying representation learning of SepDiff

Figure 4: Quantitative comparison over generative representation learning models on discrete data (a). SepDiff demonstrates competitive performance in capturing latent for classification, achieving approximately 0.84 AUROC for bFashionMNIST and 0.91 for bMNIST . Additionally, it shows robust generative capabilities, with FID values ranging from 0.5 to 0.6 for bMNIST and around 5 for bFashionMNIST . Among SepDiffs, SepDiff$d$ with a categorical distribution is particularly effective in modelling discrete data distributions, yielding lower FID values of 0.5 for bMNIST and 4.2 for bFashionMNIST . As shown in (b), the learned semantics exhibit progressive, time-varying changes. By varying time encodes at 200, 300, 400 time steps, more attributes will be influenced in the reconstruction stage: the `Wavy_hair`, `Brown_hair`, `Arched_Eyebrows` attributes in the first line, the `Double_Chin`, `Mustache`, `Goatee` attributes in the second line and the `Young`, `High_Cheekbones`, `Arched_Eyebrows` attributes in the third line. Notations: [AUROC, FID]; [(●, bMNIST), (■, bFashionMNIST)]; [(–, SepDiff$d$),(– · –, SepDiff$c$)].

FFHQ-64 and CelebA. For example, on FFHQ-64, manipulating latent dimensions controls attributes like `Mustache`, `Brown Hair`, and `Eyeglasses`, while other attributes remain constant. Similarly, on CelebA, attributes such as `Smiling`, `Pale Skin`, and `Big Nose` are independently manipulated without affecting others.

4.7 RQ6: Time Efficiency

We report the time efficiency in the training and inference phases.

Training Efficiency When compared to BFN, SepDiff incurs an additional time complexity due to the new self-encoder module with a lightweighted U-Net network. Assuming this light-weighted U-Net network has four layers, with the neuron counts being $h1, h2, h3, h4$, the additional time complexity would be $O(h1 + h2 + h3 + h4)$. These three methods share the same U-Net architecture. The training is conducted on two NVIDIA H100 GPUs, each with 80 GB of memory, to ensure sufficient computational resources for handling large-scale datasets. From this table, we can see SepDiff requires around 25% more training time than DDIM, largely due to the additional semantic encoder. However, the training speed of SepDiff is comparable to that of DiffAE, owing to the lightweighted modules integrated within the encoder.

Inference Efficiency The SepDiff also needs the steps for a generation like the diffusion model. We evaluate the impact of steps (n = 100, 500, 1000) on sampling speed with the experiments conducted on one H100 GPU with 80GB of memory. We illustrate some results, each value represents the number of 64x64 images generated per second at the current time step.

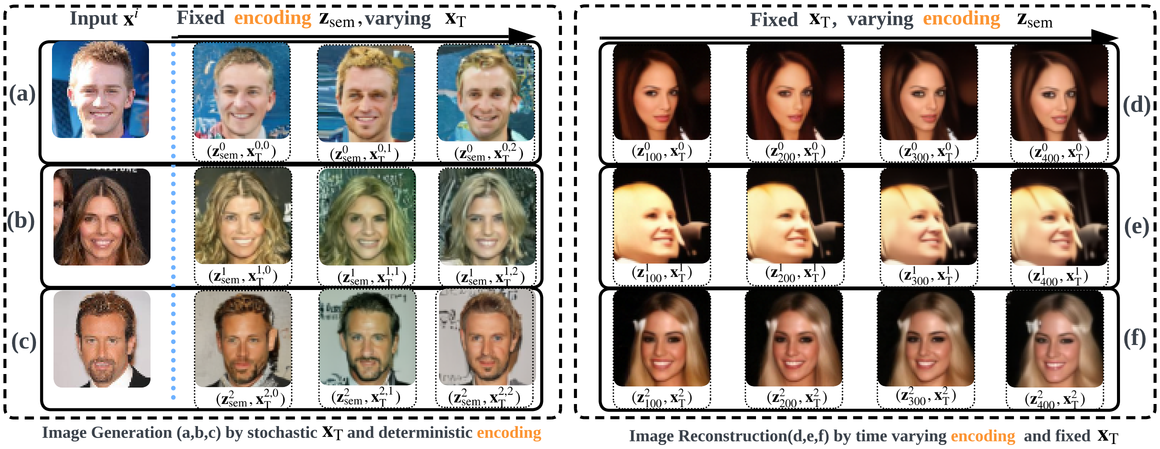

Input $\mathbf{x}^i$
Fixed encoding $\mathbf{z}_{\text{sem}}$, varying $\mathbf{x}_{\text{T}}$
(a)
$(\mathbf{z}^0_{\text{sem}}, \mathbf{x}^{0,0}_{\text{T}})$
$(\mathbf{z}^0_{\text{sem}}, \mathbf{x}^{0,1}_{\text{T}})$
$(\mathbf{z}^0_{\text{sem}}, \mathbf{x}^{0,2}_{\text{T}})$
(b)
$(\mathbf{z}^1_{\text{sem}}, \mathbf{x}^{1,0}_{\text{T}})$
$(\mathbf{z}^1_{\text{sem}}, \mathbf{x}^{1,1}_{\text{T}})$
$(\mathbf{z}^1_{\text{sem}}, \mathbf{x}^{1,2}_{\text{T}})$
(c)
$(\mathbf{z}^2_{\text{sem}}, \mathbf{x}^{2,0}_{\text{T}})$
$(\mathbf{z}^2_{\text{sem}}, \mathbf{x}^{2,1}_{\text{T}})$
$(\mathbf{z}^2_{\text{sem}}, \mathbf{x}^{2,2}_{\text{T}})$
Image Generation (a,b,c) by stochastic $\mathbf{x}_{\text{T}}$ and deterministic encoding
Fixed $\mathbf{x}_{\text{T}}$, varying encoding $\mathbf{z}_{\text{sem}}$
(d)
$(\mathbf{z}^0_{100}, \mathbf{x}^0_{\text{T}})$
$(\mathbf{z}^0_{200}, \mathbf{x}^0_{\text{T}})$
$(\mathbf{z}^0_{300}, \mathbf{x}^0_{\text{T}})$
$(\mathbf{z}^0_{400}, \mathbf{x}^0_{\text{T}})$
(e)
$(\mathbf{z}^1_{100}, \mathbf{x}^1_{\text{T}})$
$(\mathbf{z}^1_{200}, \mathbf{x}^1_{\text{T}})$
$(\mathbf{z}^1_{300}, \mathbf{x}^1_{\text{T}})$
$(\mathbf{z}^1_{400}, \mathbf{x}^1_{\text{T}})$
(f)
$(\mathbf{z}^2_{100}, \mathbf{x}^2_{\text{T}})$
$(\mathbf{z}^2_{200}, \mathbf{x}^2_{\text{T}})$
$(\mathbf{z}^2_{300}, \mathbf{x}^2_{\text{T}})$
$(\mathbf{z}^2_{400}, \mathbf{x}^2_{\text{T}})$
Image Reconstruction(d,e,f) by time varying encoding and fixed $\mathbf{x}_{\text{T}}$

Figure 5: The left panel (a-b) shows high-level latent semantic captured by z_{sem} from SepDiff 's encoders. By fixing z_{sem}, the global characters of the images are invariant. By varying the stochastic x_T, the local attributes in the corresponding generated images may vary, such as the `Narrow_Eyes` attribute in (a), the `Blond_Hair` attribute in (b), and the `Mouth_Slightly_Open` attribute in (c). The right panel (d-f) illustrates the time-varying changes that SepDiff 's progressive encodes interfaced. By varying time encodes at 100, 200, 300 time steps, more attributes will be influenced in the reconstruction stage: the `Big_Lips, Pointy_Nose` attributes in (d), the `Blond_Hair, Bald` attributes in (e) and the `Wavy_Hair, High_Cheekbones` attributes in (f).

4.8 RQ7: Ablation Studies

The coefficients γ, λ in Eq. 6 will regulate the information flow from $\boldsymbol{\theta}$ to $\mathbf{z}$ by the variational bottleneck rule [7, 38, 50], resulting in the tradeoff between generation and representation learning.

5 Related Works on Generative Representation Learning Models

In this section, we categorize sample-based generative representation learning models into three distinct groups and compare with our SepDiff , a variant of PDM , to highlight promising advantages.

Diffusion Models Recent advances have demonstrated that diffusion models [20, 41] are capable of generating high-quality data. Nonetheless, compared to the autoencoder framework, the intermediate outputs in diffusion stages are high-dimensional and lack smoothness, making them unsuitable for representation learning. Contemporary research focuses on encoding a conditional latent space to acquire low-dimensional semantic representations. However, those sample based models [35, 49], such as VAEs and diffusion models, exhibit limitations when applied to discrete data.

Deep Hierarchical VAEs Deep hierarchical VAEs have seen progress in capturing latent dependence structures for encoding an expressive posterior, statistically or semantically. VQVAE-based [36, 48] models have local-to-global features-based explanatory hierarchies at the image level, forming a codebook-based discrete posterior. In [40, 44], recursive latent structures in multi-layer networks form an aggregated posterior. NVAE [47] demonstrates that depth-wise hierarchies encoded by residual networks can approximate the posterior precisely despite using shallow networks. Unlike the observation-based encoder, where the information flow between input and latent is maximized in encoding-decoding pipelines in the sample space, SepDiff uses progressive encoders in the parameter space to capture the dynamic semantics.

Pretrained Diffusion Models Pretrained diffusion models [37], [4] have shown that the upsampling features from a U-Net can capture semantic information useful for downstream tasks. This discovery has sparked increasing research in leveraging these upsampling features of pretrained diffusion models across various applications, including classification [32, 53], semantic segmentation [4, 65], panoptic segmentation [55], semantic correspondence [17, 29, 43, 60], and image editing [18, 46]. In most of these approaches, identifying the optimal denoising step and upsampling layer is crucial for achieving high predictive performance. These approaches do not suggest fundamental changes to model architectures or training methodologies, leaving the specific architectural components and techniques for learning useful semantic representations unclear. SepDiff uses these discoveries to construct efficient self-encoders.

6 Conclusion and Future Directions

In this work, we introduce SepDiff , a novel unified parameter-space representation learning framework designed to handle both continuous and discrete data. Unlike traditional encoder-based methods that map observations into static latent semantics, SepDiff employs a self-encoder to iteratively derive structured latent semantics from intermediate parameters at each step of the generation process. This approach enables more effective representation learning across diverse data types. We developed new sampling and reverse-sampling methods for SepDiff to support downstream generation and reconstruction tasks in the parameter space. We validate SepDiff through experiments spanning seven representation and generation tasks across two variants. The results demonstrate its superior ability to extract low-dimensional, smooth, and time-varying semantics, leading to unified representations and a clearer semantic understanding of the underlying data.

References

[1] Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A Saurous, and Kevin Murphy. 2018. Fixing a Broken ELBO. *ICML* (2018).

[2] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. 2021. Structured Denoising Diffusion Models in Discrete State-Spaces. *NeurIPS* (2021).

[3] Juhan Bae, Michael R Zhang, Michael Ruan, Eric Wang, So Hasegawa, Jimmy Ba, and Roger Grosse. 2023. Multi-Rate VAE: Train Once, Get the Full Rate-Distortion Curve. *ICLR* (2023).

[4] Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. 2021. Label-Efficient Semantic Segmentation with Diffusion Models. *arXiv preprint arXiv:2112.03126* (2021).

[5] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation Learning: A Review and New Perspectives. *TPAMI* 35, 8 (2013), 1798–1828.

[6] Chris Burgess and Hyunjik Kim. 2018. 3D Shapes Dataset. https://github.com/deepmind/3dshapes-dataset/.

[7] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. 2017. Understanding Disentangling in β-VAE. *NeurIPS* (2017).

[8] Marc-André Carbonneau, Julian Zaidi, Jonathan Boilard, and Ghyslain Gagnon. 2022. Measuring Disentanglement: A Review of Metrics. *TNNLS* (2022).

[9] Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. 2018. Isolating Sources of Disentanglement in Variational Autoencoders. *NeurIPS* (2018).

[10] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. 2023. Analog Bits: Generating Discrete Data using Diffusion Models with Self-Conditioning. *ICLR* (2023).

[11] Li Deng. 2012. The MINST Database of Handwritten Digit Images for Machine Learning Research. *IEEE Signal Processing Magazine* 29, 6 (2012), 141–142.

[12] Sixun Dong, Huazhang Hu, Dongze Lian, Weixin Luo, Yicheng Qian, and Shenghua Gao. 2023. Weakly Supervised Video Representation Learning with Unaligned Text for Sequential Videos. *CVPR* (2023).

[13] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. 2019. Unsupervised Scalable Representation Learning for Multivariate Time Series. *NeurIPS* (2019).

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. *NeurIPS* (2014).

[15] Alex Graves, Rupesh Kumar Srivastava, Timothy Atkinson, and Faustino Gomez. 2023. Bayesian Flow Networks. *arXiv preprint arXiv:2308.07037* (2023).

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. *CVPR* (2016).

[17] Eric Hedlin, Gopal Sharma, Shweta Mahajan, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 2024. Unsupervised Semantic Correspondence Using Stable Diffusion. *NeurIPS* (2024).

[18] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. 2022. Prompt-to-Prompt Image Editing with Cross Attention Control. *arXiv preprint arXiv:2208.01626* (2022).

[19] Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. *ICLR* (2017).

[20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. *NeurIPS* (2020).

[21] Hyunjik Kim and Andriy Mnih. 2018. Disentangling by Factorising. *ICML* (2018).

[22] Yeongmin Kim, Kwanghyeon Lee, Minsang Park, Byeonghu Na, and Il-Chul Moon. 2025. Diffusion Bridge AutoEncoders for Unsupervised Representation Learning. *ICLR* (2025).

[23] Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. *ICLR* (2014).

[24] Rahul Krishnan, Dawen Liang, and Matthew Hoffman. 2018. On the Challenges of Learning with Inference Networks on Sparse, High-dimensional Data. *AISTATS* (2018).

[25] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning Multiple Layers of Features from Tiny Images*. Technical Report. University of Toronto.

[26] Tianhong Li, Huiwen Chang, Shlok Kumar Mishra, Han Zhang, Dina Katabi, and Dilip Krishnan. 2023. MAGE: MAsked Generative Encoder to Unify Representation Learning and Image Synthesis. *CVPR* (2023).

[27] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. *ICCV* (2015).

[28] Zeyu Lu, Chengyue Wu, Xinyuan Chen, Yaohui Wang, Lei Bai, Yu Qiao, and Xihui Liu. 2024. Hierarchical Diffusion Autoencoders and Disentangled Image Manipulation. *WACV* (2024).

[29] Grace Luo, Lisa Dunlap, Dong Huk Park, Aleksander Holynski, and Trevor Darrell. 2024. Diffusion Hyperfeatures: Searching through Time and Space for Semantic Correspondence. *NeurIPS* (2024).

[30] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial Autoencoders. *arXiv preprint arXiv:1511.05644* (2015).

[31] Cristian Meo, Louis Mahon, Anirudh Goyal, and Justin Dauwels. 2024. αTC-VAE: On the Relationship between Disentanglement and Diversity. *ICLR* (2024).

[32] Soumik Mukhopadhyay, Matthew Gwilliam, Yosuke Yamaguchi, Vatsal Agarwal, Namitha Padmanabhan, Archana Swaminathan, Tianyi Zhou, and Abhinav Shrivastava. 2023. Do Text-free Diffusion Models Learn Discriminative Visual Representations? *arXiv preprint arXiv:2311.17921* (2023).

[33] Khalid Oublal, Said Ladjal, David Benhaiem, Emmanuel LE BORGNE, and François Roueff. 2024. Disentangling Time Series Representations via Contrastive Independence-of-Support on l-Variational Inference. *ICLR* (2024).

[34] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. 2022. On Aliased Resizing and Surprising Subtleties in GAN Evaluation. *CVPR* (2022).

[35] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. 2022. Diffusion Autoencoders: Toward a Meaningful and Decodable Representation. *CVPR* (2022).

[36] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. 2019. Generating Diverse High-Fidelity Images with VQ-VAE-2. *NeurIPS* (2019).

[37] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. *CVPR* (2022).

[38] Huajie Shao, Shuochao Yao, Dachun Sun, Aston Zhang, Shengzhong Liu, Dongxin Liu, Jun Wang, and Tarek Abdelzaher. 2020. ControlVAE: Controllable Variational Autoencoder. *ICML* (2020).

[39] Ken Shoemake. 1985. Animating Rotation with Quaternion Curves. *SIGGRAPH* (1985).

[40] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. 2016. Ladder Variational Autoencoders. *NeurIPS* (2016).

[41] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. *ICLR* (2021).

[42] Yuxuan Song, Jingjing Gong, Hao Zhou, Mingyue Zheng, Jingjing Liu, and Wei-Ying Ma. 2024. Unified Generative Modeling of 3D Molecules with Bayesian Flow Networks. *ICLR* (2024).

[43] Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. 2023. Emergent Correspondence from Image Diffusion. *NeurIPS* (2023).

[44] Jakub Tomczak and Max Welling. 2018. VAE with a VampPrior. *AISTATS* (2018).

[45] Sana Tonekaboni, Chun-Liang Li, Sercan O Arik, Anna Goldenberg, and Tomas Pfister. 2022. Decoupling Local and Global Representations of Time Series. *AISTATS* (2022).

[46] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. 2023. Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation. *CVPR* (2023).

[47] Arash Vahdat and Jan Kautz. 2020. NVAE: A Deep Hierarchical Variational Autoencoder. *NeurIPS* (2020).

[48] Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural Discrete Representation Learning. *NeurIPS* (2017).

[49] Yingheng Wang, Yair Schiff, Aaron Gokaslan, Weishen Pan, Fei Wang, Christopher De Sa, and Volodymyr Kuleshov. 2023. InfoDiffusion: Representation Learning Using Information Maximizing Diffusion Models. *ICML* (2023).

[50] Zhangkai Wu, Longbing Cao, and Lei Qi. 2024. eVAE: Evolutionary Variational Autoencoder. *TNNLS* (2024).

[51] Zhangkai Wu, Longbing Cao, Qi Zhang, Junxian Zhou, and Hui Chen. 2024. Weakly Augmented Variational Autoencoder in Time Series Anomaly Detection. *arXiv preprint arXiv:2401.03341* (2024).

[52] Zhangkai Wu, Xuhui Fan, and Longbing Cao. 2025. ProgDiffusion: Progressively Self-encoding Diffusion Models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*. 1633–1644.

[53] Weilai Xiang, Hongyu Yang, Di Huang, and Yunhong Wang. 2023. Denoising Diffusion Autoencoders are Unified Self-supervised Learners. *ICCV* (2023).

[54] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MINST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747* (2017).

[55] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. 2023. Open-Vocabulary Panoptic Segmentation with Text-to-Image Diffusion Models. *CVPR* (2023).

[56] Kaiwen Xue, Yuhao Zhou, Shen Nie, Xu Min, Xiaolu Zhang, Jun Zhou, and Chongxuan Li. 2024. Unifying Bayesian Flow Networks and Diffusion Models through Stochastic Differential Equations. *ICML* (2024).

[57] Tao Yang, Yuwang Wang, Yan Lv, and Nanning Zheng. 2023. Disdiff: Unsupervised Disentanglement of Diffusion Probabilistic Models. *NeurIPS* (2023).

[58] Eric Yeats, Frank Liu, David Womble, and Hai Li. 2022. NashAE: Disentangling Representations through Adversarial Covariance Minimization. *ECCV* (2022).

[59] Zhongqi Yue, Jiankun Wang, Qianru Sun, Lei Ji, Eric I-Chao Chang, and Hanwang Zhang. 2024. Exploring Diffusion Time-Steps for Unsupervised Representation Learning. *ICLR* (2024).

[60] Junyi Zhang, Charles Herrmann, Junhwa Hur, Luisa Polania Cabrera, Varun Jampani, Deqing Sun, and Ming-Hsuan Yang. 2024. A Tale of Two Features: Stable Diffusion Complements DINO for Zero-Shot Semantic Correspondence. *NeurIPS* (2024).

[61] Zijian Zhang, Zhou Zhao, and Zhijie Lin. 2022. Unsupervised Representation Learning from Pre-trained Diffusion Probabilistic Models. *NeurIPS* (2022).

[62] He Zhao, Piyush Rai, Lan Du, Wray Buntine, Dinh Phung, and Mingyuan Zhou. 2020. Variational Autoencoders for Sparse and Overdispersed Discrete Data. *AISTATS* (2020).

[63] Haojie Zhao, Dong Wang, and Huchuan Lu. 2023. Representation Learning for Visual Object Tracking by Masked Appearance Transfer. *CVPR* (2023).

[64] Shengjia Zhao, Jiaming Song, and Stefano Ermon. 2019. InfoVAE: Balancing Learning and Inference in Variational Autoencoders. *AAAI* (2019).

[65] Wenliang Zhao, Yongming Rao, Zuyan Liu, Benlin Liu, Jie Zhou, and Jiwen Lu. 2023. Unleashing Text-to-Image Diffusion Models for Visual Perception. *ICCV* (2023).

[66] Mingyuan Zhou, Tianqi Chen, Zhendong Wang, and Huangjie Zheng. 2023. Beta Diffusion. *NeurIPS* (2023).

A Preliminaries

A.1 Parameter Diffusion Models

In [15], PDMs assume two types of distributions: a simple *input distribution* $P_{\mathrm{I}}(\cdot)$ representing the initial belief about observations and an *output distribution* $P_{\mathrm{O}}(\cdot)$ simulating the observation distribution. The parameters of input distribution are first updated through a Bayesian inference scheme and then passed into a neural network $\psi(\cdot)$ to form the parameters of output distributions. The main objective of PDMs is to minimize the divergence between the ground-truth data distribution and the output distribution, ensuring that the output distribution closely approximates the ground-truth data distribution.

Following the notations in diffusion models, we denote $\mathbf{x}_0$ as the observations. There are T *reverse* steps in PDMs which gradually reveals the information of $\mathbf{x}_0$ through $\{\mathbf{x}_T, \mathbf{x}_{T-1}, \dots, \mathbf{x}_1\}$ to the input distribution[2]. At each step t, $\mathbf{x}_t$ is first noised through a *sender distribution* $p_{\mathrm{S}}(\widehat{x}_t \mid \mathbf{x}_t; \alpha_t)$, with α_t denoting the precision. Combined with input distribution $p_{\mathrm{I}}(\mathbf{x}_t; \boldsymbol{\theta}_{t+1})$, the posterior distribution of $\mathbf{x}_t$ is obtained as $p(\mathbf{x}_t; h(\boldsymbol{\theta}_{t+1}, \widehat{x}_t, \alpha_t)) \propto p_{\mathrm{I}}(\mathbf{x}_t; \boldsymbol{\theta}_{t+1}) p_{\mathrm{S}}(\widehat{x}_t \mid \mathbf{x}_t; \alpha_t)$, where $\boldsymbol{\theta}_t = h(\boldsymbol{\theta}_{t+1}, \widehat{x}_t, \alpha_t)$ is the Bayesian update function. By feeding this intermediate (posterior) parameter $\boldsymbol{\theta}_t$ into a neural network $\psi(\cdot)$, $\mathbf{x}_t$'s output distribution $p_{\mathrm{O}}(\cdot)$ is parameterized as $p_{\mathrm{O}}(\mathbf{x}_t; \psi(\boldsymbol{\theta}_t))$. Finally, a *receiver distribution* $p_{\mathrm{R}}(\cdot)$ is defined as the expectation of the sender distribution with respect to the output distribution, i.e., $p_{\mathrm{R}}(\widehat{x}_t; \psi(\boldsymbol{\theta}_t), \alpha_t) := \mathbb{E}_{p_{\mathrm{O}}(\mathbf{x}_t;\psi(\boldsymbol{\theta}_t))}[p_{\mathrm{S}}(\widehat{x}_t \mid \mathbf{x}_t; \alpha_t)]$. See Figure 6 (a) for a visualization of the relationships between these distributions.

In PDMs , the joint distribution over the observation $\mathbf{x}_0$ and the intermediates $\{\mathbf{x}_t\}_t$ is defined as:

$$p(\mathbf{x}_0, \{\mathbf{x}_t\}_t|-) := p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0)) \prod_{t=1}^{T} p_{\mathrm{R}}(\widehat{x}_t; \psi(\boldsymbol{\theta}_t), \alpha_t) \tag{7}$$

. This intractable joint distribution can be approximated under the variational inference framework as follows:

$$\begin{aligned}
&\log p(\mathbf{x}_0) \\
&\geq \mathbb{E}_{p_{\mathrm{F}}(\boldsymbol{\theta}_{1:T}|-)p_{\mathrm{S}}(\{\mathbf{x}_t\}_t|-)} \left[\log \frac{p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0)) \prod_{t=1}^{T} p_{\mathrm{R}}(\widehat{x}_t; \psi(\boldsymbol{\theta}_t), \alpha_t)}{\prod_{t=1}^{T} p_{\mathrm{S}}(\widehat{x}_t \mid \mathbf{x}_t; \alpha_t)} \right] \\
&= -\sum_{t=1}^{T} \underbrace{\mathbb{E}_{p_F(\boldsymbol{\theta}_t|-)} D_{\mathrm{KL}} p_{\mathrm{S}}\left(\widehat{x}_t \mid \mathbf{x}_0; \alpha_{T:t}\right) p_{\mathrm{R}}\left(\widehat{x}_t; \psi(\boldsymbol{\theta}_t), \alpha_t\right)}_{\mathcal{L}_t^{\mathrm{R}}(\mathbf{x})} \\
&\qquad + \underbrace{\mathbb{E}_{p_F(\boldsymbol{\theta}_0|-)} \ln p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0))}_{\mathcal{L}^{\mathrm{D}}(\mathbf{x})}, \qquad (8)
\end{aligned}$$

where $p_{\mathrm{F}}(\boldsymbol{\theta}_t|-)$ is the distribution of $\boldsymbol{\theta}_t$ (see Appendix A.2 for a detailed calculation). Maximizing Eq. 8 equals minimizing the discrepancy $\mathcal{L}_t^{\mathbf{R}}(\mathbf{x})$ between the sender and receiver distributions and penalizing Distortion $\mathcal{L}^{\mathbf{D}}(\mathbf{x})$ to maximize the likelihood distribution over data.

A.2 Bayesian Flow Distribution

Bayesian flow distribution $p_{\mathrm{F}}(\cdot \mid \mathbf{x}; t)$ is the marginal distribution over input parameters at time t, given prior distribution, accuracy schedule α and Bayesian update distribution $p_U(\cdot \mid \boldsymbol{\theta}, \mathbf{x}; \alpha)$, as follows:

$$p_{\mathrm{F}}(\boldsymbol{\theta} \mid \mathbf{x}; t) = p_U\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}_0, \mathbf{x}; \beta(t)\right). \tag{9}$$

B Proofs

B.1 Derivation of ELBO for SepDiff

We derive the ELBO of SepDiff defined in Eq. (4) in Eq. (12).

We can obtain the expectation of the prior matching term over the $q(\boldsymbol{\theta}_t)$ as

$$\begin{aligned}
&-\mathbb{E}_{q(\boldsymbol{\theta}_t)} D_{KL}[q_{\boldsymbol{\phi}}(z_t|\boldsymbol{\theta}_t) \| p(z_t)] \\
=&\mathbb{E}_{q(\boldsymbol{\theta}_t)}[\mathbb{E}_{q(z_t|\boldsymbol{\theta}_t)}[\log p(z_t) - \log q_{\boldsymbol{\phi}}(z_t|\boldsymbol{\theta}_t)]] \\
=&\mathbb{E}_{q(z_t,\boldsymbol{\theta}_t)}\left[\log \frac{p(z_t)}{q_{\boldsymbol{\phi}}(z_t, \boldsymbol{\theta}_t)} + \log q(\boldsymbol{\theta}_t)\right] \\
=&\mathbb{E}_{q(z_t,\boldsymbol{\theta}_t)}\left[\log \frac{p(z_t)}{q_{\boldsymbol{\phi}}(z_t)} + \log \frac{q_{\boldsymbol{\phi}}(z_t)}{q_{\boldsymbol{\phi}}(z_t|\boldsymbol{\theta}_t)}\right] \\
=&-D_{KL}[q_{\boldsymbol{\phi}}(z_t) \| p(z_t)] - MI_{z_t,\boldsymbol{\theta}_t}.
\end{aligned} \tag{10}$$

Next, we give the scale parameters λ and γ for $D_{KL}[q_{\boldsymbol{\phi}}(z_t) \| p(z_t)]$ and $MI_{z_t,\boldsymbol{\theta}_t}$, respectively. The Eq. (10) can be rewritten as

$$\begin{aligned}
&-\lambda D_{KL}[q_{\boldsymbol{\phi}}(z_t) \| p(z_t)] - MI_{z_t,\boldsymbol{\theta}_t} + \gamma MI_{z_t,\boldsymbol{\theta}_t} \\
=&\mathbb{E}_{q_{\boldsymbol{\phi}}(z_t,\boldsymbol{\theta}_t)}\left[-\lambda \log \frac{q_{\boldsymbol{\phi}}(z_t)}{p(z_t)} - (\gamma - 1) \log \frac{q_{\boldsymbol{\phi}}(z_t)}{q_{\boldsymbol{\phi}}(z_t|\boldsymbol{\theta}_t)}\right] \\
=&-(\lambda + \gamma - 1) D_{KL}[q_{\boldsymbol{\phi}}(z_t) \| p(z_t)] \\
&-(1 - \gamma)\mathbb{E}_{q(\boldsymbol{\theta}_t)}\big[D_{KL}[q_{\boldsymbol{\theta}}(z_t|\boldsymbol{\theta}_t) \| p(z_t)]\big].
\end{aligned} \tag{11}$$

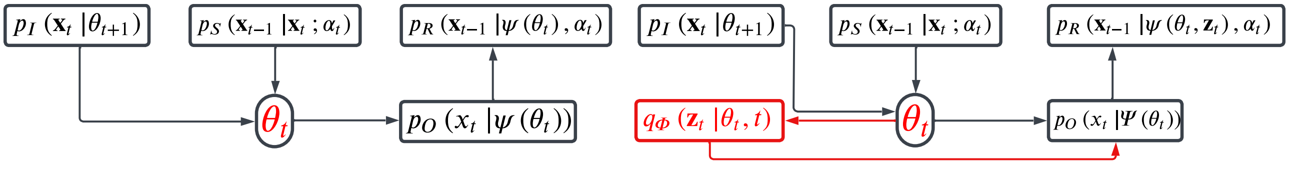
B.2 Mutual Information Learning

Unlike the rest of the terms that can be optimized directly using reparameterization tricks, the TC term cannot be directly optimized due to intractable marginal distribution $q_{\boldsymbol{\phi}}(\mathbf{z}_t)$. Here, we follow the guidance in [64] to replace the TC term with any strict divergence D, where $D\left(q_{\boldsymbol{\phi}}(\mathbf{z}) \| p(\mathbf{z})\right) = 0$ iff $q_{\boldsymbol{\phi}}(\mathbf{z}) = p(\mathbf{z})$. We implement the Maximum-Mean Discrepancy (MMD) [64] from the divergence

[2] It is noted that the index t is used reversely in [15]. We make such changes to be consistent with the diffusion model settings [20, 41].

Table 3: Examples of detailed distribution formats in PDMs . $\theta_{t+1} = \{\mu_{t+1}, \rho_{t+1}^{-1}\})$.

Data type	$p_{\mathrm{I}}(\mathbf{x}_t\|\boldsymbol{\theta}_{t+1})$	$p_{\mathrm{S}}(\widehat{x}_t\|\mathbf{x}_t;\alpha_t)$	$\boldsymbol{\theta}_t = h(\boldsymbol{\theta}_{t+1},\widehat{x}_t,\alpha_t)$
Continuous data	$\mathcal{N}(\mathbf{x}_t;\mu_{t+1},\rho_{t+1}^{-1})$	$\mathcal{N}(\widehat{x}_t;\mathbf{x},\alpha_t^{-1})$	$\mu_t = \frac{\alpha_t\widehat{x}_t+\rho_{t+1}\mu_{t+1}}{\alpha_t+\rho_{t+1}}$
Discrete data	$\mathrm{Cat}(\mathbf{x}_t;\frac{1}{K}\cdot\mathbf{1})$	$\mathcal{N}(\widehat{x}_t;\alpha_t K\mathbf{e}_{\mathbf{x}_t}-\alpha_t,\alpha_t K\mathbf{I})$	$\boldsymbol{\theta}_t = \frac{e^{\widehat{x}_t}\theta_{t+1}}{\sum_k e^{\mathbf{x}_{t-1,k}}\theta_{t+1,k}}$
Data type	$p_{\mathrm{O}}(\mathbf{x}_t\|\boldsymbol{\theta}_t)$	$p_{\mathrm{R}}(\widehat{x}_t\|\psi(\boldsymbol{\theta}_t),\alpha_t)$	
Continuous data	$\delta(\mathbf{x}_t-\psi(\boldsymbol{\theta}_t))$	$\mathcal{N}(\widehat{x}_t;\psi(\boldsymbol{\theta}_t),\alpha_t^{-1})$	
Discrete data	$\mathrm{Cat}(\mathrm{softmax}(\psi(\boldsymbol{\theta}_t)))$	$\sum_k p_O(k;\psi(\boldsymbol{\theta}_t))\mathcal{N}(\widehat{x}_t;\alpha_t K\mathbf{e}_k-\alpha_t,\alpha_t K\mathbf{I})$	


$p_I(\mathbf{x}_t|\theta_{t+1})$
$p_S(\mathbf{x}_{t-1}|\mathbf{x}_t;\alpha_t)$
$p_R(\mathbf{x}_{t-1}|\psi(\theta_t),\alpha_t)$
θ_t
$p_O(x_t|\psi(\theta_t))$
$p_I(\mathbf{x}_t|\theta_{t+1})$
$p_S(\mathbf{x}_{t-1}|\mathbf{x}_t;\alpha_t)$
$p_R(\mathbf{x}_{t-1}|\psi(\theta_t,\mathbf{z}_t),\alpha_t)$
$q_\Phi(\mathbf{z}_t|\theta_t,t)$
θ_t
$p_O(x_t|\Psi(\theta_t))$

(a) **Probabilistic Graphic Structure of PDMs** (b) **Probabilistic Graphic Structure of SepDiff**

Figure 6: The relationships between distributions in PDMs (a) and SepDiff (b).

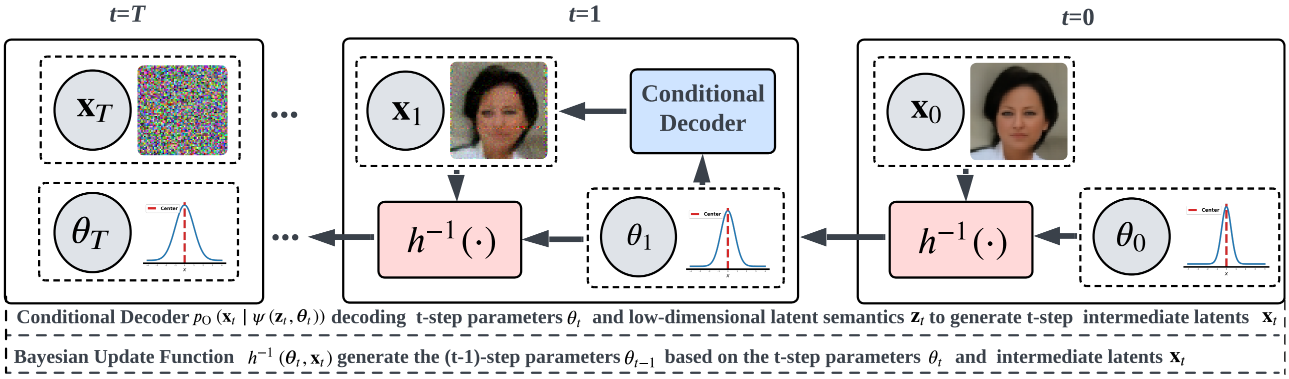

t=T
t=1
t=0
$\mathbf{x}_T$
θ_T
$\mathbf{x}_1$
Conditional Decoder
$h^{-1}(\cdot)$
θ_1
$\mathbf{x}_0$
$h^{-1}(\cdot)$
θ_0
Center
Conditional Decoder $p_O(\mathbf{x}_t | \psi(\mathbf{z}_t,\theta_t))$ decoding t-step parameters θ_t and low-dimensional latent semantics $\mathbf{z}_t$ to generate t-step intermediate latents $\mathbf{x}_t$
Bayesian Update Function $h^{-1}(\theta_t,\mathbf{x}_t)$ generate the (t-1)-step parameters θ_{t-1} based on the t-step parameters θ_t and intermediate latents $\mathbf{x}_t$

Figure 7: Reverse-sampling process in PDMs .

family. MMD is a statistical measure that quantifies the difference between two probability distributions by comparing their mean embeddings in a high-dimensional feature space. By defining the kernel function $\kappa(\cdot,\cdot)$, D_{MMD} is denoted as:

$$D_{\mathrm{MMD}}\left(q(\cdot)\|p(\cdot)\right) = \mathbb{E}_{p(\mathbf{z}),p(\mathbf{z}')}\left[\kappa\left(\mathbf{z},\mathbf{z}'\right)\right] - 2\mathbb{E}_{q(\mathbf{z}),p(\mathbf{z}')}\left[\kappa\left(\mathbf{z},\mathbf{z}'\right)\right] + \mathbb{E}_{q(\mathbf{z}),q(\mathbf{z}')}\left[\kappa\left(\mathbf{z},\mathbf{z}'\right)\right]. \tag{13}$$

B.3 Evaluation Metrics

FID for Generation We employ clean-fid [34] in [3] to evaluate the **unconditional generation task** and **sample quality task** quality. The *Fréchet Inception Distance* (FID) evaluation process can be outlined as follows: First, a raw sample set is derived from the dataset, typically comprising collected and downsampled samples that may undergo resizing and compression before training. Second, the generated image set is prepared, where images are often stored as unsigned 16-bit integers, introducing quantization and potential additional compression. FID assesses how effectively a generative model replicates the training distribution by approximating real and generated samples as Gaussians in the feature space of an Inception Network and computing their Wasserstein distance. As a distributional metric, FID highlights both sample fidelity and diversity. In Tables 2, we measure FID between 10k random samples from raw dataset and 10k randomly generated samples. **DCI and TAD for Disentanglement** For comprehensive and fair quantitative evaluation, we select the following measures guided by [8] to assess the **disentanglement task** : (1) prediction-based metric: Disentanglement, Completeness and Informativeness (DCI) (2) information-based metric: Total AUROC Difference (TAD) [58]. **AUROC for Classification** To assess **latent classification task** performance, we train a logistic regression model on the auxiliary latent encodings of images to predict labels. The evaluation metric is AUROC, and in cases where multiple annotations are present,

[3] https://github.com/GaParmar/clean-fid

$$
\begin{aligned}
&\log p(\mathbf{x}_0) \\
&= \log \int_{\{\mathbf{z}_t\}_t} \int_{\{\mathbf{x}_t\}_t} p\left(\mathbf{x}_0, \{\mathbf{x}_t\}_t, \{\mathbf{z}_t\}_t \mid \boldsymbol{\theta}_0, \alpha\right) \mathrm{d}\{\mathbf{z}_t\}_t \mathrm{d}\{\mathbf{x}_t\}_t \\
&= \log \int_{\{\mathbf{z}_t\}_t} \int_{\{\mathbf{x}_t\}_t} \int_{\{\boldsymbol{\theta}_t\}_t} p(\{\boldsymbol{\theta}_t\}_t|-) p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0, \mathbf{z}_0)) \prod_{t=T}^{1} p(\mathbf{z}_t) \mathbb{E}_{p_{\mathrm{O}}(\mathbf{x}_t;\psi(\boldsymbol{\theta}_t,\mathbf{z}_t))} [p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_t; \alpha_t)] \\
&\qquad \mathrm{d}\{\mathbf{z}_t\}_t \mathrm{d}\{\mathbf{x}_t\}_t \mathrm{d}\{\boldsymbol{\theta}_t\}_t \\
&= \log \int_{\{\mathbf{z}_t\}_t} \int_{\{\mathbf{x}_t\}_t} \int_{\{\boldsymbol{\theta}_t\}_t} p(\{\boldsymbol{\theta}_t\}_t|-) \frac{p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0, \mathbf{z}_0)) \prod_{t=T}^{1} p(\mathbf{z}_t) \mathbb{E}_{p_{\mathrm{O}}(\mathbf{x}_t;\psi(\boldsymbol{\theta}_t,\mathbf{z}_t))} [p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_t; \alpha_t)]}{\prod_{t=1}^{T} p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_t; \alpha_t) q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)} \\
&\qquad\qquad \cdot \prod_{t=1}^{T} p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_t; \alpha_t) q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t) \mathrm{d}\{\mathbf{z}_t\}_t \mathrm{d}\{\mathbf{x}_t\}_t \mathrm{d}\{\boldsymbol{\theta}_t\}_t \\
&\geq \mathbb{E}_{\prod_{t=1}^{T} p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_t;\alpha_t) q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t,t) p(\boldsymbol{\theta}_t|-)} \left[\log \frac{p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0, \mathbf{z}_0)) \prod_{t=T}^{1} p(\mathbf{z}_t) \mathbb{E}_{p_{\mathrm{O}}(\mathbf{x}_t;\psi(\boldsymbol{\theta}_t,\mathbf{z}_t))} [p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_t; \alpha_t)]}{\prod_{t=1}^{T} p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_t; \alpha_t) q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)} \right] \\
&= \sum_{t=1}^{T} \mathbb{E}_{p_F(\boldsymbol{\theta}_t|-)} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_t)} \left\{ \mathbb{E}_{p_{\mathrm{S}}(\mathbf{x}_{t-1} \mid \mathbf{x}_0;\alpha_{T:t})} \left[\log \frac{p_{\mathrm{S}}\left(\mathbf{x}_{t-1} \mid \mathbf{x}_0; \alpha_{T:t}\right)}{p_{\mathrm{R}}\left(\mathbf{x}_{t-1}; \psi(\boldsymbol{\theta}_t, \mathbf{z}_t), \alpha_t\right)} \right] \right. \\
&\qquad\qquad \left. - \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_t \mid \boldsymbol{\theta}_t)} \left[\log \frac{q_{\boldsymbol{\phi}}\left(\mathbf{z}_t \mid \boldsymbol{\theta}_t\right)}{p(\mathbf{z}_t)} \right] \right\} + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_0,\boldsymbol{\theta}_0)} \left[\ln p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0, \mathbf{z}_0)) \right] \\
&= - \sum_{t=1}^{T} \mathbb{E}_{p_F(\boldsymbol{\theta}_t|-)} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_t)} \{ D_{\mathrm{KL}} [p_{\mathrm{S}}\left(\mathbf{x}_{t-1} \mid \mathbf{x}_0; \alpha_{T:t}\right) \| p_{\mathrm{R}}\left(\mathbf{x}_{t-1}; \psi(\boldsymbol{\theta}_t, \mathbf{z}_t), \alpha_t\right)] \\
&\qquad\qquad - D_{\mathrm{KL}} [q_{\boldsymbol{\phi}}\left(\mathbf{z}_t \mid \boldsymbol{\theta}_t\right) \| p(\mathbf{z}_t)] \} + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_0,\boldsymbol{\theta}_0)} \left[\ln p_{\mathrm{O}}(\mathbf{x}_0; \psi(\boldsymbol{\theta}_0, \mathbf{z}_0)) \right] := \mathcal{L}_{\mathsf{ELBO}}
\end{aligned} \tag{12}
$$

we report the average accuracy/AUROC across all predicted labels. The dataset is split, with 80% allocated for training and the remaining 20% reserved for testing. Performance is measured on the test set using AUROC. To ensure robustness, this evaluation follows a 5-fold cross-validation protocol, with the final results presented as the mean ± one standard deviation.

B.4 Interpolation

The **latent space interpolation task** can be described as follows. Firstly, we noise source images to generate latent pairs by sender distribution, $< \mathbf{x}_1^1, \mathbf{x}_1^2 >$, where $\mathbf{x}_1^1 \sim q(\cdot \mid \mathbf{x}_N^1)$ and $\mathbf{x}_1^2 \sim q(\cdot \mid \mathbf{x}_N^2)$. Then, we implement two methods from [39] to generate four interpolated latent pairs $\bar{\mathbf{x}}_{1:4}$, i.e., linear interpolation, and spherical interpolation:

$$
\begin{aligned}
\bar{\mathbf{x}}_i &= (1 - \lambda_{\mathtt{inter}}) \mathbf{x}_0^1 + \lambda_{\mathtt{inter}} \mathbf{x}_0^2, \\
\bar{\mathbf{x}}_i &= \frac{\sin((1 - \alpha_{\mathtt{inter}}) \theta_{\mathtt{inter}})}{\sin(\theta_{\mathtt{inter}})} \mathbf{x}_0^1 + \frac{\sin(\alpha_{\mathtt{inter}} \theta_{\mathtt{inter}})}{\sin(\theta_{\mathtt{inter}})} \mathbf{x}_0^1,
\end{aligned} \tag{14}
$$

where $\lambda_{\mathtt{inter}}$ is the scale coefficient, $\alpha_{\mathtt{inter}} \in [0, 1]$ denotes the interpolation steps, and $\theta_{\mathtt{inter}} = \arccos \left(\frac{(\mathbf{x}_0^1)^\top \mathbf{x}_0^2}{\|\mathbf{x}_0^1\| \|\mathbf{x}_0^2\|} \right)$ is the angle between $\mathbf{x}_0^1$ and $\mathbf{x}_0^2$.

B.5 Model Structures

Encoder Architecture In our proposed encoder architecture, the self-encoder $q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$ also conditions on step $(t + 1)$'s upsampling layers $\{\mathbf{u}_{t+1,l}\}_{l=1}^{L}$, where L is the number of layers in the U-Net architecture. For the l-th upsampling layer $\mathbf{u}_{t+1,l}$ at step $t + 1$, we upsample it to the size of $\mathbf{x}_t$, update by the Bayesian update function, and pass through a bottleneck layer $B_l(\cdot)$ [16] to the low-dimensional size. As a result, the self-encoder is defined as:

$$
q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t) = \mathcal{N}\left(\mathbf{z}_t; g_\mu(\boldsymbol{\theta}_t, \{\mathbf{u}_{t+1,l}\}_{l=1}^{L}, t), g_\sigma(\boldsymbol{\theta}_t, \{\mathbf{u}_{t+1,l}\}_{l=1}^{L}, t)^2\right), \tag{15}
$$

where $g_\mu(\cdot), g_\sigma(\cdot)$ use the same structure as:

$$
\begin{aligned}
& g_\mu(\boldsymbol{\theta}_t, \{\mathbf{u}_{t+1,l}\}_{l=1}^{L}, t), g_\sigma(\boldsymbol{\theta}_t, \{\mathbf{u}_{t+1,l}\}_{l=1}^{L}, t) \\
&:= \sum_{l=0}^{L} \omega_l \cdot B_l(h(\mathbf{x}_t, \mathbf{u}_{t+1,l})) + \omega_{L+1} \cdot B_{L+1}(\boldsymbol{\theta}_t),
\end{aligned} \tag{16}
$$

where ω_l is the mixing weight of the l-th layer.

Algorithm 1: Unconditional generation process of SepDiff.

```
Input: number of steps $T \in \mathbb{N}$, $\sigma_1 \in \mathbb{R}^+$, optimized
       parameters $\boldsymbol{\phi}, \boldsymbol{\theta}$.
Output: $\boldsymbol{\theta}_0$.
$\boldsymbol{\theta}_T \sim \mathcal{N}(\boldsymbol{\theta}_T; \mathbf{0}, \mathbf{I})$
for $t = T$ to 1 do
    $\mathbf{z}_t \sim q_{\boldsymbol{\phi}}(\mathbf{z}_t|\boldsymbol{\theta}_t, t)$
    $\boldsymbol{\theta}_{t-1} \sim p_{\boldsymbol{\theta}}(\boldsymbol{\theta}_{t-1}|\boldsymbol{\theta}_t, \mathbf{z}_t)$
end
Return $\boldsymbol{\theta}_0$
```