Sparse Image Synthesis via Joint Latent and RoI Flow

Ziteng Gao Jay Zhangjie Wu Mike Zheng Shou*

Show Lab, National University of Singapore

Abstract

Natural images often exhibit underlying sparse structures, with information density varying significantly across different spatial locations. However, most generative models rely on dense grid-based pixels or latents, neglecting this inherent sparsity. In this paper, we explore modeling visual generation paradigm via sparse non-grid latent representations. Specifically, we design a sparse autoencoder that represents an image as a small number of latents with their positional properties (i.e., regions of interest, RoIs) with high reconstruction quality. We then explore training flow-matching transformers jointly on non-grid latents and RoI values. To the best knowledge, we are the first to address spatial sparsity using RoIs in generative process. Experimental results show that our sparse flow-based transformers have competitive performance compared with dense grid-based counterparts with significantly reduced lower compute, and reaches a competitive 2.76 FID with just 64 latents on class-conditional ImageNet 256×256 generation.

1 Introduction

Deep visual generative models have advanced significantly in recent years, achieving impressive visual quality on image [1, 2], video [3], and 3D domains [4]. Current visual generation pipelines typically start by encoding raw data (e.g., images) into compact latent representations with autoencoders, and then use diffusion, masking modeling, or autoregressive methods to generate such latents, as this pipeline exemplified by latent diffusion [1]. Flagship text-to-image models, e.g., Stable Diffusion [5] and FLUX [6], follow this line and compress spatial dimensions at typically $8\times$ factor, significantly lowering computational costs and modeling complexity in generative training.

Though being a core component of visual generation, autoencoders conventionally assume a grid-based space of latent structures with the uniform information density. However, natural images often exhibit highly non-uniform information density and require adaptive computation across spatial locations [7, 8]. For example, in a landscape image, the sky background occupies numerous pixels while being worth fewer latent units to reconstruct and generate. In contrast, intricate foreground objects may require more latents to capture their details. Existing visual generation pipelines fail to address this point, as they rely on dense uniform grid-based latent structures and cannot adaptively allocate more computation to intricate foregrounds.

This paper aims to study this point. First, we propose a *sparse visual autoencoder* that learns to compress an image into a set of sparse non-grid latents along with their positional property, i.e., region of interests (RoIs), and then recover image pixels from them. The RoIs explicitly characterize the spatial locations of the latents in bounding box formulation, and can be learned jointly with latents in an end-to-end manner by the plain reconstruction loss. The resulting sparse visual autoencoder reaches high compression rates by prioritizing latents to detailed regions while maintaining high reconstruction fidelity. Then, we design *sparse flow-based transformers* to generate latents and RoIs by modeling the joint flow of them with the velocity prediction in the denoising process [9, 10]. At

*The corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

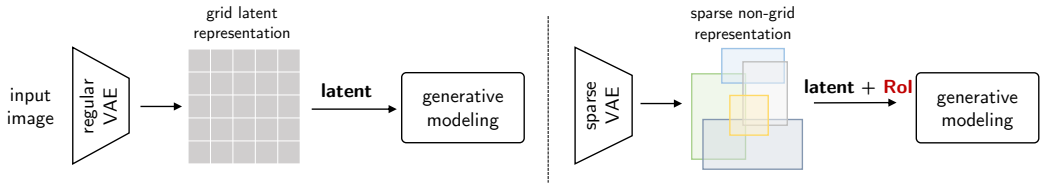

Figure 1: **Left:** conventional autoencoders encode pixels into latent grid representations. **Right:** our method encodes them into fewer non-grid latents with region of interests (RoIs).

every timestep, our model learns to estimate *both* latent and RoI velocity from initial noise to target samples. Divergent from prior grid-based latent approaches, our method dynamically adjusts latent spatial positions during sampling via ordinary differential equations (ODEs) at inference, allowing adaptive refinement of both content and spatial focus.

We show the feasibility of representing and generating images with sparse latents and RoIs on the challenge ImageNet benchmark. Our proposed sparse flow autoencoder, SF-VAE, can represent 256×256 images with just 64 latents with 0.70 reconstruction FID, or even down to 32 latents with 1.70 rFID. Then, the presented sparse flow-based transformers, SF-SiTs, have competitive performance on par with diffusion/flow-based grid-based transformers [11, 10]. The largest SF-SiT, XL variant, can reach 2.76 FID with classifier-free guidance [12] on the class-conditional ImageNet generation benchmark with just 64 latents.

2 Related Work

Diffusion models. In recent years, generative models has been marked as a breakthrough in the field of visual synthesis [1, 13, 14]. Commercial systems like DALL-E [2] or FLUX [6] are typically rooted in denoising diffusion architectures. The seminal work, denoising diffusion probabilistic models [15], take the image generative process as a gradual denoising trajectory, iteratively refining pure noise into target images. Building on this foundation, subsequent advancements further accelerated and refined diffusion-based generation. Improved variants including [16, 17, 9] investigate the training and sampling trajectories, enabling high-quality results with fewer sampling steps. Latent diffusion models [1] democratize the high resolution image synthesis by operating in a compressed latent space with reduced computational costs. The following up work, including diffusion/flow-matching transformers [18, 11, 10], also follows this convention to speed up training. Although training diffusion models directly on raw pixels is technically feasible [19, 20], the preference for latent space modeling stems from practical challenges: raw pixel data often contains high-frequency details and perceptually complex patterns that are computationally intensive and difficult for diffusion processes to model effectively.

Latent space for diffusion models. The compact latent space is crucial for diffusion models to achieve high-quality image synthesis. Latent diffusion models [1] propose to train an autoencoder to map raw pixels to a latent space first, where the latent space is typically $8\times$ spatially downsampled and comes with 4 channels, reaching a compression rate of 48. The follow up work on autoencoders, including [5, 21, 22], mainly investigate the channel number and shows that increasing channel number can improve the quality of diffusion samples via larger transformer models. Recently proposed deep compression autoencoders (DC-AE) [23] compress the latent space at more aggressive spatial downsampling rates, e.g., 32 or 64, further reducing the training cost of diffusion models.

However, there is a lack of exploration and discussion on the structure of latents for diffusion models. Most autoencoders for diffusion models encode pixels into dense 2D grid-based latents and ignore the underlying non-uniform and sparse structures in natural images, where a background region in an image might be worth less latents than foregrounds. Here in this paper, we study this sparsity as well as visual non-uniformity explicitly with region of interests (RoIs) along with latents for diffusion models, following the sparse visual generation research line [8].

3 Methods

Our image synthesis pipeline follows the common practice of latent diffusion models: an autoencoder first that compresses images into a set of latents and RoIs, followed by a generative model that takes noised latents and RoIs as input and predicts diffusion targets. We first describe the design of our *sparse flow autoencoders*, and then introduce *sparse flow-based generative transformers* for modeling joint flow.

3.1 Sparse Flow Autoencoders

Given an image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, conventional autoencoders encode it into grid-based latent representations and decode latents back into pixels by the encoder $\mathcal{E}$ and decoder $\mathcal{D}$:

$$\mathbf{z} = \mathcal{E}(\mathbf{I}) \in \mathbb{R}^{H/f \times W/f \times d}, \tag{1}$$

$$\hat{\mathbf{I}} = \mathcal{D}(\mathbf{z}) \in \mathbb{R}^{H \times W \times 3}, \tag{2}$$

where f is the downsampling factor, typically 8 in practice, and d is the latent dimension. The training for $\mathcal{E}$ and $\mathcal{D}$ is done by minimizing the reconstruction loss $\ell_{\text{rec}}(\mathbf{I}, \hat{\mathbf{I}})$. Variational autoencoders also impose a Kulback-Leibler divergence loss ℓ_{KL} on $\mathbf{z}$ to regularize the latent distribution [24].

Latent and RoI representation. Different from grid-based latent representations, we propose sparse flow variational autoencoders, SF-VAE, to use a sparse set of latents $\mathbf{z} \in \mathbb{R}^{N \times d}$ with their corresponding regions of interests (RoIs) $\mathbf{r} \in \mathbb{R}^{N \times 4}$ to represent an image. The latent space of SF-VAE is simply structured as one flattened dimension space, and the spatial property of latents is characterized in RoIs. The RoIs are represented as bounding boxes in the format of (x, y, h, w), where (x, y) and (h, w) are center points and height and width. The encoder $\mathcal{E}$ now outputs both latents and RoIs, and the decoder $\mathcal{D}$ takes both latents and RoIs as input to reconstruct raw pixels:

$$(\mathbf{z}, \mathbf{r}) = \mathcal{E}(\mathbf{I}), \qquad \hat{\mathbf{I}} = \mathcal{D}(\mathbf{z}, \mathbf{r}). \tag{3}$$

By eliminating the grid-based spatial prior, the number of latents N can be decoupled from image pixels $H \times W$ and can be arbitrarily chosen and further greatly reduced according to our experiments.

Encoding pixels into latents and RoIs. Typical object detectors [25, 26] can encode image pixels into latent representations and RoIs. However, they often lack emphasis on backgrounds and need bounding box annotations as individual supervision. Here, we resort to the SparseFormer architecture [27] to build our encoder $\mathcal{E}$ from scratch, which can encode image pixels into latents and RoIs in *an end-to-end manner without bounding box supervision*. SparseFormer takes early image features $\tilde{\mathbf{I}}$ as input and gradually refines latents $\mathbf{z}$ and RoIs $\mathbf{r}$ via local image features within RoIs $\mathbf{r}$ by several SparseFormer transformer layers, where refinement on $\mathbf{z}$ and $\mathbf{r}$ are both differentiable:

$$(\mathbf{z}^t, \mathbf{r}^t) = \textsc{SparseFormerLayer}_t(\tilde{\mathbf{I}}, \mathbf{z}^{t-1}, \mathbf{r}^{t-1}), \tag{4}$$

where $\mathbf{z}^0$ and $\mathbf{r}^0$ are parameters of the model. The RoIs $\mathbf{r}$ are updated using delta-formulation [25] on (x, y, h, w).

Decoding latents and RoIs back into pixels. To learn the distribution of latents and RoIs given an image by just pixel reconstruction, we need to decode them back into raw pixels in *a fully differentiable way*, that is, the reconstruction loss needs to be both differentiable to latents and RoIs. Unfortunately, to our best knowledge, mapping latents and RoIs back into raw pixels in a differentiable way has not been deeply investigated in previous works. We have first tried a simple idea, RoI-aware cross attention, but they struggle to recover high frequency details even in a long training schedule, and cannot learn compact latent RoIs with pixel reconstruction.

Inspired by advance in neural rendering [28, 29], we design our decoder $\mathcal{D}$ with the neural field approach and divide-and-conquer strategy. Specifically, we consider a latent and its RoI indexed by i as a neural field function $\mathcal{F}_{(\mathbf{z}_i, \mathbf{r}_i)} : \mathbb{R}^2 \to \mathbb{R}^3$ whose input is the pixel coordinate and output is the RGB tuple. In other words, a latent and its RoI can be decoded into a pixel image individually. Considering rich high frequency details in natural images [30], we design a neural field based on cosine transform bases similar to 2D discrete cosine transform (DCT) [31] but in a continuous

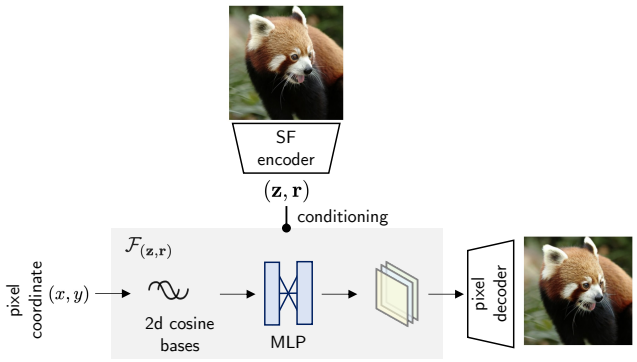

Figure 2: **Structure of our SF-VAE and neural field-based decoder**. For a latent tuple $(\mathbf{z}_i, \mathbf{r}_i)$, the neural field function $\mathcal{F}_{(\mathbf{z}_i,\mathbf{r}_i)}$ first transforms pixel coordinates into 2D cosine bases conditioned on $\mathbf{r}_i$ and then feed these bases into an MLP whose parameters are conditioned on $\mathbf{z}_i$ to get downsampled intermediate features. The pixel decoder then decode these features into raw image pixels. For clarity, we show only a single latent case.

coordinate space. Given a raw pixel's coordinate (x, y) in the image, we first compute relative coordinate (x', y') with regard to the RoI $\mathbf{r}_i$ for a single pixel output:

$$x' = (x - x_i)/w_i, \qquad y' = (y - y_i)/h_i, \tag{5}$$

then compute our customized cosine transform bases $\mathbf{X} \in \mathbb{R}^C$:

$$\mathbf{X}_c = \cos\left[\mathbf{f}_y(c)(y' + 0.5)\pi\right] \cos\left[\mathbf{f}_x(c)(x' + 0.5)\pi\right], \tag{6}$$

where the number of bases C needs to be a squared number, and $\mathbf{f}_y(c)$ and $\mathbf{f}_x(c)$ are the frequency of the c-th channel in y and x directions, linearly increasing from 0 to $\sqrt{C} - 1$.

We use a two-layered MLP to transform these cosine bases into final output values, where MLP's weights are conditioned by the latent $\mathbf{z}_i$ through a shared feed forward layer. Note that all these computation can be done with matrix multiplication efficiently. However, we find that directly recovering raw RGB pixels from $\mathcal{F}_{(\mathbf{z}_i,\mathbf{r}_i)}$ is not memory friendly despite being efficient, since we need to "trace" 65536 pixels with their bases and MLP activations for a single latent in a 256×256 image, which is unrealistic. Therefore, we retarget the neural field to output a downsampled feature map $\mathbf{I}' \in \mathbb{R}^{H/8 \times W/8 \times C}$, use the softmax function to blend different feature maps produced by different latents, and use a upsampling decoder, $\mathcal{D}_{\text{pix}}$, to upsample $\mathbf{I}'$ to the final image $\hat{\mathbf{I}}$ of the input size.

Overall autoencoder architecture. In the resulting SF-VAE, a latent does not need to correspond to a fixed spatial region across different images, and we can decouple the number of latents N from the image size and further reduce N. Our default number of latents and RoIs is $N = 64$ for a 256×256 image, where the latent dimension d is 32, reaching a high $96\times$ compression rate[2]. We train SF-VAE in an end-to-end manner using regular VAE loss with a minor difference: we only apply the KL loss to latent variables $\mathbf{z}$ since RoIs $\mathbf{r}$ do not necessarily follow a standard Gaussian distribution. We also design our SF-VAE to be as parameter lightweight as possible. Visualizations in Figure 3 show that SF-VAE can learn semantic and compact latent RoIs and achieve high reconstruction quality, allowing latents to focus on intricate foreground objects.

3.2 Sparse Flow Transformers

Now we have compact latent space defined by latent variables $\mathbf{z}$ and RoIs $\mathbf{r}$ for an image, we describe study *how to generate them jointly*. Here we first describe the flow matching framework [9, 32] for continuous data modeling, and then delve into our flow formulation and the design of our flow-based generative transformers.

Flow matching. Assume we want to model a target data distribution $p(\mathbf{x})$, flow matching formulates a process starting from a sample drawn from the starting distribution, typically $\mathbf{x}_0 \sim \mathcal{N}(0, 1)$, to a

[2]Compression rate is ~ 85 if a RoI also counted four values.

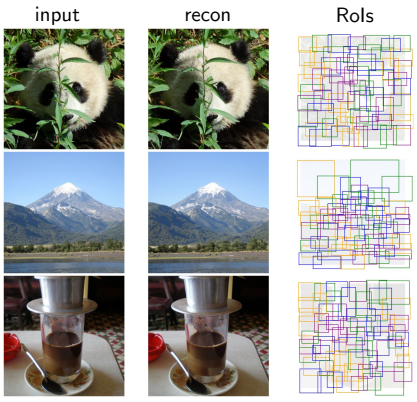

Figure 3: **Visualizations of SF-VAE**, and learned latent RoIs ($N = 64$). Note different colors are only used for distinguishing different RoIs.

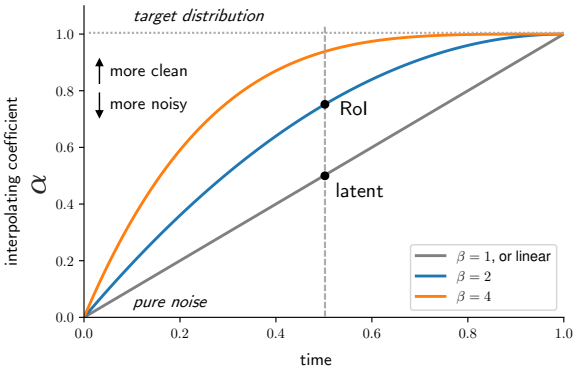

Figure 4: **Asynchronous interpolating schedule** for latents and RoIs. RoIs approach target RoIs faster than latents as time t increases using $\beta = 2$ polynomial schedule.

sample from the target distribution $\mathbf{x}_1 \sim p(\mathbf{x})$ by continuous time process:

$$\mathbf{x}_t = \sigma_t \mathbf{x}_0 + \alpha_t \mathbf{x}_1, \tag{7}$$

where $t \in [0, 1]$, σ_t and α_t are interpolating coefficients characterizing an interpolating flow. The flow matching requires $\sigma_0 = 1$, $\alpha_0 = 0$, $\sigma_1 = 0$, and $\alpha_1 = 1$. A popular choice is the linear interpolation, $\sigma_t = 1 - t$ and $\alpha_t = t$. The velocity, or the derivation on $\mathbf{x}_t$ in this process is

$$\mathbf{v}_t = \frac{d\mathbf{x}_t}{dt} = \dot{\sigma}_t \mathbf{x}_0 + \dot{\alpha}_t \mathbf{x}_1. \tag{8}$$

The model $\mathcal{F}$ is trained by minimizing the L_2 loss between the predicted velocity and the true velocity over sampled points and sampled paths:

$$\mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1} \|\mathcal{F}(t, \mathbf{x}_t) - \mathbf{v}_t\|^2. \tag{9}$$

During inference, flow matching draws an initial sample $\mathbf{x}_0 \sim \mathcal{N}(0, 1)$ and solve the oridinary differential equation (ODE) to get the target sample $\mathbf{x}_1$:

$$\frac{d\mathbf{x}_t}{dt} = \mathcal{F}(t, \mathbf{x}_t). \tag{10}$$

Flow matching eliminates the noise introduction in the sampling stage [15] and can model most arbitrary distribution, which is suitable for our RoI distribution modeling. Recent SiT models [10] also shows flow matching as the prediction target with transformers [33] can surpass ones with the diffusion target.

Joint latent and RoI flow. Different from SiT models only to model grid-based latents, we need to model both the joint distribution of latents $\mathbf{z}$ with their RoIs $\mathbf{r}$ by SF-VAE in our flow-based generative transformers. The latents $\mathbf{z}$ in SF-VAE are not structured as a grid, and their positional properties are encoded in RoIs $\mathbf{r}$. Here, we keep most of the transformer architecture in SiT unchanged but retarget it to take both latents and RoIs as input and predict the velocity of both them:

$$(\hat{\mathbf{v}}_{z,t}, \hat{\mathbf{v}}_{r,t}) = \mathcal{F}(t, \mathbf{z}_t, \mathbf{r}_t), \tag{11}$$

where $\mathbf{z}_t$ and $\mathbf{r}_t$ are interploated latents and RoIs following Equ 7 at time t, $\hat{\mathbf{v}}_{z,t}$ and $\hat{\mathbf{v}}_{r,t}$ are estimated velocity of latents and RoIs. We name it as SF-SiT for modeling joint latent and RoI flow. We choose the initial distribution of latents and RoIs both to be standard Gaussian distributions for simplicity.

Given the number of latents N already being highly reduced (e.g., 64), we do not perform any latent "grouping" and "ungrouping" operations in SiT to reduce computation. In other words, a latent from SF-VAE directly corresponds to a token in the transformer in SF-SiT.

RoI-based positional encoding. Since we get rid of the grid-based latents, it is crucial to inject positional information dependent on RoIs $\mathbf{r}_t$ into the transformer. Otherwise, the latents $\mathbf{z}$ becomes fully permutation invariant due to the transformer's architecture. We use the sinusoidal positional encoding on four values of RoIs, (x, y, h, w), and concatenate them together. We add them as position encoding to a corresponding token in the transformer. It is worth noting that this positional encoding is totally based on RoIs and therefore changes for the same latent across different training and inference timesteps as RoIs move across different timesteps.

Asynchronous interpolation schedule. The time t indicates the signal-to-noise ratio (SNR) in the interpolation schedule, where $t = 0$ means pure noise and $t = 1$ means target data. Recall that we need to model both $\mathbf{z}$ and $\mathbf{r}$ and therefore SF-SiT are required to handle pure noise RoIs at time $t = 0$ as well as not-so-clean RoIs in early t period. This differs from the regular SiT where the position information is always clean, and the model have strong position information given at any t to predict the latent velocity to denoise them. Therefore, we design an asynchronous interpolation schedule for RoIs compared with latents, where we interpolate RoIs to approach the target RoIs faster than latents in a polynomial way:

$$\sigma_{\mathbf{r},t} = (1-t)^{\beta}, \qquad \alpha_{\mathbf{r},t} = 1 - (1-t)^{\beta}, \tag{12}$$

where β is a control parameter, and reduces to the synchronous linear interpolation when $\beta = 1$. We keep the linear interpolation schedule untouched for latents:

$$\sigma_{\mathbf{z},t} = 1 - t, \qquad \alpha_{\mathbf{z},t} = t. \tag{13}$$

This asynchronous interpolation schedule for latents and RoIs allows the velocity prediction for $\mathbf{z}_t$ to be more position-aware with cleaner RoIs as $\mathbf{r}_t$ approaches the target RoIs faster, as shown in the Fig 4. It is worth noting that the asynchronous interpolation schedule only affects the training phase, i.e., the velocity computation of latents and RoIs. The inference logic is the same as regular SiT, where we solve the ODE to sample the target latents and RoIs. To force the model to learn the accurate RoI distribution, we also impose an additional L_1 between the predicted RoI velocity and the ground-truth, with a balancing weight w_{L_1}, as proposed by LayoutFlow [34] to model bounding box flow more accurately in layout generation.

4 Experiments

To verify the feasibility of synthesizing images with sparse non-grid latents, we conduct experiments on the standard ImageNet benchmark [35], mainly on 256×256 images. We first discuss our autoencoders, SF-VAE, and then present the results of our generative model, SF-SiT. Note that our aim is not to achieve new state-of-the-art results on a benchmark, but to explore sparsity in image generation pipeline.

4.1 Sparse Flow Autoencoders

Setup. As discussed in previous section, our SF-VAE consists up of a SparseFormer encoder $\mathcal{E}$ and a neural field-based decoder $\mathcal{D}$, where the latent space defined by latent variables and RoIs. To keep as lightweight as possible, we design a SparseFormer encoder of 8 blocks with transformer dimension 512 to extract latents and RoIs, where the leading 4 blocks extract RoI regional features from the image. The neural field decoder also consists of 8 transformer blocks of 512 dim, and then

Table 1: **Reconstruction results** on 50K ImageNet validation samples.

resolution	method	latent shape	params	rFID↓	PSNR↑	SSIM↑	LPIPS↓
256×256	SD-VAE-ema-f8	$32 \times 32 \times 4$	84M	0.63	24.98	0.804	0.062
	DC-AE-in-f32c32	$8 \times 8 \times 32$	323M	0.77	23.92	0.765	0.086
	DC-AE-mix-f32c32	$8 \times 8 \times 32$	323M	0.96	23.75	0.763	0.088
	SF-VAE	64×32	133M	0.70	23.24	0.743	0.085
512×512	SD-VAE-ema-f8	$64 \times 64 \times 4$	84M	0.19	27.36	0.849	0.061
	DC-AE-in-f32c32	$16 \times 16 \times 32$	323M	0.20	26.23	0.815	0.078
	SF-VAE	256×32	133M	0.29	25.03	0.787	0.088

produce dynamic MLP's parameters, where the hidden dimension of two-layer MLP is 64. The output of the neural field decoder is then decoded into raw pixels by $\mathcal{D}_{\text{pixel}}$ of conventional VAE decoder architectures [1] but we reduce the dimension of most convolutional layers. The computational cost of the resulting SF-VAE decoder is 153GFLOPs for a 256×256 image, where the SD-VAE-ema-f8 decoder needs 311GFLOPs. The latent $\mathbf{z}$ is 32-dim, and the default number of latents is 64. The loss follows the convention of VAE in latent diffusion models, with the L_2 loss for pixel reconstruction, perceptual loss [36] and GAN adversarial loss [37] for visual perceptual details, as well as KL divergence [24] for the latent regularization on $\mathbf{z}$. We train the SF-VAE on the ImageNet training set with a batch size of 128 for 320K iterations (equivalent to 32 epochs) with a learning rate of 10^{-4}. We perform reconstruction FID evaluation [38] on the ImageNet validation 50K samples.

Reconstruction results. We compare our SF-VAE with the popular SD-VAE-f8 [1] that comes with a downsampling rate of 8, resulting in a latent shape of $32 \times 32 \times 4$. We also include the recently proposed deep compression autoencoder (DC-AE) [23] with a latent shape of $8 \times 8 \times 32$ for comparison, which is exactly the same as our default latent shape. The results are shown in Table 1.

The reconstruction FID of our SF-VAE is 0.70, which is competitive with the SD-VAE (rFID 0.63) with 1024 latents. Also, if counting the latent dimension together, the SF-VAE comes with $64 \times 32 = 2048$ units, much less than SD-VAE's $32 \times 32 \times 4 = 4096$ units. Compared with DC-AE also trained on ImageNet samples and with the same latent shape, our SF-VAE achieves a lower rFID score while maintaining significantly fewer parameters.

More importantly, the reconstruction results further demonstrate that non-grid latents are also effective for image reconstruction. These findings align with recent studies, which show that explicit 2D structured latents are not necessary for high-quality reconstructions and token numbers can be extremely compressed [39–41]. However, our methods does not necessarily belong to the 1D tokenization family, since there are explicit geometric properties, i.e., RoIs, associated with each latent. These 2D RoI representations also maintain the translation equivariance property of this pipeline, while 1D tokenization methods typically fail to address this point.

Varying number of latents and higher image resolution. As discussed before, SF-VAE enjoys the flexibility of decoupling the number of latents from a specified image resolution. Here, we investigate our SF-VAE with more aggressive latent reduction for a fixed resolution of 256×256 in Table 2. We do not change the architecture of our SF-VAE, but only changes the number of latents from 64 to 48 and 32 and maintains the latent dimension unchanged. To reduce training costs, we initialize SF-VAE with new latent configurations from pre-trained weights of the 64-latents model, and finetune them for 20K iterations. Here, we find 32 tokens have already been able to reconstruct pixels with an rFID of 1.70, and 48 tokens give a competitive 0.99 rFID.

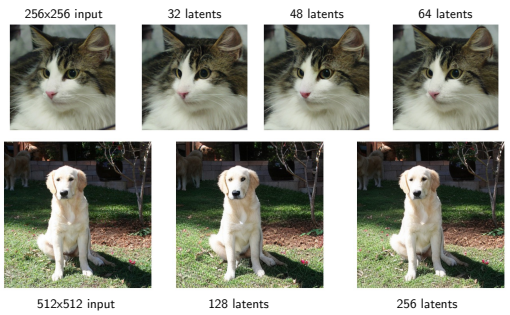

Figure 5: SF-VAE reconstructions with different latent numbers. Better zoom in.

Table 2: Token numbers for SF-VAE reconstructions.

resolution	#latent	rFID
256×256	128	0.35
	64	0.70
	48	0.99
	32	1.70
512×512	256	0.66
	128	1.29

Table 3: Effects of asynchronous interpolating schedule.

interpolating schedule	FID↓
linear, $\beta = 1$	44.4
async, $\beta = 2$	**42.0**
async, $\beta = 4$	44.6
async, $\min(2 \times t, 1)$	54.0
async, $\min(4 \times t, 1)$	55.9
decoupled, RoIs first	63.3

Table 4: Effect of L_1 loss.

w_{L_1}	FID
0.0	43.9
0.2	**42.0**
0.5	42.6
1.0	45.1

Table 5: Comparison with diffusion/flow-based transformers with grid-based latents under 400K training iterations without CFG. *SiT reproduced using official code. † sampled with ODE Euler sampler with NFE=250 for fair comparison. Flops are calculated with one single NFE.

method	#tokens	#params	flops (G)	FID↓	IS↑
DiT-B/2	256	130M	21.8	43.5	-
SiT-B/2	256	130M	21.8	33.0	-
SiT-B/2*†	256	130M	21.8	37.0	39.6
SF-SiT-B†	64	138M	5.9	**33.6**	**41.4**
DiT-XL/2	256	675M	114.4	19.5	-
SiT-XL/2	256	675M	114.4	17.2	-
SiT-XL/2*†	256	675M	114.4	18.8	70.9
SF-SiT-XL†	64	685M	29.3	**18.6**	**71.6**

We also study SF-VAE on higher image resolution of 512×512 in Table 2. We first scale up the number of latents to 256 similar to dense grid-based VAEs, where latent numbers are increasing quadratically regarding input image size. Then we reduce the number of latents to 128. We find that our SF-VAE can still achieve competitive reconstructions for higher images in Figure 5.

4.2 Sparse Flow Transformers

Setup. Our SF-SiT generally follows the design of original SiT transformers [10], where the input and output are retargeted to latents $\mathbf{z}$ and RoIs $\mathbf{r}$, and the velocity of them, $\hat{\mathbf{v}}_z$ and $\hat{\mathbf{v}}_r$. We mainly experiment SF-SiT-B and SF-SiT-XL variants, whose configurations strictly follow the original SiT-B and SiT-XL. Since SF-SiT are now required to output RoI velocity besides latent velocity, we add an additional AdaLN head [42] to predict RoI velocity, where the output is also initialized to zeros similar to latent prediction.

We train SF-SiT variants from scratch on ImageNet training set with 64 latents and RoIs from SF-VAE, unless otherwise specified. In SF-SiT, each latent directly corresponds to a token in the transformer blocks, as our architecture eliminates extra patchifying operations. The training configurations are the same as the original SiT models, i.e., the global batch is 256 and the Adam optimizer [43] with constant learning rate 10^{-4}. We set the default β to 2 in the asynchronous interpolating schedule and the weight w_{L_1} to 0.2 for the additional L1 loss on RoI velocity prediction. We evaluate our SF-SiT using ADM evaluation pipeline [44] and report FID with 50K samples.

Comparison with grid-based diffusion/flow transformers. We first compare our methods under the 400K iteration budget with grid-based DiT/SiT baseline [11, 10] in Table 5 without classifier-free guidance (CFG). It is worth noting that original SiT models adopt the SDE sampler during inference and report their results. This differs from SF-SiT models, which use simple ODE sampler for latent and RoI inference. Due to the lack of publicly available intermediate SiT checkpoints, we reproduce SiT-B and SiT-XL variants using the official code, and inference samples with ODE Euler sampler. We also align the number of function evaluations (NFE) in SF-SiT and SiT models to 250 to make fair comparison.

As shown in Table 5, SF-SiT models, despite operating on significantly reduced latents, achieve performance on par with SiT models using the Euler ODE sampler. We experimented SF-SiT models with the original SiT's SDE sampler but observed degraded RoI quality and visual distortion in decoded images. We hypothesize this sensitivity arises from that SF-VAE encoded RoIs being deviating from standard Gaussian distributions, making them incompatible with the noise introduced by SDE sampling. Overall, these findings suggest that jointly modeling the flow of latents and RoIs have competitive performance paired with ODE-based samplers.

Ablation studies. For all ablation studies, we resort to SF-SiT-B variant with 200K budget due to the limited computational resource. We also inference samples in ablations with the Heun ODE solver with 50 NFE for efficiency.

Asynchronous interpolation schedule. Our proposed SF-SiT introduces additional RoIs $\mathbf{r}$ as additional modeling targets. As discussed before, we propose the asynchronous interpolation schedule

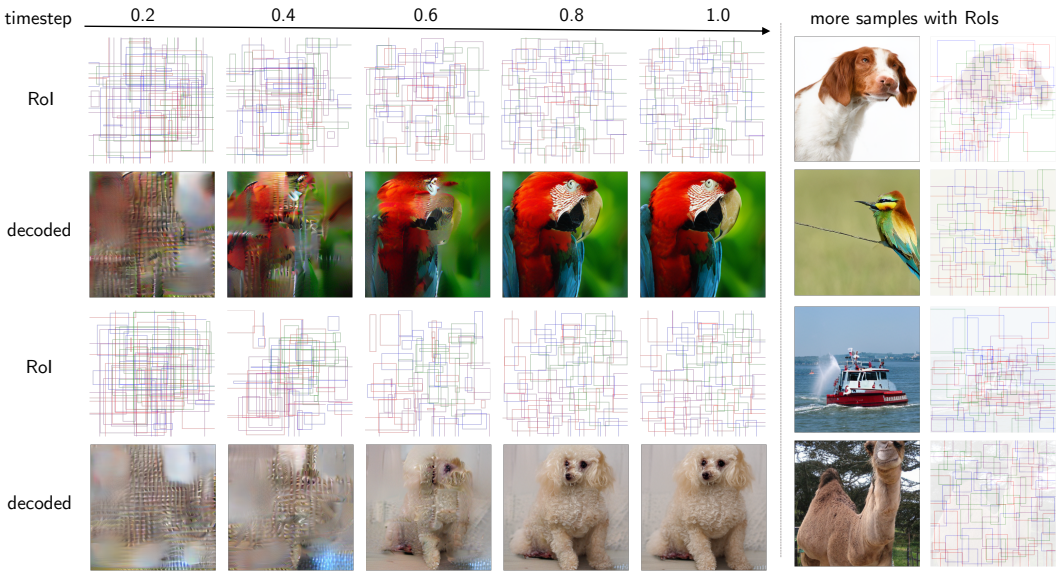

Figure 6: Visualization of RoI flows and decoded images with SF-SiT-XL and CFG=4.0.

for $\mathbf{r}$ to be more compatible in the joint modeling with the latents $\mathbf{z}$. Here we investigate the effect of the asynchronous interpolation schedule in Table 3. We can find that the asynchronous interpolation schedule $\beta = 2$ for $\mathbf{r}$ gives the best result. In contrast, synchronous interpolation ($\beta = 1$) between $\mathbf{r}$ and $\mathbf{z}$, or purely linear interpolation for both, results in the inferior FID. We also try more aggressive asynchronous schedule, where the interpolated RoI $\mathbf{r}$ reaches the target RoI linearly but at the earlier timestep, e.g., $t = 0.25$ for $\min(4 \times t, 1)$. These clamped linear schedules are also asynchronous, where we suppose to sample RoIs first and then sample latents in a hard way, leading to significant degradation in FID. Finally, the decoupled interpolation schedule indicates that we first schedule RoI denoising separately and individually when t is in $[0, 0.5]$ and then schedule latent denoising when t is in $[0.5, 1]$, where the denoising processes for $\mathbf{r}$ and $\mathbf{z}$ are fully decoupled and disjoint, leading to the worst FID. These findings show that the joint flow of latents and RoIs is crucial to good performance in SF-SiT framework but also needs soft asynchronous interpolation between $\mathbf{z}$ and $\mathbf{r}$ to have better performance.

RoI velocity L_1 loss. We also investigate the effect of L_1 loss on the RoI velocity prediction, as proposed by. In line with [34], the L_1 loss weight 0.2 gives the best result.

Longer training schedule. We further train the largest SF-SiT variant, SF-SiT-XL, with longer training schedule. To accelerate convergence on limited hardware resource, we use REPA [45] for our SF-SiT-XL training. We use DINOv2-B/14 [46] as our REPA align target. Since our SF-SiT

Table 6: Class conditional generation results on 256×256 ImageNet. P refers to precision and R refers to recall following [44].

method	#tokens	train steps	FID↓	IS↑	P↑	R↑
LDM-4 [1]	-	-	10.56	103.5	0.71	0.62
DiT-XL/2 [11]	256	7M	9.62	121.5	0.67	0.67
SiT-XL/2 [10]	256	7M	8.26	131.6	0.68	0.67
SiT-XL/2 w/ REPA [45]	256	1M	6.4	-	-	-
SF-SiT-XL w/ REPA	64	1.4M	7.35	131.9	0.70	0.66
DiT-XL/2 (CFG=1.5)	256	7M	2.27	278.2	0.83	0.57
SiT-XL/2 (CFG=1.5)	256	7M	2.06	277.5	0.83	0.59
SF-SiT-XL w/ REPA (CFG=1.5)	64	1.4M	2.99	279.2	0.82	0.54
SF-SiT-XL w/ REPA (CFG=1.375)	64	1.4M	2.76	247.7	0.81	0.58

does not form grid-based feature maps, we need to transform intermediate representations in SF-SiT transformer blocks into a grid using the time-dependent interpolated $\mathbf{r}_t$ as positional properties. Here, we use a simple cross attention from dense queries on grids to SF-SiT tokens with the positional embedding based on $\mathbf{r}_t$ added into keys and values. Dense grid queries are then passed through an MLP to align with DINOv2.

Table 6 shows SF-SiT-XL (w/ 64 latents) results with longer training budget. Note that here we only want to show the potential of SF-SiT-XL on our limited budget with the fast convegence of REPA (it have already took 7 days on 8 A100s to complete 1.4M steps). SF-SiT-XL reaches the competitive performance (7.35 FID w/o CFG and 2.76 w/ CFG) with significantly reduced tokens and computation used. It is expected to have better results when training for more iterations and we will investigate them further when computational resources are available.

5 Conclusion

In this paper, we proposed a paradigm for visual generative modeling by using non-grid sparse latents with their positional properties, challenging the dense grid convention for image synthesis. We first designed sparse flow variational autoencoder, SF-VAE, which encodes raw image pixels into latents together with their RoIs. With SF-VAE, we can compress 256×256 images with down to 32 latents for good reconstruction fidelity. We then propose sparse flow-based generative transformers, SF-SiT, to model the joint flow of latents and RoIs with the highly reduced latent space. With carefully-designed asynchronous interpolation schedule and loss for RoIs, SF-SiT models have competitive performance compared with diffusion/flow transformers. Although not targeting new state-of-the-art results, SF-SiT-XL with just 64 tokens still reaches a competitive 2.76 FID on ImageNet with longer training schedule. We hope that our work can facilitate further research directions on non-grid and sparse generative methods, as well as sparse approaches for generation in other domain, e.g., audio and video.

Limitations. Due to the lack of computational resources, we only train our SF-SiT-XL variant with the help of representation alignment (REPA) for fast convergence, and under fewer tokens settings, it might require more training iterations to converge with RoI modeling compared with grid-based SiT/DiTs for large parameter models. Also, the large scale text-to-image experiments remain exploration in the future. As this paper mainly aims for visual generative modeling, it inherits safety risks regarding its generated content. As the standard ImageNet being the training dataset, which is known to be a relatively safe for academic purposes, these concerns might not be fully addressed in our current framework.

Acknowledgement. This project is supported by the National Research Foundation, Singapore under its NRFF Award NRF-NRFF13-2021-0008.

References

[1] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10674–10685. IEEE, 2022.

[2] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 8821–8831. PMLR, 2021.

[3] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, Dhruv Choudhary, Dingkang Wang, Geet Sethi, Guan Pang, Haoyu Ma, Ishan Misra, Ji Hou, Jialiang Wang, Kiran Jagadeesh, Kunpeng Li, Luxin Zhang, Mannat Singh, Mary Williamson, Matt Le, Matthew Yu, Mitesh Kumar Singh, Peizhao Zhang, Peter Vajda, Quentin Duval, Rohit Girdhar, Roshan Sumbaly, Sai Saketh Rambhatla, Sam S. Tsai, Samaneh Azadi, Samyak Datta, Sanyuan Chen, Sean Bell, Sharadh Ramaswamy, Shelly Sheynin, Siddharth Bhattacharya, Simran Motwani, Tao Xu, Tianhe Li, Tingbo Hou, Wei-Ning Hsu, Xi Yin, Xiaoliang Dai, Yaniv Taigman, Yaqiao Luo, Yen-Cheng Liu, Yi-Chiao Wu, Yue Zhao, Yuval Kirstain, Zecheng He, Zijian He, Albert Pumarola, Ali K. Thabet, Artsiom Sanakoyeu, Arun Mallya, Baishan Guo, Boris Araya, Breena Kerr, Carleigh Wood, Ce Liu, Cen Peng, Dmitry Vengertsev, Edgar Schönfeld, Elliot Blanchard, Felix Juefei-Xu, Fraylie Nord, Jeff Liang, John Hoffman, Jonas Kohler, Kaolin Fire, Karthik Sivakumar, Lawrence Chen, Licheng Yu, Luya Gao, Markos Georgopoulos, Rashel Moritz, Sara K. Sampson, Shikai Li, Simone Parmeggiani, Steve Fine, Tara Fowler, Vladan Petrovic, and Yuming Du. Movie gen: A cast of media foundation models. *CoRR*, abs/2410.13720, 2024.

[4] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*. OpenReview.net, 2023.

[5] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: improving latent diffusion models for high-resolution image synthesis. In *ICLR*. OpenReview.net, 2024.

[6] Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2023.

[7] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. The MIT Press, 07 2010.

[8] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W. Battaglia. Generating images with sparse representations. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 7958–7968. PMLR, 2021.

[9] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*. OpenReview.net, 2023.

[10] 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 *ECCV (77)*, volume 15135 of *Lecture Notes in Computer Science*, pages 23–40. Springer, 2024.

[11] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, pages 4172–4182. IEEE, 2023.

[12] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *CoRR*, abs/2207.12598, 2022.

[13] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, pages 12873–12883. Computer Vision Foundation / IEEE, 2021.

[14] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. Maskgit: Masked generative image transformer. In *CVPR*, pages 11305–11315. IEEE, 2022.

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[16] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*. OpenReview.net, 2021.

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

[18] 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 *CVPR*, pages 22669–22679. IEEE, 2023.

[19] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 13213–13232. PMLR, 2023.

[20] Emiel Hoogeboom, Thomas Mensink, Jonathan Heek, Kay Lamerigts, Ruiqi Gao, and Tim Salimans. Simpler diffusion (sid2): 1.5 FID on imagenet512 with pixel-space diffusion. *CoRR*, abs/2410.19324, 2024.

[21] Xiaoliang Dai, Ji Hou, Chih-Yao Ma, Sam S. Tsai, Jialiang Wang, Rui Wang, Peizhao Zhang, Simon Vandenhende, Xiaofang Wang, Abhimanyu Dubey, Matthew Yu, Abhishek Kadian, Filip Radenovic, Dhruv Mahajan, Kunpeng Li, Yue Zhao, Vladan Petrovic, Mitesh Kumar Singh, Simran Motwani, Yi Wen, Yiwen Song, Roshan Sumbaly, Vignesh Ramanathan, Zijian He, Peter Vajda, and Devi Parikh. Emu: Enhancing image generation models using photogenic needles in a haystack. *CoRR*, abs/2309.15807, 2023.

[22] Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *CVPR*, pages 15703–15712. Computer Vision Foundation / IEEE, 2025.

[23] 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. *CoRR*, abs/2410.10733, 2024.

[24] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[25] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

[26] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV (1)*, volume 12346 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2020.

[27] Ziteng Gao, Zhan Tong, Limin Wang, and Mike Zheng Shou. Sparseformer: Sparse visual recognition via limited latent tokens. In *ICLR*. OpenReview.net, 2024.

[28] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV (1)*, volume 12346 of *Lecture Notes in Computer Science*, pages 405–421. Springer, 2020.

[29] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul P. Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Nießner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhöfer, and Vladislav Golyanik. Advances in neural rendering. *Comput. Graph. Forum*, 41(2):703–735, 2022.

[30] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020.

[31] N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

[32] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*. OpenReview.net, 2023.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[34] Julian Jorge Andrade Guerreiro, Naoto Inoue, Kento Masui, Mayu Otani, and Hideki Nakayama. Layoutflow: Flow matching for layout generation. In *ECCV (36)*, volume 15094 of *Lecture Notes in Computer Science*, pages 56–72. Springer, 2024.

[35] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[36] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595. Computer Vision Foundation / IEEE Computer Society, 2018.

[37] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[38] 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. In *NIPS*, pages 6626–6637, 2017.

[39] Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. *CoRR*, abs/2406.07550, 2024.

[40] Shivam Duggal, Phillip Isola, Antonio Torralba, and William T. Freeman. Adaptive length image tokenization via recurrent allocation. In *ICLR*. OpenReview.net, 2025.

[41] Wilson Yan, Volodymyr Mnih, Aleksandra Faust, Matei Zaharia, Pieter Abbeel, and Hao Liu. Elastictok: Adaptive tokenization for image and video. In *ICLR*. OpenReview.net, 2025.

[42] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, pages 3942–3951. AAAI Press, 2018.

[43] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

[44] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, pages 8780–8794, 2021.

[45] 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. *CoRR*, abs/2410.06940, 2024.

[46] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *Trans. Mach. Learn. Res.*, 2024, 2024.

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: The main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper discusses the limitations of the work performed by the authors.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: The paper provides sufficient information to reproduce the experimental results.

   Guidelines:

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

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

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

Answer: [No]

Justification: Now our codebase is not ready for being publicly available. We will release the code upon acceptance.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The paper provides sufficient information to reproduce the experimental results.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: Experiments in this paper are computation-consuming and due to the limited computational resources, we cannot run the experiments repeatedly.

   Guidelines:

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

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The paper provides sufficient information on the computer resources needed to reproduce the experiments.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The paper adheres to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [NA]

    Justification: The paper focuses on improving models in the academic area.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: The paper is investigating the new diffusion models on the standard ImageNet dataset from scratch. The ImageNet dataset is commonly used for academic research and is considered relatively safe.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: This paper properly credits the creators or original owners of assets used in the paper and mentions the license and terms of use explicitly.

    Guidelines:

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

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

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

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

    Answer: [NA]

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

    Guidelines:

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

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

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

    Answer: [NA]

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

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

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 rigorousness, or originality of the research, declaration is not required.

Answer: [No]

Justification: No LLMs used in this paper.

Guidelines:

- The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.