LDM-is-AE: Latent Diffusion Model is an Auto-Encoder for End-to-End Image Generation

Zhengqiang Zhang[1,2], Lingchen Sun[1,2], Rongyuan Wu[1,2], Qiaosi Yi[1,2], Xiangtao Kong[1,2], Chaodong Xiao[1,2], Lei Zhang[1,2,†]

[1]The Hong Kong Polytechnic University [2]OPPO Research Institute

https://github.com/PolyU-VCLab/LDMisAE

Abstract

Latent Diffusion Models (LDMs) typically adopt a two-stage pipeline: an auto-encoder (AE) is first pre-trained to define a latent space, then a diffusion model is trained to perform denoising within it. Such a two-stage design introduces a representation mismatch, as the latent space is optimized for reconstruction rather than adapting the denoising dynamics. We reveal that the **LDM** itself **is** an **AE**, and consequently present **LDM-is-AE**, an end-to-end one-stage LDM training framework that eliminates the need for a separately trained tokenizer. Our key observation is that the LDM backbone actually performs a *latent-to-feature-to-latent* transformation at each denoising step, which can be interpreted as an internal decoding–encoding process. Leveraging this structure, we split the DiT backbone into two reciprocal components, **DiT-E** (i.e., DiT Encoding) and **DiT-D** (i.e., DiT Decoding), and impose image-space supervision on the intermediate features across all timesteps. Our model encourages the internal representation to align with the image domain throughout denoising, thereby establishing an explicit *latent-to-image-to-latent* path. At the zero-noise timestep, our model further performs an *image-to-latent-to-image* mapping, corresponding to an auto-encoding process. As a result, LDM-is-AE jointly learns latent representations and denoising dynamics in an end-to-end manner, yielding a diffusion-native latent space tailored to the generation process. Experiments demonstrate that LDM-is-AE exhibits highly competitive generation performance, achieving an FID of 1.80 and 1.90 on 256×256 and 512×512 class-conditional image generation, respectively.

1 Introduction

Recent advances in latent diffusion models (LDMs)–including those seminal works such as LDM [29], DiT [26], SiT [23], LightningDiT [40], REPA [42], Stable Diffusion [29, 27, 11], and FLUX [19]–have significantly improved the study and practical deployment of image generation. Most of these methods follow a two-stage training paradigm: an auto-encoder is first pre-trained to map an image into a compact latent representation (e.g., converting a 256×256 image into a 32×32 latent [29]), then a diffusion model is trained in this latent space to achieve feature transformation. The encoder and decoder are typically pre-trained using reconstruction objectives on auxiliary datasets (e.g., OpenImages [17]) to obtain a compact yet reconstructible representation, thereby simplifying the optimization landscape for subsequent diffusion training.

The latent space strongly affects diffusion model optimization, and hence many prior works focus on improving the latent representation. Early studies [6, 41, 1, 45] mainly pursue more aggressive compression to facilitate diffusion model learning efficiency. For example, DCAE [6] achieves

[†]Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

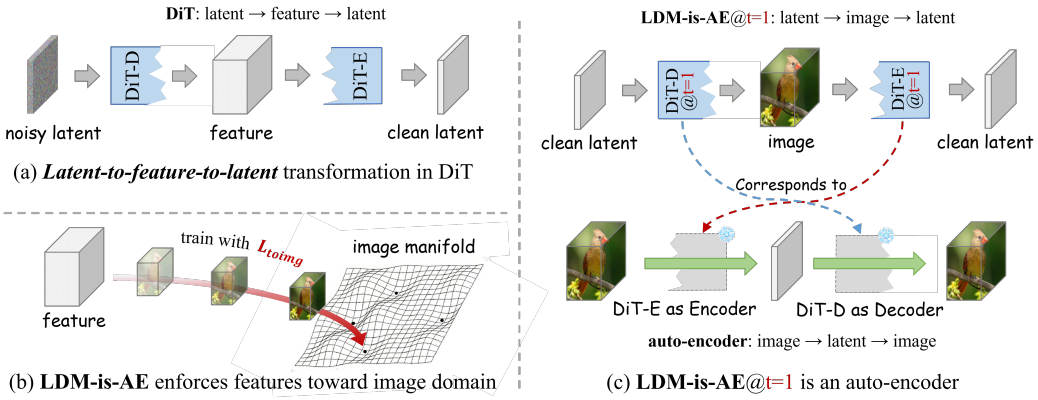


Figure 1: **Illustration of the auto-encoding nature of LDM.** (a) The LDM backbone (i.e., a DiT) naturally performs a *latent-to-feature-to-latent* transformation. (b) Image-space supervision can align intermediate features with the image domain across timesteps. (c) At the zero-noise timestep ($t = 1$), an explicit *latent-to-image-to-latent* path is established, which in turn enables the reverse *image-to-latent-to-image* auto-encoding process.

satisfactory reconstruction under extreme compression ratios. TiTok [41] and FlexTok [1] transform 2D latents into compact 1D token sequences using transformers, substantially reducing the number of latent tokens. GPSToken [45] adopts spatially adaptive 2D Gaussians for image representation to improve reconstruction quality while preserving compactness. Beyond dimensionality reduction, recent works [42, 40, 3] have shown that latent structure also plays a crucial role in diffusion optimization. REPA [42] and VAVAE [40] align latent representations with pre-trained vision foundation models to inject semantic structure, while MAETok [3] uses multiple auxiliary decoders to align latents with diverse target features. Other methods, such as RAE [46] and SVG [32], leverage pre-trained vision encoders to provide fixed semantic representations for latent diffusion models. Despite these advances, existing latent diffusion models still largely rely on a two-stage training paradigm, in which a pre-trained auto-encoder defines a fixed latent space for subsequent diffusion modeling. This two-stage design incurs **substantial pre-training overhead** and introduces a **representation mismatch**: the latent space is optimized primarily for reconstruction and keeps fixed during diffusion model training, limiting its adaptation to the image generation process.

In this paper, we reveal that the **LDM** itself **is** an **A**uto-**E**ncoder, and present **LDM-is-AE**, an end-to-end one-stage LDM training framework that eliminates the need for a separately pre-trained tokenizer. Actually, the diffusion backbone, such as the diffusion transformer (DiT), naturally performs a *latent-to-feature-to-latent* transformation through the denoising process, as illustrated in Fig. 1(a). This transformation can be interpreted as an internal decoding–encoding process, suggesting that latent diffusion naturally follows an inverse auto-encoding structure. Unfortunately, this structure remains unexploited in LDM formulations due to the lack of an explicit connection between the intermediate feature space and the image domain. To fill this gap, we partition the diffusion backbone into two reciprocal components, termed **DiT-D** and **DiT-E**, which correspond to the *decoding* and *encoding* parts of the transformation, respectively. We then align the intermediate representations with the image domain through an image-space supervision loss $\mathcal{L}_{\text{toimg}}$, making this internal structure explicit and establishing a *latent-to-image-to-latent* path during denoising. At the zero-noise timestep ($t = 1$), this path can be equivalently viewed in reverse as an *image-to-latent-to-image* auto-encoding process, as illustrated in Fig. 1(c). With our formulation, the DiT backbone itself serves as an auto-encoder, enabling end-to-end joint learning of feature representations and diffusion denoising.

To align the intermediate features with the image domain, we insert a lightweight MLP head at the end of DiT-D so that the resulting feature directly matches the channel dimension of the pixel-unshuffled image, and then impose image-space supervision in this aligned space. We further insert a second lightweight MLP head at the beginning of DiT-E to project the aligned feature back to the hidden size expected by the original diffusion backbone. Directly forcing high-dimensional intermediate features to stay in the image domain may over-constrain the representation and harm the generative performance. To mitigate this issue, we introduce *time-aware auxiliary feature mixing*, which preserves extra feature capacity during denoising while ensuring a valid auto-encoding path at the zero-noise timestep. Furthermore, we train DiT-E in a *residual learning* manner on top of an

interpolated image-aligned base space to stabilize the training process. Experiments on ImageNet show that LDM-is-AE attains an FID of 1.80 and an IS of 314 for 256×256 generation, and an FID of 1.90 and an IS of 320 for 512×512 generation, using a **generator-only one-stage pipeline**. It remains competitive with strong end-to-end latent baselines while avoiding auxiliary encoders/decoders, thereby reducing training computation.

In summary, our contributions are threefold:

- We reveal that latent diffusion backbones exhibit an auto-encoding behavior, and propose **LDM-is-AE**, a one-stage training framework that jointly learns latent representations and diffusion models in an end-to-end manner.
- We explicitly decompose the DiT backbone into a **DiT-D** and a **DiT-E** part and impose image-space supervision on intermediate representations, establishing a *latent-to-image-to-latent* path during the denoising process and recovering an *image-to-latent-to-image* auto-encoding process at the zero-noise timestep.
- Extensive experiments on ImageNet demonstrate that **LDM-is-AE** consistently yields high image generation quality with less computations, validating the effectiveness of learning diffusion-native latent models for image generation in an end-to-end manner.

2 Related Work

Latent Diffusion Models. Early LDMs largely inherit U-Net backbones from pixel-space diffusion [29, 27], whereas recent works [26, 2, 11, 19, 44, 34] usually adopt transformer-based denoisers, including DiT [26], U-ViT [2], PixArt-α [5], DDT [37], Lumina [28], SD3 [11], and FLUX [19], reflecting a shift toward scalable transformer architectures for latent diffusion. Several works improve diffusion training through alternative formulations or optimization strategies. For example, SiT [23] reformulates diffusion from an interpolant-based flow-matching perspective. MaskDiT [12] accelerates training with masked transformers while REPA [42] improves optimization by aligning diffusion features with pre-trained visual representations. Despite these advances, most methods still operate on a fixed latent space defined by a separately trained autoencoder.

Autoencoder Design. Classical tokenizers for latent diffusion are built on VAE and VQ paradigms [15, 35, 10], which learn continuous or discrete latent representations through reconstruction objectives. Recent work has explored both higher compression ratio and more structured latent representation. DCAE [6] targets high-fidelity reconstruction under extreme compression ratios. TiTok [41] and FlexTok [1] convert 2D latent grids into compact 1D token sequences, substantially reducing the number of latent tokens. GPSToken [45] uses spatially adaptive 2D Gaussians to improve reconstruction quality while preserving compactness. Meanwhile, several methods aim to enrich the semantic structure of the latent space. MAETok [3] aligns latents with multiple target features through auxiliary decoders. REPA [42] and VAVAE [40] align latent representations with pre-trained vision models, while RAE [46] and SVG [32] adopt frozen visual foundation models as encoders. EQ-VAE [16] and VAVAE [40] further show that latent spaces optimized for reconstruction are not necessarily well suited to the following image generation. Nevertheless, these two-stage designs incur substantial pretraining overhead and introduce a representation mismatch since the latent space is optimized primarily for reconstruction rather than denoising and generation.

Joint Tokenization and Generation. Several recent methods have attempted to jointly train tokenization and generation rather than treating them as disjoint stages. REPA-E [20] enables end-to-end VAE+diffusion training with a representation-alignment loss, but still maintains separate encoder, decoder, and diffusion modules. UNITE [9] shares a generative encoder between tokenization and denoising, but still relies on a separate decoder for reconstruction. DSD [38] uses a single network as encoder, decoder, and denoiser, but combines these roles in a modular rather than coupled manner.

Despite the progress in joint tokenization and generation, prior approaches still treat autoencoding and denoising as separate modules or explicit multi-objectives (reconstruction and denoising). In contrast, our work adopts a different perspective: we show that latent diffusion backbones already contain an internal decoding–encoding structure. By explicitly aligning the intermediate feature with the image domain, this hidden capability can be exposed as a *latent-to-image-to-latent* path, which naturally induces an *image-to-latent-to-image* auto-encoding view. As a result, we can conclude that latent representation learning becomes a native part of diffusion modeling itself, rather than a separately pre-trained stage appended to it.

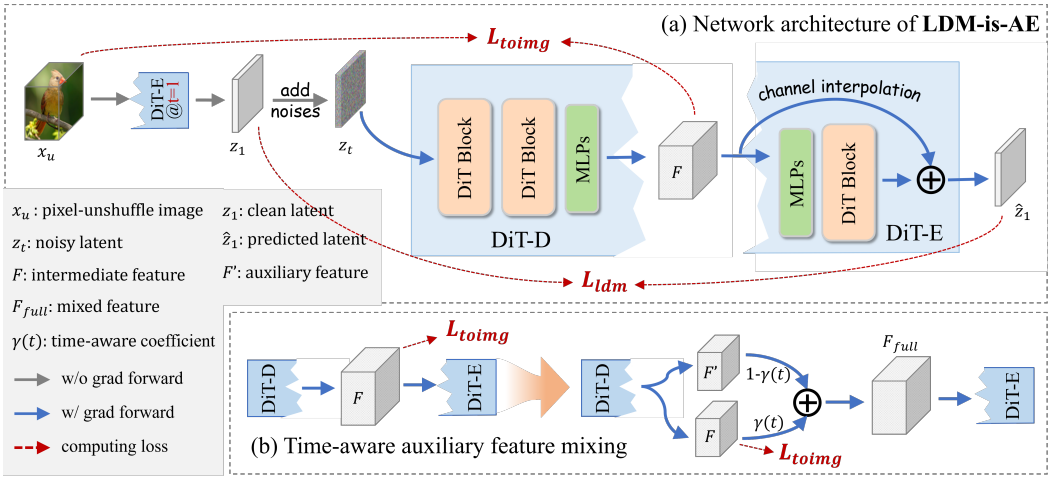


Figure 2: Architecture and training design of LDM-is-AE. (a) Network architecture and training pipeline. (b) Time-aware auxiliary feature mixing.

3 Methodology

This section presents the proposed **LDM-is-AE** framework. We first review the conventional two-stage latent diffusion pipeline in Sec. 3.1, then in Sec. 3.2 we reveal the implicit *latent-to-feature-to-latent* behavior of the diffusion backbone and introduce LDM-is-AE by turning this hidden decoding–encoding process into an auto-encoder through image-domain supervision. Finally, in Sec. 3.3 we present the one-stage training algorithm.

3.1 Preliminaries

To reduce computational cost and ease optimization, latent diffusion models [29, 26, 23] typically follow a two-stage paradigm. First, an auto-encoder consisting of an encoder $\mathcal{E}$ and a decoder $\mathcal{D}$ is pre-trained to compress an image x into a compact latent space:

$$x \to z_1 = \mathcal{E}(x) \to \hat{x} = \mathcal{D}(z_1), \tag{1}$$

which forms an explicit *image-to-latent-to-image* path. The auto-encoder is typically optimized with reconstruction losses such as pixel ℓ_2, perceptual LPIPS [43], and adversarial objectives [29, 10].

After the tokenizer is fixed, diffusion learning is performed in the latent space. Recent latent diffusion frameworks often adopt flow matching [23, 40, 42]. Given a clean latent $z_1 = \mathcal{E}(x)$ and Gaussian noise $z_0 \sim \mathcal{N}(0, I)$, the noisy latent at timestep $t \in [0, 1]$ is defined as:

$$z_t = tz_1 + (1-t)z_0, \tag{2}$$

where $t = 0$ corresponds to pure noise and $t = 1$ to the clean latent. The diffusion backbone f_θ is trained to recover the clean latent:

$$\hat{z}_1 = f_\theta(z_t, t, c), \tag{3}$$

where c denotes conditioning, such as class labels or text prompts. Following JiT [21], we adopt the v-loss implementation under the z_1-prediction parameterization:

$$\mathcal{L}_{\text{ldm}} = \mathbb{E}_{t,z_0,z_1}\left[\|(f_\theta(z_t,t,c) - z_t)/(1-t) - (z_1 - z_t)/(1-t)\|^2\right]. \tag{4}$$

At inference time, an ODE solver progressively denoises a random sample $z_0 \sim \mathcal{N}(0, I)$ into z_1, which is then decoded by $\mathcal{D}$ to obtain the final image.

3.2 LDM-is-AE: Formulation

Motivation. A standard DiT-style diffusion backbone maps a noisy latent z_t to a higher-dimensional intermediate feature and then projects that feature back to latent space to obtain $\hat{z}_1$. Thus, the diffusion denoising process naturally forms a *latent-to-feature-to-latent* transformation:

$$z_t \to F \to \hat{z}_1. \tag{5}$$

As shown in Fig. 1 (a), the first half of the DiT expands the compact latent into a richer intermediate representation and the second half compresses it back to latent space, admitting a natural decoding–encoding interpretation. Accordingly, we conceptually decompose the backbone into two parts:

- **DiT-D** (decoding part): the earlier blocks that map z_t to the intermediate feature F;
- **DiT-E** (encoding part): the later blocks that map F back to latent space and produce $\hat{z}_1$.

However, in standard LDMs the feature F is not explicitly tied to the image domain. As a result, this internal decoding–encoding process is hidden and cannot directly guide representation learning. We aim to make this hidden structure explicit by aligning the intermediate feature with the image domain. Once this connection is established, the denoising backbone acts not only as a latent denoiser but also as an auto-encoder within the one-stage diffusion training pipeline.

Image-space Alignment. Let $F \in \mathbb{R}^{3p^2 \times h_F \times w_F}$ denote the image-aligned intermediate feature produced from the noisy latent z_t, where z_t is constructed from the latent of image x according to Eq. 2. Lightweight MLP projections are used only at the DiT-D/DiT-E interface to match the feature dimensions. We reshape $x \in \mathbb{R}^{3\times h\times w}$ with pixel-unshuffle using stride p, obtaining:

$$x_u = \mathrm{PixelUnshuffle}(x, p) \in \mathbb{R}^{3p^2 \times h_F \times w_F}. \tag{6}$$

We then impose image-space supervision directly on the aligned representations:

$$\mathcal{L}_{\text{toimg}} = \mathbb{E}_{t,x}\Big[w(t)\big(\|F - x_u\|^2 + w_{\text{lpips}}\,\mathrm{LPIPS}(F, x_u)\big)\Big], \tag{7}$$

where $w(t)$ is a time-dependent weight that places greater emphasis on late denoising steps, and w_{lpips} controls the perceptual term. We set $w(t) = \frac{1}{(1-t)^2}$ by default to match the loss magnitude of v-loss across different timestep. With this supervision, the hidden transformation in Eq. 5 becomes an explicit *latent-to-image-to-latent* path.

Auto-encoder. One key observation can be made at the zero-noise timestep. When $t = 1$, the input to the diffusion backbone is the clean latent z_1, so the denoising path reduces to:

$$z_1 \to F \approx x_u \to \hat{z}_1. \tag{8}$$

As shown conceptually in Fig. 1 (c), the first half of the backbone maps the clean latent z_1 to an image-aligned intermediate representation $F \approx x_u$, while the second half maps this image-aligned representation back to latent space. That is, at $t = 1$, DiT-D plays the role of an internal decoder from latent space to image-domain features, and DiT-E plays the role of an internal encoder from image-domain features back to latent space. This yields a *latent-to-image-to-latent* path inside the diffusion model. Once this path is established, we can view its reverse as an *image-to-latent-to-image* auto-encoding process: the image-aligned representation serves as the image-side input, DiT-E encodes it into latent space, and DiT-D decodes the latent back to the image-aligned representation. In this sense, the explicit *latent-to-image-to-latent* path naturally builds an auto-encoder without introducing a separately pre-trained tokenizer.

Time-aware Auxiliary Feature Mixing. Directly forcing the full high-dimensional feature F to stay in the image domain may over-constrain the representation and reduce the capacity for diffusion denoising. To preserve additional feature freedom while keeping the clean-step auto-encoding path valid, we introduce *time-aware auxiliary feature mixing*. Concretely, DiT-D outputs a tensor whose channels are evenly split into the original image-aligned feature F and the auxiliary feature F'. As illustrated in Fig. 2 (b), the feature consumed by DiT-E is defined as:

$$F_{\text{full}} = \gamma(t)F + (1 - \gamma(t))F', \tag{9}$$

where $\gamma(t) = t^k$ is a gating function such that $\gamma(t) \to 1$ as $t \to 1$. In this way, the image-aligned feature F dominates near the clean endpoint, while the auxiliary feature F' contributes more at noisy timesteps to preserve denoising capacity. The image-space loss is applied only to F, and the auxiliary branch provides additional support without interfering the auto-encoding path at $t = 1$.

Residual DiT-E Design. DiT-E is structurally designed as a residual latent predictor, as illustrated in Fig. 2 (a). Specifically, it normalizes the model output and adds it to a channel-interpolated skip connection derived from the input feature. This design preserves the global structure encoded in the image-aligned feature, while enabling DiT-E to focus on the corrective component required for accurate latent recovery, thereby stabilizing training.

Algorithm 1: Training loop of LDM-is-AE

Inputs: training set $\mathcal{X}$, total iterations T;
for $i = 1, \ldots, T$ **do**
  $(x, c) = \texttt{sample_batch}(\mathcal{X})$, $t \sim \mathcal{U}[0, 1]$, $z_0 \sim \mathcal{N}(0, I)$;
  // AE Encoding;
  $x_u = \texttt{pixel_unshuffle}(x, p)$; `// Eq. 6`
  `with torch.no_grad():`
    $z_1 = \texttt{dit_e}(x_u, t{=}1, c)$; `// image-to-latent at` $t{=}1$
  // LDM Denoising;
  $z_t = tz_1 + (1 - t)z_0$;
  $F, F' = \texttt{dit_d}(z_t, t, c)$; `// split output to obtain` F `and` F'
  $F_{\text{full}} = \gamma(t)F + (1 - \gamma(t))F'$; `// auxiliary feature mixing`
  $\hat{z}_1 = \texttt{dit_e}(F_{\text{full}}, t, c)$; `// map mixed feature back to latent`
  $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{ldm}}(\hat{z}_1, z_1) + w_{\text{toimg}}\mathcal{L}_{\text{toimg}}(F, x_u)$; `// Eqs. 7, 4 and 10`
  $\mathcal{L}_{\text{total}}$`.backward();`
  `optimizer.step();`
end

3.3 One-Stage Training

Our model is trained in an end-to-end manner with a simple one-stage objective that combines latent denoising and image-domain alignment:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{ldm}} + w_{\text{toimg}}\mathcal{L}_{\text{toimg}}, \tag{10}$$

where w_{toimg} balances the latent denoising objective and the image-space supervision term.

The training pipeline is straightforward. Given an image x, we first construct its image-side target x_u according to Eq. 6 and obtain the clean latent z_1 by applying DiT-E to x_u at $t = 1$. In implementation, this AE encoding step is detached from gradient computation, avoiding explicit reconstruction-objective optimization on the encoding path and preserving the auto-encoding behavior. We then construct z_t according to Eq. 2, feed z_t into DiT-D to obtain the original image-aligned feature F and the auxiliary feature F', and supervise F by Eq. 7. After time-aware auxiliary feature mixing in Eq. 9, the transformed feature is fed to DiT-E, which maps it back to latent space and predicts $\hat{z}_1$. During training, $\mathcal{L}_{\text{toimg}}$ is applied to the intermediate feature F through Eq. 7, while $\mathcal{L}_{\text{ldm}}$ is applied to the final latent prediction through Eq. 4. All components are optimized jointly rather than being separately trained into two stages.

Algorithm 1 summarizes the training loop of LDM-is-AE in a code-like form.

4 Experiments

As in prior works [26, 23, 42, 46, 40, 9, 7, 36, 21], we evaluate LDM-is-AE on class-conditional ImageNet generation. Sec. 4.1 describes the experimental settings; Sec. 4.2 presents the main results; Sec. 4.3 analyzes the latent space; and Sec. 4.4 reports the key ablation studies.

4.1 Experimental Settings

Experiment Setup. Our model is trained on ImageNet [30]. Training images are center-cropped, resized to 256×256, and randomly horizontally flipped. Following JiT [21], we optimize all models with AdamW [22]. The default training recipe uses a global batch size of 1024, a learning-rate warmup of 5 epochs, zero weight decay, EMA with decay 0.9999, and `bfloat16` mixed precision. Additional architectural and optimization details are provided in **Appendix A**.

Evaluation Protocol. For the main comparisons, we generate 50K images with a 50-step Heun sampler and classifier-free guidance scale 2.2 over the interval $[0.1, 1.0]$ [18]. We report FID [13] and IS [31] as the main evaluation metrics. FID is computed on 50K class-balanced samples, with 50 generated images for each of the 1000 ImageNet classes. For the ablation studies, we generate 10K images with the same sampling setup. For the auto-encoding path, we additionally report PSNR, rFID, and gFID when evaluating AE quality.

Table 1: Class-conditional ImageNet generation at 256×256 resolution. For **Models**, **Gen.** means generator, **AE** means autoencoder, **Dec.** means decoder, and **VFM** indicates the vision foundation model. For **Repr.**, **Pixel** means pixel diffusion, **Fixed** means fixed latent representation in training, and **Dynamic** means dynamically evolved representation in training. **Aux. Data** denotes external training data beyond ImageNet, and **Training FLOPs** report the generator-only training cost ($\times 10^{19}$).

	Method	Models	Repr.	Total Params (M)	Epochs	Aux. Data	Training FLOPs	w/ CFG	
								FID↓	IS↑
Two-stage	DiT-XL/2 [26]	Gen.+AE	Fixed	759	1400	[17]	45.4	2.27	278
	SiT-XL/2 [23]	Gen.+AE	Fixed	759	1400	[17]	45.4	2.06	270
	LightningDiT [40]	Gen.+AE+VFM	Fixed	745	800	-	19.1	1.35	295
	REPA-SiT [42]	Gen.+AE+VFM	Fixed	759	800	[17]	25.9	1.29	306
	DDT-XL/2 [37]	Gen.+AE+VFM	Fixed	759	400	[17]	19.2	1.26	311
	REPA-E (tuning) [20]	Gen.+AE+VFM	Fixed	759	800	[17]	57.8	1.12	303
	SVG-XL [32]	Gen.+AE+VFM	Fixed	758	1400	[33]	22.8	1.92	265
	RAE-DiT^{DH} [46]	Gen.+AE+VFM	Fixed	839	800	[25]	-	1.13	263
One-stage	REPA-E (scratch) [20]	Gen.+AE+VFM	Dynamic	759	80	-	5.78	1.67	-
	UNITE-XL [9]	Gen.+Dec.	Dynamic	763	240	-	12.0	1.75	310
	DSD [38]	Gen.+VFM	Dynamic	205	50	-	-	3.35	255
	ADM-U [8]	Gen.	Pixel	554	400	-	-	4.59	187
	RIN [14]	Gen.	Pixel	410	480	-	20.5	3.42	182
	PixNerd [36]	Gen.+VFM	Pixel	700	160	-	5.49	2.15	297
	PixelFlow [7]	Gen.	Pixel	677	320	-	239	1.98	282
	JiT-H/16 [21]	Gen.	Pixel	953	600	-	14.0	1.86	303
	LDM-is-AE (Ours)	Gen.	Dynamic	961	300	-	7.02	1.80	314

Training FLOPs ($\times 10^{19}$) report the generator-only forward training compute, measured as processed examples $\times$ forward-pass FLOPs, which **do not include the cost of training separated AE or VFM**.

4.2 Image Generation Results

Results at 256×256 Resolution. We compare LDM-is-AE with recent ImageNet generators in Tab. 1. The two-stage methods include DiT-XL/2 [26], SiT-XL/2 [23], LightningDiT [40], REPA-SiT [42], DDT-XL/2 [37], REPA-E (tuning) [20], SVG-XL [32], and RAE-DiT^{DH} [46]. The one-stage methods include REPA-E (scratch) [20], UNITE-XL [9], DSD [38], ADM-U [8], RIN [14], PixNerd [36], PixelFlow [7], and JiT-H/16 [21]. We see that two-stage methods generally achieve better FID than one-stage methods. However, the strongest two-stage methods all rely on external VFMs [25]. The two-stage methods without VFM, namely DiT-XL/2 and SiT-XL/2, only achieve FIDs of 2.27 and 2.06. This indicates that the two-stage methods benefit substantially from external VFM supervision, which needs additional training cost. Note that in Tab. 1, **Training FLOPs count only generator training, excluding separated AE training and vision foundation model (VFM) pretraining**. Even under this conservative accounting, one-stage methods remain substantially cheaper.

For one-stage methods, LDM-is-AE achieves the best IS over all methods, and the second-best FID among methods without VFM. It improves over PixelFlow, JiT-H/16, and attains IS 314 compared with 310 for UNITE-XL. Our FID (1.80) also outperforms PixNerd, which reports FID 2.15 despite using DINOv2 [25]. Relative to REPA-E (scratch), LDM-is-AE attains comparable quality under weaker assumptions, since REPA-E (scratch) relies on DINOv2 and a separate Gen.+AE pipeline, whereas LDM-is-AE learns the latent interface within a single generator trained from scratch.

The columns **Models** and **Repr.** in Table 1 clarify the key differences among methods. LDM-is-AE is the only method that has a *Dynamic* representation with a pure *Gen.*. Other dynamic-representation methods require additional *AE*, *Dec.* or *VFM*, whereas pixel-space methods operate directly on pixels and conventional latent-diffusion baselines use fixed latent interfaces. LDM-is-AE has 961M parameters, compared with 953M for JiT-H/16 and 839M for RAE-DiT^{DH}. Our generator training cost is lower than JiT-H/16 (7.02 vs. 14.0). Although UNITE-XL is trained for fewer epochs, it still incurs higher generator FLOPs (12.0), since it requires two full generator passes in each training iteration, whereas ours requires only one. REPA-E (scratch) and PixNerd report smaller generator-only FLOPs, but both depend on pre-trained DINOv2, which moves part of the representation-learning cost outside the tabulated budget. Overall, LDM-is-AE jointly learns a latent interface adapted to the diffusion process within a single-generator one-stage framework, achieving the best IS and highly competitive FID among one-stage methods at low generator training cost.



Figure 3: Class-conditional ImageNet samples generated by LDM-is-AE at 256×256 resolution.

Table 2: Class-conditional ImageNet generation at 512×512 resolution.

	Method	Models	Repr.	Total Params (M)	w/ CFG	
					FID↓	IS↑
Two-stage	DiT-XL/2 [26]	Gen.+AE	Fixed	759	3.04	241
	SiT-XL/2 [23]	Gen.+AE	Fixed	759	2.62	252
	REPA-SiT-XL/2 [42]	Gen.+AE+VFM	Fixed	759	2.08	275
One-stage	ADM-G [8]	Gen.	Pixel	559	7.72	173
	RIN [14]	Gen.	Pixel	320	3.95	216
	PixNerd-XL/16 [36]	Gen.+VFM	Pixel	700	2.84	246
	DeCo [24]	Gen.	Pixel	682	2.22	290
	JiT-H/32 [21]	Gen.	Pixel	956	1.94	309
	LDM-is-AE (Ours)	Gen.	Dynamic	961	**1.90**	**320**

Fig. 3 shows representative samples generated by LDM-is-AE. The samples cover diverse semantic categories and show coherent global composition, recognizable object structure, and plausible fine-scale texture. The beetle preserves a compact global silhouette, the bird shows stable part arrangement and clear foreground separation, and the cat retains plausible fur texture. The mushroom and ostrich further illustrate clean object boundaries and locally consistent details across different categories. These examples are consistent with the quantitative results and indicate that image-space alignment of the intermediate feature does not visibly degrade perceptual quality. Additional visual results are provided in **Appendix C**.

Results at 512×512 Resolution. We further train LDM-is-AE on ImageNet at 512×512 resolution. As shown in Tab. 2, it attains an FID of 1.90 and an IS of 320, the best among the compared methods. It improves over the strongest pixel-space baseline, JiT-H/32 [21] (1.94), and over the latent two-stage methods, suhc as REPA-SiT-XL/2 [42] (2.08) and SiT-XL/2 [26] (2.62), indicating that the our training framework also works at a higher resolution.

Text-to-Image Generation. LDM-is-AE can be further scaled to text-to-image generation. The detailed results can be found in Appendix B.

4.3 Latent Space Analysis

We compare the latent space of LDM-is-AE with SDVAE [29], REPA-E [20], and VAVAE [40]. PSNR, rFID, and gFID are all evaluated on ImageNet val-50K. PSNR measures instance-level fidelity; rFID measures the distribution gap between original and reconstructed images; gFID measures the distribution gap between reconstructed and natural images. Tab. 3 summarizes the quantitative comparison, and Fig. 4 shows the training-time evolution of the learned latent space.

Auto-encoding in LDM. Two-stage LDMs (even the end-to-end framework REPA-E [20]) optimize the encoder through reconstruction gradients and decode from clean latents. LDM-is-AE differs on both fronts: gradients from $\mathcal{L}_{\text{toimg}}$ stop at the DiT-E boundary (see Fig. 2 (a)), and DiT-D consumes $z_t = tz_1 + (1-t)z_0$, where fine-grained information is largely destroyed except near $t \to 1$. Despite

Table 3: Reconstruction performance on ImageNet val-50k. **Bold** indicates the best results.

Method	PSNR↑	gFID↓	rFID↓
SDVAE	25.94	3.415	0.6750
REPA-E	25.11	2.745	0.4980
VAVAE	26.59	2.566	**0.2650**
LDM-is-AE	**27.57**	**1.821**	0.7879

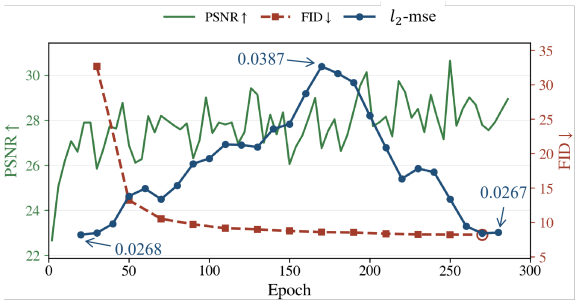
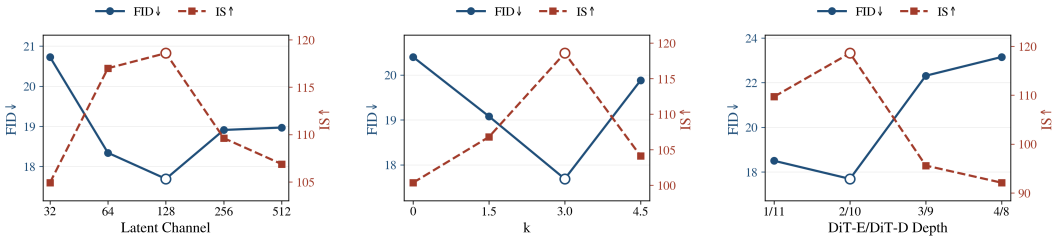


Figure 4: Training-time evolution of latent space.



(a) Latent channel dimension. (b) Auxiliary mixing exponent k. (c) DiT-E/DiT-D depth split.

Figure 5: Ablations on the three main design choices in LDM-is-AE.

these reconstruction-unfavorable conditions, imposing $\mathcal{L}_{\text{toimg}}$ activates the auto-encoding structure of the backbone: the clean path at $t = 1$ achieves the highest PSNR (27.57), compared with 26.59 for VAVAE, 25.94 for SDVAE, and 25.11 for REPA-E. Additional reconstruction examples and latent space visualizations are provided in **Appendix D**.

Diffusion-native Latent Space. While producing competitive rFID and gFID, the rFID and gFID of those tokenizers exhibit opposite trends across methods. VAVAE attains the lowest rFID (0.2650) but a higher gFID (2.566). REPA-E, although also end-to-end, stops the diffusion gradient at the latent interface, so its tokenizer remains primarily shaped by reconstruction, yielding competitive rFID (0.4980) but a higher gFID (2.745). By contrast, LDM-is-AE attains a higher rFID (0.7879) but a substantially lower gFID (1.821), improving over REPA-E (2.745), VAVAE (2.566), and SDVAE (3.415) on gFID. This rFID-gFID divergence suggests that our latent space is not optimized for input-distribution fidelity, but is instead driven toward the natural image distribution through the denoising objective, which is suggestive of a diffusion-native representation. One interesting point is that the generation FID (1.80), the reconstruction gFID (1.82), and the FID of a random 50K ImageNet training subset (1.74) are numerically very close. This is consistent with the **diffusion-native latent space interpretation** and indicates competitive generation performance without additional biased supervision from a pre-trained VFM.

Latent Evolution During Training. We further examine how the latent space evolves during training. As shown in Fig. 4, PSNR saturates early (about 20 epochs), which is consistent with the residual design of DiT-E: once the image-aligned base latent is established, the encoder mainly needs to predict a small correction for reconstruction. In contrast, FID (for 5K images) improves rapidly at early epochs and continues to decrease throughout training. We also measure the ℓ_2-distance between the latent of each image and the latent obtained 10 epochs earlier, averaged over 100 validation images. This latent drift first increases, then decreases, and finally enters a plateau, which occurs substantially later than PSNR saturation (about 260 epochs vs. 20 epochs). The late-stage evolution of the latent space can be naturally explained by continued adaptation to the denoising objective, which further supports the diffusion-native interpretation.

4.4 Ablation Studies

We investigate the key design choices of LDM-is-AE using a JiT-B/16 backbone trained for 100 epochs, with evaluation conducted on 10K generated samples, unless stated otherwise.

Latent Channel. We vary the latent channel to show how compression strength influences generation performance. As shown in Fig. 5(a), performance follows an inverted-U trend as the latent width increases: moving from 32 to 64 and 128 channels improves FID from 20.73 to 18.34 and 17.69,

Table 4: Ablation on architecture backbone and LPIPS supervision. All models are evaluated *without* classifier-free guidance. **Bold** marks our method and the best result in each column.

Method	FID↓	IS↑	sFID↓	Prec.↑	Recall↑
SiT-B/2	45.19	37.40	**26.16**	0.471	0.613
+ **LDM-is-AE**	**41.68**	**40.70**	27.34	**0.485**	**0.626**
JiT-B/16	30.27	54.80	22.01	0.5249	0.6753
+ LPIPS	27.57	61.09	21.71	**0.5623**	0.6662
+ **LDM-is-AE**	**25.95**	**67.45**	**20.38**	0.5545	**0.6836**

while IS increases from 105 to 117 and 119. Further increasing the latent width degrades FID and IS. A setting of 128 channels gives the best generation quality.

k in Auxiliary Feature Mixing. The exponent k in Eq. 9 controls how long the image-aligned feature F stays dominant over the auxiliary feature F': a smaller k keeps F dominant over more timesteps, whereas a larger k confines it closer to the clean endpoint $t = 1$ and leaves more freedom to F' at noisy steps. As shown in Fig. 5(b), both too small and too large exponents degrade FID and IS, and $k = 3.0$ provides the best balance for generation quality.

DiT-E/DiT-D Depth. We vary the number of attention blocks assigned to DiT-E and DiT-D while keeping the total backbone depth fixed. As shown in Fig. 5(c), the 2/10 split (2 for DiT-E while 10 for DiT-D) achieves the best overall performance, with FID 17.69 and IS 119, indicating that the two parts do not make equal demands on capacity.

Backbone. We further apply the same recipe to SiT-B/2 to verify the robustness of our framework to the backbone architecture. As shown in Tab. 4, LDM-is-AE lowers FID from 45.19 to 41.68 and raises IS from 37.40 to 40.70, together with better Prec. (0.485 vs. 0.471) and Recall (0.626 vs. 0.613). The sFID is slightly higher (27.34 vs. 26.16). The same trend holds on this different backbone architecture, indicating that our one-stage training framework is not tied to a particular transformer design and works across backbone architectures.

LPIPS Supervision. To separate the effect of LPIPS supervision from that of the proposed latent interface, we compare vanilla JiT-B/16, JiT-B/16 trained with an additional LPIPS loss, and LDM-is-AE on the same backbone, all trained for 200 epochs. As shown in Tab. 4, LPIPS supervision alone improves FID from 30.27 to 27.57 and IS from 54.80 to 61.09 without CFG strategy, yet LDM-is-AE improves them further, to 25.95 and 67.45. Prec. is on par with the LPIPS-only variant (0.5545 vs. 0.5623), while FID, IS, sFID, and Recall all improve. The gains of exposing an image-aligned interface inside the diffusion backbone are therefore orthogonal to those of LPIPS supervision alone.

We provide analysis on the sensitivity to the loss weights $w(t)$ and w_{lpips} in Appendix E.

5 Conclusion

We presented **LDM-is-AE**, a one-stage end-to-end latent diffusion framework by revealing that the diffusion backbone followed a decoding–encoding structure. By aligning the intermediate DiT feature with the image domain, we turned the implicit latent-to-feature-to-latent transformation into an explicit latent-to-image-to-latent path, thereby integrating latent representation learning into diffusion training without a separately pre-trained tokenizer. LDM-is-AE demonstrated competitive class-conditional generation quality, while maintaining a simpler and cheaper training pipeline than conventional two-stage latent diffusion. By jointly learning the latent representation and denoising dynamics, we proved that the DiT can function as both a denoiser and an auto-encoder for learning diffusion-native latent spaces. The explicit intermediate representation may also support applications such as controllable generation, interactive editing, and analysis of diffusion dynamics.

Limitations. One limitation of LDM-is-AE lies in its residual latent prediction, which may constrain exploration of the latent space. In future work, we will investigate stronger optimization strategies and the intermediate image-aligned interfaces to further improve overall performance.

Acknowledgments and Disclosure of Funding

Funding disclosure. This research is supported by the PolyU-OPPO Joint Innovative Research Center.

Competing interests. The authors declare no competing interests.

References

[1] Roman Bachmann, Jesse Allardice, David Mizrahi, Enrico Fini, Oğuzhan Fatih Kar, Elmira Amirloo, Alaaeldin El-Nouby, Amir Zamir, and Afshin Dehghan. Flextok: Resampling images into 1d token sequences of flexible length. *arXiv preprint arXiv:2502.13967*, 2025.

[2] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22669–22679, 2023.

[3] Hao Chen, Yujin Han, Fangyi Chen, Xiang Li, Yidong Wang, Jindong Wang, Ze Wang, Zicheng Liu, Difan Zou, and Bhiksha Raj. Masked autoencoders are effective tokenizers for diffusion models. *arXiv preprint arXiv:2502.03444*, 2025.

[4] Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. Blip3-o: A family of fully open unified multimodal models-architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

[5] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, et al. Pixart-\alpha: Fast training of diffusion transformer for photorealistic text-to-image synthesis. *arXiv preprint arXiv:2310.00426*, 2023.

[6] Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, Yao Lu, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. *arXiv preprint arXiv:2410.10733*, 2024.

[7] Shoufa Chen, Chongjian Ge, Shilong Zhang, Peize Sun, and Ping Luo. Pixelflow: Pixel-space generative models with flow. *arXiv preprint arXiv:2504.07963*, 2025.

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

[9] Shivam Duggal, Xingjian Bai, Zongze Wu, Richard Zhang, Eli Shechtman, Antonio Torralba, Phillip Isola, and William T Freeman. End-to-end training for unified tokenization and latent denoising. *arXiv preprint arXiv:2603.22283*, 2026.

[10] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.

[11] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.

[12] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Masked diffusion transformer is a strong image synthesizer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 23164–23173, 2023.

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

[14] Allan Jabri, David J Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. In *Proceedings of the 40th International Conference on Machine Learning*, pages 14569–14589, 2023.

[15] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.

[16] Theodoros Kouzelis, Ioannis Kakogeorgiou, Spyros Gidaris, and Nikos Komodakis. EQ-VAE: Equivariance regularized latent space for improved generative image modeling. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=UWhW5YYLo6`.

[17] Ivan Krasin, Tom Duerig, Neil Alldrin, Andreas Veit, Sami Abu-El-Haija, Serge Belongie, David Cai, Zheyun Feng, Vittorio Ferrari, Victor Gomes, Abhinav Gupta, Dhyanesh Narayanan, Chen Sun, Gal Chechik, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, 2016.

[18] Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. *Advances in Neural Information Processing Systems*, 37:122458–122483, 2024.

[19] Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024.

[20] Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. Repa-e: Unlocking vae for end-to-end tuning of latent diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18262–18272, 2025.

[21] Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

[22] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

[23] Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pages 23–40. Springer, 2024.

[24] Zehong Ma, Longhui Wei, Shuai Wang, Shiliang Zhang, and Qi Tian. DeCo: Frequency-decoupled pixel diffusion for end-to-end image generation. *arXiv preprint arXiv:2511.19365*, 2025.

[25] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[26] 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.

[27] 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. *arXiv preprint arXiv:2307.01952*, 2023.

[28] Qi Qin, Le Zhuo, Yi Xin, Ruoyi Du, Zhen Li, Bin Fu, Yiting Lu, Xinyue Li, Dongyang Liu, Xiangyang Zhu, et al. Lumina-image 2.0: A unified and efficient image generative framework. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20031–20042, 2025.

[29] 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.

[30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[31] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

[32] Minglei Shi, Haolin Wang, Wenzhao Zheng, Ziyang Yuan, Xiaoshi Wu, Xintao Wang, Pengfei Wan, Jie Zhou, and Jiwen Lu. Latent diffusion model without variational autoencoder, 2025. URL `https://arxiv.org/abs/2510.15301`.

[33] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

[34] Lingchen Sun, Rongyuan Wu, Zhengqiang Zhang, Ruibin Li, Yujing Sun, Shuaizheng Liu, and Lei Zhang. Self-transcendence: Is external feature guidance indispensable for accelerating diffusion transformer training? *arXiv preprint arXiv:2601.07773*, 2026.

[35] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[36] Shuai Wang, Ziteng Gao, Chenhui Zhu, Weilin Huang, and Limin Wang. Pixnerd: Pixel neural field diffusion. *arXiv preprint arXiv:2507.23268*, 2025.

[37] Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. *arXiv preprint arXiv:2504.05741*, 2025.

[38] Xiyuan Wang and Muhan Zhang. Diffusion as self-distillation: End-to-end latent diffusion in one model. *arXiv preprint arXiv:2511.14716*, 2025.

[39] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[40] Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. *arXiv preprint arXiv:2501.01423*, 2025.

[41] Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. *Advances in Neural Information Processing Systems*, 37:128940–128966, 2024.

[42] 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.

[43] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[44] Zhengqiang ZHANG, Ruihuang Li, and Lei Zhang. Frecas: Efficient higher-resolution image generation via frequency-aware cascaded sampling. In *The Thirteenth International Conference on Learning Representations*.

[45] Zhengqiang Zhang, Rongyuan Wu, Lingchen Sun, and Lei Zhang. Gpstoken: Gaussian parameterized spatially-adaptive tokenization for image representation and generation. *Advances in neural information processing systems*, 2025.

[46] Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders, 2025. URL `https://arxiv.org/abs/2510.11690`.

Table 5: Text-to-image generation on the GenEval benchmark.

Method	Sin.Obj.	Two.Obj	Counting	Colors	Pos	Color.Attr.	Overall↑
PixArt-α [5]	0.98	0.50	0.44	0.80	0.08	0.07	0.48
SD3 [11]	0.98	0.84	0.66	0.74	0.40	0.43	0.68
PixNerd [36]	0.97	0.86	0.44	0.83	0.71	0.53	0.73
DeCo [24]	1.00	0.92	0.72	0.91	0.80	0.79	0.86
LDM-is-AE	0.99	0.95	0.75	0.93	0.62	0.74	0.83

Appendix

This appendix contains the following parts:

A Experimental setup (referring to Sec. 4.1 of the main paper);

B Scalability to text-to-image generation (referring to Sec. 4.2 of the main paper);

C Additional visual results for image generation (referring to Sec. 4.2 of the main paper);

D Reconstruction examples and latent visualizations (referring to Sec. 4.3 of the main paper);

E Sensitivity to loss weights (referring to Sec. 4.4 of the main paper).

A Experimental Setup

This section supplements Sec. 4.1 of the main paper with the detailed training configuration used in our experiments. For the main experiments, we use a JiT-H/16 backbone with 30 DiT-D layers and 2 DiT-E layers, trained for 300 epochs. For the ablation studies, we use JiT-B/16 with 10 DiT-D layers and 2 DiT-E layers, trained for 100 epochs unless otherwise stated. We use a latent channel dimension of 128, a pixel patch size of 16, and a noise scale of 1.0. The learning rates for DiT-D and DiT-E are 2×10^{-4} and 2×10^{-7}, respectively. We use the v-loss in Eq. 4 for diffusion and set $w(t) = 1/(1-t)^2$ in Eq. 7 to match the weighting induced by the v-loss. We set $w_{\text{toimg}} = 1$ in Eq. 10, $w_{\text{lpips}} = 1$ in Eq. 7, and $\gamma(t) = t^3$ in Eq. 9. The encoder and the decoder follow decoupled learning-rate schedules. The encoder learning rate is decayed by a factor of 3 for every 50 epochs. From epoch 200 onward, the encoder learning rate is set to zero and the decoder learning rate is reduced to $0.1\times$. Only the samples with $t \geq 0.5$ update the encoder.

B Scalability to Text-to-Image Generation

Beyond class-conditional ImageNet generation, LDM-is-AE can also be scaled to text-to-image generation. Our training pipeline largely follows the DeCo [24] protocol. We initialize LDM-is-AE from our 512×512 class-conditional model, replace the class conditioning with a Qwen3-1.7B [39] text encoder, and train on the BLIP3o dataset [4] at 512×512 with an effective batch size of 1024. Training first adapts the text branch with a frozen backbone and then fine-tunes the full model for about 100k iterations. We then evaluate on the GenEval benchmark. As shown in Tab. 5, LDM-is-AE reaches an overall score of 0.83, substantially outperforming PixArt-α (0.48) [5], SD3 (0.68), and PixNerd (0.73) [36], and is competitive with DeCo (0.86) [24]. It surpasses DeCo on Two.Obj., Counting, and Colors, while the remaining gap lies mainly in Pos. and Color. attributions, which are closely tied to pixel-space modeling and thus less favorable to our latent diffusion model.

C Additional Visual Results on Image Generation

This section provides additional qualitative results for 256×256 image generation in Fig. 6. The samples cover diverse semantic categories, including animals, plants, food, natural scenes, and man-made objects, and further illustrate the visual quality and category coverage of LDM-is-AE.



Figure 6: Additional class-conditional ImageNet samples generated by LDM-is-AE. All images are produced at a resolution of 256×256.

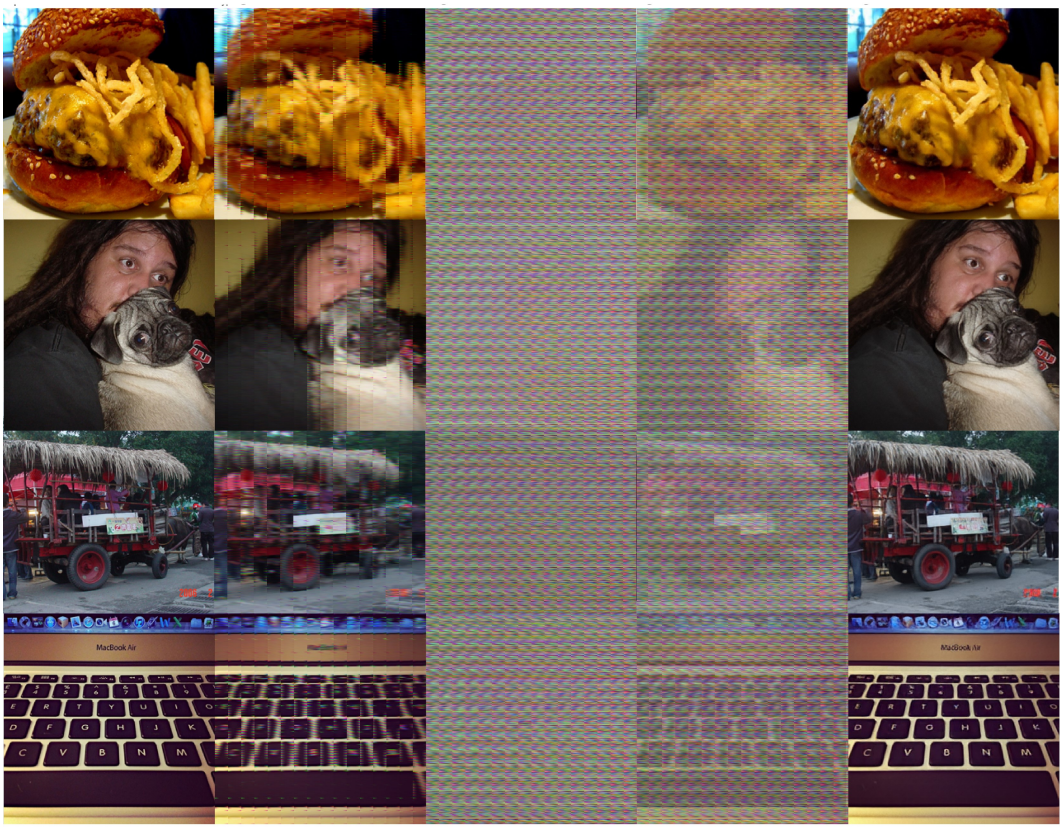


Figure 7: Reconstruction examples and latent visualizations for the auto-encoding path.

D Reconstruction Examples and Latent Visualizations

This section supplements Sec. 4.3 of the main paper with reconstruction examples and visualizations of the learned latent representation.

In Sec. 4.3, we report reconstruction metrics for the auto-encoding path in LDM-is-AE. Here, we further visualize the reconstructed images and the intermediate latent z_1 in Fig. 7. For latent-space visualization, we apply one-dimensional interpolation along the channel dimension to match the shape of x_u, and then map the result to RGB space through PixelShuffle.

The *base of* z_1 denotes the channel-interpolated image-aligned signal that provides the skip-path output to DiT-E. The *residual of* z_1 denotes the correction predicted by DiT-E on top of this base signal, and z_1 denotes the resulting latent representation. Compared with the pixel-space image, z_1 preserves the global structure while discarding much of the fine-grained appearance information. By contrast, the reconstructed image remains natural and closely matches the input image in both overall structure and semantic details.

E Sensitivity to Loss Weights

Eq. 7 introduces $w(t)$ to control the overall strength of the image-space supervision and w_{lpips} to balance the LPIPS and MSE terms. We scale both weights by 0.3, 1.0 (default), and 3.0. As shown in Tab. 6, the default setting $(1.0, 1.0)$ obtains the best IS (119) and a near-best FID (17.69). FID is robust to increasing either weight (18.28 for $w_{\text{lpips}} = 3.0$ and 17.94 for $w(t) = 3.0$), whereas decreasing $w(t)$ to 0.3 degrades FID to 26.59 and IS to 68, indicating that the image-space supervision must be applied at full strength.

Table 6: Sensitivity to the loss weights $w(t)$ and w_{lpips} of Eq. 7. All models are evaluated without classifier-free guidance.

Scales $(w(t), w_{\text{lpips}})$	FID↓	IS↑
$(1.0, 1.0)$	17.69	119
$(1.0, 0.3)$	19.67	108
$(1.0, 3.0)$	18.28	96
$(0.3, 1.0)$	26.59	68
$(3.0, 1.0)$	17.94	91

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: For example, in the "Experiments" section, we evaluate our method on image generation, latent-space analysis, and ablation studies to comprehensively validate the main claims stated in the abstract and introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations in "Conclusion" section.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: We do not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide detailed experimental settings in "Experiments" section and supplementary material.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We will release the codes and models.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: We provide detailed experimental settings in "Experiments" section.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: We follow same experimental settings and content of previous works, which do not include such experiments. We will take this into consideration in future works.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

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

8. **Experiments compute resources**

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

   Answer: [No]

   Justification: We report the training FLOPs in the paper, but do not yet provide the full hardware and execution-time details required for this item.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We follow the guidelines.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [No]

    Justification: We do not include such discussion in the current submission. We will take this into consideration in future works.

    Guidelines:

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

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

11. **Safeguards**

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

    Answer: [No]

    Justification: We follow previous works, which do not include such safeguards. The safeguards are valuable. We will take this into consideration in future works.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cited all the origin papers in the paper.

    Guidelines:

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

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

13. **New assets**

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

    Answer: [Yes]

    Justification: We will release the detailed document of codes and models.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [N/A]

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

    Guidelines:

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

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [N/A]

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

    Guidelines:

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

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

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.