Advanced Routing as Regularization Allocation for Efficient Diffusion Transformer Training

Qin Ma **Xiaoqi Sun** **Bo Li** **Yuquan Zhou** **WEIZHONG ZHANG**[†]

School of Data Science, Fudan University
weizhongzhang@fudan.edu.cn

Abstract

Diffusion Transformers (DiTs) have achieved strong performance in visual generation, but their dense token processing leads to slow training convergence and high computational cost. Recent token routing methods such as TREAD accelerate training by allowing a random subset of tokens to bypass intermediate layers. It can be expected that carefully tuning the routing ratios over time steps or layers can achieve more significant accelerations. However, the theoretical mechanisms underlying stochastic routing remain underexplored, making the tuning of routing ratios intractable. In this paper, we first theoretically show that stochastic token routing can be interpreted as an implicit route-sensitivity regularizer. Under this view, the routing ratio determines the strength of the induced regularization: larger routing ratios introduce stronger route-induced perturbations and therefore stronger regularization pressure. This further motivates a noise-conditioned routing strategy. In diffusion training, samples at larger timesteps contain heavier noise and often provide less stable gradient signals, suggesting that they may require stronger regularization. We therefore assign larger routing ratios to higher-noise samples, allowing the routing-induced regularization strength to adapt to the noise condition. Based on this insight, we propose *Noise-Aware Routing*, which partitions samples within each mini-batch into low-, middle-, and high-noise regions and assigns progressively larger routing ratios to them. This simple noise-conditioned allocation synergistically bridges computational savings with enhanced model priors. Extensive experiments on class-conditional ImageNet generation show that our method accelerates training convergence and improves generation quality. Noise-Aware Routing achieves up to **15.6**$\times$ and **20**$\times$ convergence speedups over DiT baselines—and outperforms TREAD—on ImageNet-256 and ImageNet-512, respectively, measured by the training iterations required to match our 400K-step FID. In the guided setting, our method achieves an FID of 2.67 with a substantially shorter training iterations without architectural changes.

1 Introduction

Diffusion models have become a dominant framework for visual generation, achieving strong results in image, video, and multimodal synthesis. While early systems mainly used convolutional U-Net backbones [1, 2, 3], recent work has increasingly adopted token-based architectures. Diffusion Transformers (DiTs) replace U-Nets with transformers over latent image tokens, bringing the scalability and generality of vision transformers to diffusion generation [4]. However, their dense token processing makes training costly, as DiTs repeatedly apply global mixing to latent-token sequences and self-attention scales quadratically with sequence length [5]. Therefore, improving DiT training efficiency without sacrificing generation quality is an important problem.

[†]Corresponding author.

Recent work has addressed this bottleneck through more efficient token usage, auxiliary representation learning, and computation control during training. Masked and self-supervised diffusion transformer methods improve DiT training by masking, reconstructing, or discriminatively supervising subsets of latent tokens, including MDT [6], MaskDiT [7], and SD-DiT [8]. Representation alignment methods instead improve the internal features learned by DiTs during generative training, as shown by REPA [9]. More broadly, token pruning, token merging, and dynamic-depth mechanisms in transformer backbones show that token redundancy can be exploited for efficient computation, including DynamicViT [10], EViT [11], Token Merging [12], and Mixture of Depths [13]. In the DiT setting considered in this work, TREAD [14] provides a particularly simple training-time token routing mechanism: during training, a randomly selected subset of tokens is transported from an early layer to a deeper layer, bypassing a contiguous block of computation before being reintroduced into the network. Since the routed tokens are only temporarily absent, TREAD differs from masking, pruning, or merging methods that remove, replace, or combine token information during computation. Since routing is used only during training, TREAD introduces no additional parameters and keeps the original inference architecture unchanged. These properties make TREAD an appealing starting point for efficient diffusion transformer training.

Despite its empirical success, the mechanism behind TREAD remains underexplored. In the original formulation, routing is primarily motivated as a way to improve training efficiency and effectiveness by allowing selected tokens to skip intermediate layers [14]. However, the stochastic nature of token routing suggests a broader interpretation. Randomly altering computation paths has long been connected to regularization: Dropout regularizes neural networks by randomly removing hidden activations during training [15], while Stochastic Depth randomly bypasses residual blocks and has been further shown to induce explicit regularization effects governed by survival probabilities [16, 17].

Motivated by this connection, we provide a route-sensitivity regularization view of TREAD. To formalize this view, we show that stochastic token routing admits a local route-consistency surrogate whose first-order approximation is controlled by the routing ratio ρ. This suggests that the routing ratio affects the strength of routing-induced regularization, and we further support this view through training-loss-gap and gradient-stability analyses. The same analysis also suggests a mismatch between fixed routing and noise-conditioned diffusion training. Different noise regimes impose different denoising requirements and therefore also demand different regularization strengths. This is consistent with prior observations that diffusion timesteps are not equally useful or equally difficult during training, motivating timestep-dependent loss weighting, SNR-aware supervision, and nonuniform timestep sampling strategies [18, 19, 20, 21]. Following this analysis, we propose Noise-Aware Routing for Regularization Matching, or simply Noise-Aware Routing. In diffusion training, higher-noise samples are farther from the data manifold and often yield less stable gradient signals, suggesting that they require stronger regularization than lower-noise samples. We therefore assign appropriate routing ratios to different noise ranges, allowing the strength of routing-induced regularization to adapt to the noise condition.

Our contributions are summarized as follows:

- We reinterpret TREAD as route-sensitivity regularization for noise-conditioned diffusion training, showing that routing ratio controls both computation and regularization strength.
- We validate this view through training-loss-gap and gradient signal-to-noise analyses, showing that routing changes training dynamics beyond computational acceleration.
- We propose **Noise-Aware Routing for Regularization Matching**, which assigns different routing ratios across noise regimes without changing the inference architecture, and we achieve an FID improvement of **5.83** over TREAD on ImageNet-512 [22].

2 Related Work

2.1 Efficient Diffusion Transformer Training

Diffusion models generate data by gradually corrupting clean samples with Gaussian noise and learning a reverse denoising process. Early denoising and score based formulations established the basic training framework [23, 1, 24], while later work improved objectives, sampling efficiency, guidance, and latent space training [25, 26, 2, 27, 3]. More recently, Diffusion Transformers (DiTs) replace convolutional U-Net backbones with transformers operating over latent image tokens [4].

DiTs exhibit strong scaling behavior as model compute increases, but their dense token processing and repeated global mixing operations also make training expensive. Improving DiT training efficiency has therefore become an important direction for scalable visual generation.

A major line of work improves DiT training by exploiting token redundancy or by changing how token representations are processed during training. Masked diffusion transformer methods reduce computation by hiding, removing, or reconstructing part of the token sequence during training [6][7][8]. Related efficient transformer methods similarly show that not all visual tokens need identical computation, using mechanisms such as token pruning, token reorganization, token merging, or dynamic depth allocation [10][11][12][13]. These methods demonstrate the usefulness of token redundancy for efficient computation, but many of them remove, replace, merge, or selectively process tokens according to masking or importance based rules. In the DiT setting considered in this work, TREAD instead uses training-time token routing [14]. It transports a randomly selected subset of tokens from an early layer to a deeper layer, allowing these tokens to bypass a contiguous block of intermediate computation before being reintroduced into the network. Since routed tokens are restored after the route, TREAD differs from token masking, pruning, or merging methods that permanently remove, replace, or combine token information during computation. Since the routing mechanism is used only during training, TREAD introduces no additional parameters and keeps the original inference architecture unchanged.

2.2 Noise Aware Training and Stochastic Regularization

A central property of diffusion training is that each sample is associated with a noise level or timestep, and different noise regimes correspond to different denoising tasks. Prior work has shown that treating these regimes uniformly is often suboptimal. P2 weighting redesigns timestep loss weights to emphasize perceptually useful denoising regimes [18], while Min-SNR treats denoising at different timesteps as a multi-task problem and mitigates conflicting optimization directions through SNR-based weighting [19].Others analyze how timestep sampling, SNR distributions, or process increments affect training efficiency, showing that supervision is not equally useful across all noise regions [20][21]. A related line of work studies the noise dependence of diffusion features and guidance behavior, showing that feature quality or guidance usefulness can vary substantially with the noise level [28][29]. Together, these works indicate that diffusion models do not treat all noise levels uniformly. They motivate noise aware training strategies, although existing approaches mainly adjust loss weighting, timestep sampling, feature extraction, representation supervision, or sampling guidance rather than token routing.

Stochastic training mechanisms are also closely connected to regularization. Dropout randomly removes hidden activations during training and is widely understood as a regularization strategy [15]. Stochastic Depth randomly bypasses residual blocks, enabling shorter effective networks during training while using the full network at test time [16]. Later theoretical work shows that Stochastic Depth induces explicit regularization effects governed by survival probabilities [17]. These observations suggest that random changes to computation paths can be viewed not only as efficiency mechanisms, but also as sources of regularization. This perspective is especially relevant for stochastic token routing, where the routing ratio controls how strongly the training computation is perturbed across noise regimes.

3 Method

This section develops our method from objective formulation to routing design. We first cast diffusion and flow-matching training into a unified noise-conditioned regression objective. We then analyze stochastic token routing as an implicit route-sensitivity regularizer, showing how the routing ratio controls the induced regularization strength. Finally, we use this regularization view to motivate noise-aware routing, which assigns condition-dependent routing ratios to different noise regions for better regularization allocation during training.

3.1 Unified Noise-Conditioned Training Objective

We first formulate diffusion and flow-matching training under a **unified noise-conditioned regression objective**. Let $x_{\mathrm{data}} \sim p_{\mathrm{data}}$ denote a clean data sample and let $\lambda \in \Lambda$ denote the conditioning

variable along the generative trajectory. For EDM-style diffusion [27], λ corresponds to the noise scale σ, while for flow matching, λ corresponds to the interpolation time t. We use x_λ to denote the corrupted or interpolated input under condition λ, and y to denote the corresponding regression target, such as clean data, noise, or velocity field.

Under this notation, a broad class of diffusion and flow-matching objectives can be written as

$$\mathcal{L}_{\text{base}}(\theta) = \mathbb{E}_{x_{\text{data}},\epsilon,\lambda}\left[w(\lambda)\,\|D_\theta(x_\lambda,\lambda) - y\|_2^2\right],$$

where D_θ is a **trajectory-conditioned generative backbone** and $w(\lambda)$ is a possible condition-dependent loss weight.

The specific choices of x_λ and y for two representative frameworks are as follows. We use $\boldsymbol{\epsilon}$ to denote the additive corruption noise in EDM-style diffusion, whereas ϵ to denote the sampled endpoint from the base distribution in flow matching, which is fixed along each interpolation path.

- **EDM-style diffusion** [27] – take $\lambda = \sigma$ being the noise scale, $\ln(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$ and
$$x_\lambda = x_{\text{data}} + \lambda\boldsymbol{\epsilon}, \qquad y = x_{\text{data}}\,,$$
where y denotes the clean-data target.
- **Flow matching** [30, 31] – take $\lambda \in [0,1]$ being the interpolation time and
$$x_\lambda = (1-\lambda)\epsilon + \lambda x_{\text{data}}, \qquad y = x_{\text{data}} - \epsilon\,,$$
where y denotes the target velocity field.

This unified form is useful because our analysis does not rely on a specific diffusion parameterization. Instead, the key requirement is that the model is trained by a squared-error objective conditioned on a trajectory variable λ. In the following sections, we study how stochastic token routing modifies this objective. Specifically, we write the routed model as $D_\theta(x_\lambda, \lambda, G)$, where G denotes the token-routing gate. This allows us to analyze routing as a perturbation of the noise-conditioned predictor and to derive the corresponding route-sensitivity regularization.

3.2 Token Routing as Route-Sensitivity Regularization

In a standard DiT [4], all latent tokens are processed by the same sequence of transformer blocks. TREAD [14] modifies only the training-time computation path by introducing a route $r_{i\to j}$ over a contiguous segment of blocks. For each training sample, a subset of tokens bypasses blocks $L_i, \ldots, L_j$, while the remaining tokens are processed normally inside the routed segment. The bypassed tokens are reinserted after the route and continue through the subsequent blocks together with the processed tokens. At inference time, all tokens follow the full DiT path. Thus, TREAD [14] introduces stochastic token routing as an efficient training-time mechanism. We next reinterpret this mechanism from a regularization perspective and show that it induces a **route-sensitivity regularizer**.

Let $G = (G_1, \ldots, G_N) \in \{0,1\}^N$ denote a token-routing gate over N latent tokens, where $G_k = 1$ means token k is processed by the routed segment and $G_k = 0$ means it follows the bypass path. For a route $r_{i\to j}$, the gate is sampled independently for each training sample according to a routing ratio ρ, defined as the fraction of tokens selected to bypass the routed segment. Equivalently, $G_k \sim \text{Bernoulli}(1-\rho)$, so that $\mathbb{E}[G] = (1-\rho)\mathbf{1}$.

Let $X^{(i)} \in \mathbb{R}^{N\times d}$ be the token representation before the routed segment. Given G, the output after the routed segment is written as

$$X^{(j+1)}(G) = \text{Diag}(G)\,\Phi_{i:j}\Big(X^{(i)}; G\Big) + \text{Diag}(\mathbf{1} - G)\,X^{(i)}, \tag{1}$$

where $\Phi_{i:j} = L_j \circ \cdots \circ L_i$ denotes the computation through the routed blocks, and L_k denotes the k-th transformer block of the DiT [4] backbone. The dependence on G indicates that only active tokens participate in the routed-block computation. In practice, routing can be implemented by gathering the active token subset for efficiency. For analysis, we use an equivalent soft attention-mask view that extends the binary gate $G \in \{0,1\}^N$ to a continuous gate $G \in [0,1]^N$.

We denote the routed predictor as $D_\theta(x_\lambda, \lambda, G)$. Stochastic token routing corresponds to optimizing the base objective under random routing gates:

$$\mathcal{L}_{\text{route}}(\theta) = \mathbb{E}_{x_{\text{data}},\epsilon,\lambda,G}\left[w(\lambda)\,\|D_\theta(x_\lambda,\lambda,G) - y\|_2^2\right]. \tag{2}$$

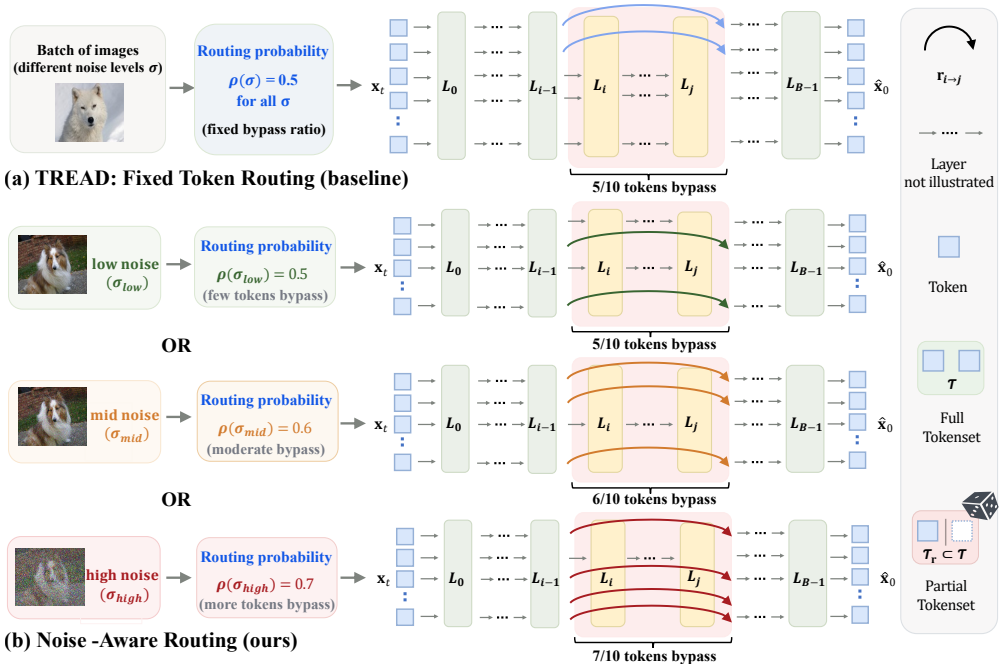


Figure 1: **Noise-aware routing for regularization allocation.** (a) Fixed token routing applies the same routing ratio to all training samples regardless of their noise level. (b) Our noise-aware routing assigns different routing ratios according to the sampled noise condition, allocating stronger routing-induced regularization to higher-noise regions.

Continuous relaxation and first-order approximation. To characterize the effect of routing, let $S = \mathbf{1} - G$ denote the bypass indicator, where $S_k = 1$ means token k is selected for bypassing and $S_k \sim \text{Bernoulli}(\rho)$. We introduce a continuous path $G(\alpha) = \mathbf{1} - \alpha S$, $\alpha \in [0, 1]$, which connects the full-path computation $G(0) = \mathbf{1}$ to the routed computation $G(1) = \mathbf{1} - S$. For a fixed training sample and condition λ, define $\varphi(\alpha) = D_\theta(x_\lambda, \lambda, G(\alpha))$ and $J_{\mathbf{1},\lambda} := \partial D_\theta(x_\lambda, \lambda, G)/\partial G|_{G=\mathbf{1}}$. A first-order Taylor expansion of $\varphi(\alpha)$ at $\alpha = 0$ gives

$$D_\theta(x_\lambda, \lambda, \mathbf{1} - S) \approx D_\theta(x_\lambda, \lambda, \mathbf{1}) - J_{\mathbf{1},\lambda} S. \tag{3}$$

Sensitivity decomposition. We define the route-induced output discrepancy and linearize it using Eq. (3):

$$\mathcal{R}_{\text{route}}(\rho, \lambda) := \mathbb{E}_S \left[\|D_\theta(x_\lambda, \lambda, \mathbf{1} - S) - D_\theta(x_\lambda, \lambda, \mathbf{1})\|_2^2 \right] \approx \mathbb{E}_S \left[\|J_{\mathbf{1},\lambda} S\|_2^2 \right]. \tag{4}$$

Because $\mathbb{E}[S] = \rho\mathbf{1}$ and $\text{Cov}(S) = \rho(1-\rho)I$, this expectation decomposes into mean and variance components:

$$\mathbb{E}_S \left[\|J_{\mathbf{1},\lambda} S\|_2^2 \right] = \underbrace{\rho^2 \|J_{\mathbf{1},\lambda} \mathbf{1}\|_2^2}_{\text{Mean route-rate sensitivity}} + \underbrace{\rho(1-\rho) \|J_{\mathbf{1},\lambda}\|_F^2}_{\text{Token-selection sensitivity}}. \tag{5}$$

A derivation is provided in Appendix B.

Equations (4)–(5) provide an implicit regularized view of stochastic token routing. This yields the following local route-consistency surrogate:

$$\mathcal{L}_{\text{reg}}(\theta) \approx \mathcal{L}_{\text{base}}(\theta) + \mathbb{E}_{x_{\text{data}},\epsilon,\lambda} \left[w(\lambda) \left(\rho^2 \|J_{\mathbf{1},\lambda} \mathbf{1}\|_2^2 + \rho(1-\rho) \|J_{\mathbf{1},\lambda}\|_F^2 \right) \right]. \tag{6}$$

This surrogate is not an additional loss explicitly optimized by TREAD [14]; it is a local characterization of the stability pressure induced by stochastic routing.

The two terms in Eq. (6) capture complementary forms of route sensitivity. The Frobenius term measures *token-selection sensitivity*, i.e., how sensitive the prediction is to which individual tokens are selected for bypassing. The directional term measures *mean route-rate sensitivity*, i.e., how sensitive the prediction is when all token gates jointly move from the full path toward the bypass path.

Remark 1 (Routing ratio determines the regularization strength) *By rewriting the penalty as $\rho\|J_{\mathbf{1},\lambda}\|_F^2 + \rho^2(\|J_{\mathbf{1},\lambda}\mathbf{1}\|_2^2 - \|J_{\mathbf{1},\lambda}\|_F^2)$, we see that when the token-selection and mean route-rate sensitivities have comparable scale, the induced penalty scales approximately linearly with the routing ratio ρ. In the general case, increasing ρ strengthens the mean route-rate component, while the token-selection component is maximized at $\rho = 0.5$. This suggests that larger routing ratios can induce stronger effective route regularization when the mean route-rate sensitivity is non-negligible.*

This decomposition provides the theoretical basis for our noise-aware routing design: the induced route-sensitivity penalty depends not only on the routing ratio ρ, but also on the trajectory condition λ through $J_{\mathbf{1},\lambda}$. We next use this observation to formulate noise-aware routing for regularization matching.

3.3 Noise-Aware Routing for Regularization Matching

Following Remark 1, the strength of the routing-induced regularization is controlled by both the routing ratio ρ and the route sensitivity of the predictor at condition λ. Intuitively, this can be viewed through a compact proxy $\rho\|J_{\mathbf{1},\lambda}\|_F^2$, where ρ is the controllable routing ratio and $\|J_{\mathbf{1},\lambda}\|_F^2$ reflects the condition-dependent sensitivity of the model to routing perturbations. This observation suggests that using a single fixed routing ratio for all noise conditions, as in TREAD [14], may impose the same routing control on regions that require different amounts of regularization.

Motivated by this mismatch, we propose **noise-aware routing**, which assigns different routing ratios to different noise regions instead of using a global fixed ratio. Specifically, we partition the trajectory conditions into noise regions $\{\mathcal{B}_b\}$ and define a region-specific routing policy:

$$\rho(\lambda) = \begin{cases} \rho_{\text{low}}, & \lambda \in \mathcal{B}_{\text{low}}, \\ \rho_{\text{mid}}, & \lambda \in \mathcal{B}_{\text{mid}}, \\ \rho_{\text{high}}, & \lambda \in \mathcal{B}_{\text{high}}. \end{cases}$$

As illustrated in Fig. 1, each training sample selects its routing ratio according to its sampled noise condition. This design keeps the training objective unchanged while adapting the strength of stochastic token routing across noise regions. In our EDM experiments, $\lambda = \sigma$, and the regions are defined over the noise scale. Specifically, we instantiate the noise regions using the log-normal distribution. Given proportions $(\pi_{\text{low}}, \pi_{\text{mid}}, \pi_{\text{high}})$, we compute two quantile thresholds of $\log\sigma \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$, namely $\tau_1 = P_{\text{mean}} + P_{\text{std}}\Phi^{-1}(\pi_{\text{low}})$ and $\tau_2 = P_{\text{mean}} + P_{\text{std}}\Phi^{-1}(\pi_{\text{low}} + \pi_{\text{mid}})$. Each training sample is then assigned to a low-, middle-, or high-noise region according to its sampled noise scale, and the routing ratio is selected sample-wise based on the corresponding region.

4 Experiments

4.1 Experimental Setup

Implementation. We evaluate our method on class-conditional ImageNet [22] generation using Diffusion Transformers [4] trained under the EDM objective [27]. Unless otherwise stated, all models are trained in the latent space of a pretrained VAE [3], following the standard DiT setup. We use DiT-S/2, DiT-B/2, DiT-L/2, and DiT-XL/2 as backbone models to evaluate the scalability of our method. The models are trained with AdamW [32] using a learning rate of 1×10^{-4}, batch size 256, and EMA decay 0.9999. We follow the EDM noise sampling strategy, where $\log\sigma \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$, and use the EDM preconditioning and loss weighting.

For routing-based methods, we apply token routing to a contiguous segment of DiT blocks $r_{i\to j}$. The route location is kept fixed across TREAD [14] and our method to ensure a fair comparison. TREAD uses a fixed routing ratio ρ for all training samples. Our noise-aware routing instead assigns a sample-wise routing ratio according to the sampled noise level. Specifically, we partition the training noise distribution with **proportions** $(\pi_{\text{low}}, \pi_{\text{mid}}, \pi_{\text{high}}) = (\mathbf{0.3}, \mathbf{0.4}, \mathbf{0.3})$, and use **region-specific routing ratios** $(\rho_{\text{low}}, \rho_{\text{mid}}, \rho_{\text{high}}) = (\mathbf{0.5}, \mathbf{0.6}, \mathbf{0.7})$. Guided by the route-sensitivity regularization view, we allocate a larger routing ratio to higher-noise regions, where stronger routing-induced regularization is expected to be more beneficial. This is motivated by the observation that high-noise samples often yield less stable gradient signals, so enforcing stronger route consistency can help stabilize the denoising predictor. For each sample in a mini-batch, the routing gate is sampled according to its assigned region, while inference procedure remain unchanged.

Sampling. For evaluation, DiT-based models are sampled with the 40-step EDM sampler [27], while SiT-based models are sampled with the 250-step Euler–Maruyama sampler [33]. We keep the VAE decoder and sampling hyperparameters fixed within each backbone family for all compared methods. Since routing is used only during training, all models are evaluated with the full DiT computation path at inference time. Therefore, noise-aware routing introduces no additional sampling cost compared with the corresponding DiT backbone.

Evaluation. We evaluate generation quality on ImageNet using FID as the primary metric. Following common practice, we also report sFID [34], Inception Score [35], Precision and Recall [36] when available. All metrics are computed using ADM TensorFlow evaluation suite [2] over $50k$ generated samples unless otherwise stated.

Compute. All experiments are conducted on 4 NVIDIA A100 GPUs. Due to the larger average routing ratio of our noise-aware schedule, $\bar{\rho} = \sum_b \pi_b \rho_b = 0.6$, more tokens bypass the routed segment during training. As a result, our method achieves a training throughput of 3.32 iterations/s, outperforming both the fixed-ratio routing baseline TREAD [14] (3.03 iterations/s) and the vanilla DiT baseline [4] (1.86 iterations/s) under the same hardware setting. This indicates that noise-aware routing not only benefits convergence, but also reduces the computational cost under the same training conditions.

4.2 Main Results on ImageNet

Table 1: **Main quantitative results on ImageNet [22]** 256×256**.** Left: FID [37] comparisons under the same training budget. Right: comparison with state-of-the-art methods. The green number denotes the FID reduction over the original DiT [4]/SiT [33]backbone. *We primarily compare our method with TREAD, as it is the most directly related token-routing baseline. For other methods, we report their published results and show that our method can achieve comparable FID with fewer training epochs.*

(a) FID comparisons

Model	#Param	Iter.	Batchsize	FID↓
DiT-S/2 [4]	33M	400K	256	68.40
+ TREAD [14]	33M	400K	256	47.04$_{-21.36}$
+ **Ours**	33M	400K	256	**41.42**$_{-26.98}$
SiT-S/2 [33]	33M	400K	256	57.60
+ TREAD [14]	33M	400K	256	38.08$_{-19.52}$
+ **Ours**	33M	400K	256	**33.77**$_{-23.83}$
DiT-B/2 [4]	130M	400K	256	43.47
+ TREAD [14]	130M	400K	256	21.34$_{-22.13}$
+ **Ours**	130M	400K	256	**20.20**$_{-23.27}$
SiT-B/2 [33]	130M	400K	256	33.00
+ TREAD [14]	130M	400K	256	15.98$_{-17.02}$
+ **Ours**	130M	400K	256	**15.19**$_{-17.81}$
DiT-L/2 [4]	458M	400K	256	23.33
+ TREAD [14]	458M	400K	256	9.10$_{-14.23}$
+ **Ours**	458M	400K	256	**8.84**$_{-14.49}$
DiT-XL/2 [4]	675M	400K	256	19.47
+ TREAD [14]	675M	400K	256	7.38$_{-12.09}$
+ **Ours**	675M	400K	256	**7.13**$_{-12.34}$

(b) State-of-the-art comparison

Method	Epochs	FID↓	sFID↓	IS↑	Prec.↑	Rec.↑
Unguided						
SD-DiT-XL/2 [8]	480	7.21	5.17	144.68	0.72	0.61
MDT-XL/2 [6]	1300	6.23	5.23	143.02	0.71	0.65
SiT-XL/2 [33]	1400	8.61	6.32	131.65	0.68	0.67
MaskDiT-XL/2 [7]	1600	5.69	10.34	177.99	0.74	0.60
DiT-XL/2 [4]	1400	9.62	6.85	121.50	0.67	0.67
TREAD [14]	160	4.96	6.83	192.49	0.79	0.54
Ours	160	**4.84**	**6.79**	**203.12**	**0.83**	**0.54**
Guided						
SD-DiT-XL/2 [8]	480	3.23	-	-	-	-
MDT-XL/2 [6]	1300	1.79	4.57	283.01	0.81	0.61
MaskDiT-XL/2 [7]	1600	2.28	5.67	276.56	0.80	0.61
SiT-XL/2 [33]	1400	2.06	4.50	270.30	0.82	0.59
DiT-XL/2 [4]	1400	2.27	4.60	278.24	0.83	0.57
REPA [9]	800	1.80	4.50	284.0	0.81	0.61
FasterDiT [20]	400	2.03	4.63	264.00	0.81	0.60
TREAD(F) [14]	220	2.76	4.61	252.78	0.83	0.57
Ours	160	3.18	5.64	245.02	0.83	0.57
Ours(F)	220	**2.67**	**4.58**	**263.12**	**0.82**	**0.56**

Comparison with baselines. Table 1(a) reports FID [37] comparisons on ImageNet [22] 256×256 with DiT [4]/SiT [33] and TREAD [14] Across different model scales, adding token routing consistently improves the original DiT/SiT backbones, confirming that stochastic routing is an effective training-time mechanism. More importantly, our method consistently outperforms the fixed-ratio TREAD baseline in terms of FID. These results show that allocating routing-induced regularization across noise regions is more effective than using a single global routing ratio. According to Table 1(b), Our method achieves a guided FID of 2.67 with only 220 epochs using classifier-free guidance (CFG) with a guidance scale of 1.5. *This shows that noise-aware routing remains compatible with CFG [38] under a shorter training schedule.* Here, (F) denotes an additional fine-tuning stage without routing; we provide the fine-tuning details in Appendix D.

Extension to flow matching. To test compatibility with flow-matching objectives [30], we apply noise-aware routing to SiT [33] backbones, where the trajectory condition λ corresponds to the interpolation time t. We partition $t \in [0, 1]$ with proportions $(0.3, 0.4, 0.3)$ and assign routing ratios $(0.5, 0.6, 0.7)$ from low- to high-noise regions. As shown in Table 1(a), our method outperforms fixed-ratio TREAD [14] on both SiT-S/2 [33] and SiT-B/2 [33], suggesting that regularization allocation transfers beyond EDM-style diffusion.

Scaling to higher resolution. We further test noise-aware routing on ImageNet [22] at 512×512 resolution using DiT-B/2 [4]. As shown in Table 2, our method achieves an FID@10K of 40.21 at 280K iterations and 35.08 at 400K, outperforming the TREAD [14] baseline trained for 400K iterations. This confirms that the improvement transfers to higher-resolution generation.

Table 2: **Scaling to ImageNet** 512×512**.** FID@10K results on ImageNet 512×512.

Model	#Param	Iter.	Batchsize	FID@10K↓
DiT [4]	130M	400K	256	62.88
+ TREAD [14]	130M	400K	256	$40.91_{-21.97}$
+ **Ours**	130M	280K	256	$\mathbf{40.21}_{-22.67}$
+ **Ours**	130M	400K	256	$\mathbf{35.08}_{-27.80}$

4.3 Ablation Studies

Table 3: **Ablation studies.** (a) Regularization allocation ablation on DiT-S/2 trained for 400K iterations on ImageNet 256×256, with noise-region proportions fixed as $(0.3, 0.4, 0.3)$. The continuous policy uses a normal-CDF mapping. (b) Resolution scaling ablation using DiT-B/2 at 400K training iterations. ImageNet-128 and ImageNet-512 are evaluated using FID@10K.

(a) Regularization allocation

ρ schedule	Pattern	FID↓
$(0.6, 0.6, 0.6)$	Uniform	46.36
$(0.7, 0.6, 0.5)$	Reverse	46.71
$(0.5, 0.6, 0.7)$	Noise-aware	**41.42**
CDF $[0.5, 0.7]$	Noise-aware(continuous)	**41.22**

(b) Resolution scaling

Dataset	TREAD	Ours	ΔFID
ImageNet-128	24.83	**24.47**	-0.36
ImageNet-256	21.34	**20.20**	-1.14
ImageNet-512	40.91	**35.08**	-5.83

Regularization allocation. Table 3(a) studies how routing allocation across noise regions affects performance. The three discrete schedules share the same average routing ratio under fixed proportions $(0.3, 0.4, 0.3)$. The uniform schedule $(0.6, 0.6, 0.6)$ achieves an FID of 46.36, while the reverse schedule $(0.7, 0.6, 0.5)$ slightly degrades to 46.71. In contrast, our noise-aware allocation $(0.5, 0.6, 0.7)$ improves FID to 41.42.A continuous noise-to-routing mapping based on the normal cumulative distribution function (Appendix A.2) further achieves an FID of 41.22, suggesting that **the benefit is not limited to discrete noise partitions.** All four strategies have the same expected routing ratio $\bar{\rho} = 0.6$, confirming that the improvement stems from allocating stronger routing-induced regularization to higher-noise regions rather than increasing the average routing ratio.

Resolution scaling. We further evaluate whether the benefit of noise-aware routing changes with image resolution. Using the same DiT-B/2 backbone [4, 14] and a 400K training budget, we compare fixed-ratio routing and our method on ImageNet-128, ImageNet-256, and ImageNet-512 [22]. As shown in Table 3(b), our method consistently improves over the fixed-ratio baseline across all resolutions, with FID reductions of 0.36, 1.14, and 5.83, respectively [37]. The improvement becomes more pronounced as the resolution increases, suggesting that higher-resolution generation benefits more from noise-conditioned regularization allocation.

This trend is consistent with our motivation. As image resolution increases, the number of latent tokens and the complexity of the denoising task grow, making training more sensitive. A single fixed routing ratio becomes less adequate under this setting. The larger gain on ImageNet-512 indicates that our method is especially useful for more challenging high-resolution generation settings.

Table 4: **Additional experiments on ImageNet-256.** (a) Routing allocation and stochasticity on DiT-S/2 at 200K training iterations. (b) Few-step sampling on DiT-B/2 at 400K training iterations without retraining.

(a) Allocation and stochasticity

Method	Token mask	FID↓
TREAD	Fixed	57.73
TREAD	Random	56.38
Ours	Fixed	55.19
Ours	Random	**53.63**

(b) Few-step sampling

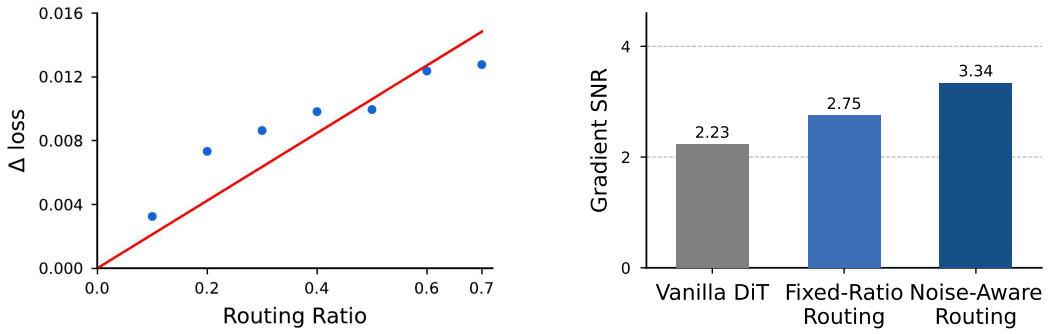
Sampler	Steps	TREAD	Ours
Heun	10	34.30	**31.99**
Heun	20	22.41	**20.63**
DPM++ 2M	5	84.62	**81.92**
DPM++ 2M	10	32.84	**28.81**

Compute allocation and routing stochasticity. Table 4(a) disentangles noise-conditioned compute allocation from token-selection stochasticity on DiT-S/2 at 200K iterations. Even with fixed token masks, our noise-aware routing achieves an FID of 55.19, outperforming both TREAD methods. This suggests that noise-conditioned routing is beneficial even without stochastic token selection. Introducing random mask resampling further improves our FID from 55.19 to 53.63, while also improving TREAD from 57.73 to 56.38. These results display complementary benefits: noise-conditioned compute allocation improves performance independently of token-selection stochasticity, while stochastic routing provides an additional gain. The full method achieves the best performance.

Few-step sampling robustness. Table 4(b) evaluates whether the improvements persist under fewer sampling steps, using the 400K-step checkpoints without retraining. With Heun sampling, our method reduces FID from 34.30 to 31.99 at 10 steps and from 22.41 to 20.63 at 20 steps. With DPM-Solver++ 2M, FID decreases from 84.62 to 81.92 at 5 steps and from 32.84 to 28.81 at 10 steps. These results demonstrate that the benefits of noise-aware routing extend to few-step generation without sampler-specific adaptation.

Broader Applicability. We further examine the applicability of our method beyond the standard experimental setting. Under extended training without routing-free fine-tuning, our method maintains its convergence advantage and achieves better final generation quality. Moreover, the improvement extends from ImageNet class-conditional generation to MS-COCO text-to-image generation, consistently outperforming TREAD. These results support the effectiveness of noise-aware routing across longer training horizons and different generation tasks. Detailed results are provided in Appendix E.

4.4 Analysis



(a) Routing ratio controls the training loss gap.

(b) Noise-aware routing improves gradient stability.

Figure 2: **Analysis of routing-induced regularization and gradient stability.** (a) We train fixed-ratio routing models with different ρ and report the average loss increase over the 8k–10k step window relative to vanilla DiT ($\rho = 0$). (b) Gradient SNR is computed using DiT-S/2 checkpoints at 40k steps on probe mini-batches of size 256. Higher SNR indicates a stronger average gradient signal relative to stochastic fluctuation.

Routing ratio controls the training loss gap. We train fixed-ratio routing models with $\rho \in \{0.1, 0.2, \ldots, 0.7\}$ and compare their training losses with vanilla DiT [4] ($\rho = 0$). For each run, we average the loss over the 8k–10k step window and define $\Delta\text{loss}(\rho) = \overline{\mathcal{L}}_\rho - \overline{\mathcal{L}}_{\rho=0}$. As shown in Fig. 2a, the loss gap grows approximately linearly with ρ. A linear fit through the origin gives $\Delta\text{loss} = 0.0212\rho$, with Pearson correlation $r = 0.95$ and uncentered $R^2 = 0.97$. This trend supports Remark 1: larger routing ratios induce stronger route perturbations and therefore a larger effective regularization penalty. We provide the corresponding estimate of the effective route-regularization coefficient in Appendix C.

Noise-aware routing improves gradient stability. We further compute a vector-level mini-batch gradient SNR [39, 40] using DiT-S/2 [4] checkpoints at 40k training steps. For each method, we sample Q probe mini-batches of size 256 from the EDM log-normal noise distribution and compute the flattened gradient $g_q = \nabla_\theta \mathcal{L}_{\mathcal{B}_q}(\theta)$ without parameter updates. We define

$$\text{SNR} = \frac{\|\bar{g}\|_2}{\sqrt{\frac{1}{Q-1}\sum_{q=1}^{Q} \|g_q - \bar{g}\|_2^2}}, \qquad \bar{g} = \frac{1}{Q}\sum_{q=1}^{Q} g_q.$$

This metric measures the ratio between the coherent gradient component and the batch-to-batch gradient fluctuation. As shown in Fig. 2b, fixed-ratio routing already improves the gradient SNR over vanilla DiT, while noise-aware routing further increases it. This suggests that allocating routing strength according to noise conditions reduces relative mini-batch gradient fluctuations and yields a cleaner training signal.

5 Conclusions

In this work, we revisited stochastic token routing for diffusion transformer training from a route-sensitivity regularization perspective. Our analysis shows that the routing ratio controls not only the amount of skipped computation, but also the strength of routing-induced regularization. This view reveals a mismatch in fixed-ratio routing: diffusion training is noise-conditioned, yet existing routing policies apply the same regularization strength across all noise regimes. To address this, we proposed Noise-Aware Routing for Regularization Matching, which assigns stronger routing to high-noise samples and more conservative routing to low-noise samples while leaving the inference architecture unchanged. Extensive experiments on class-conditional ImageNet generation demonstrate that this simple allocation strategy consistently improves over dense DiT training and fixed-ratio routing baselines across model scales, objectives, and resolutions.

Acknowledgments and Disclosure of Funding

This work was supported by the Smart Grid-National Science and Technology Major Project (2026ZD0812403), the National Natural Science Foundation of China (62472097), and the Shanghai Municipal Science and Technology Commission (Grant No. 25511102200). The computations in this research were performed on the CFFF platform of Fudan University. The authors have no competing interests to declare.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020. URL `https://arxiv.org/abs/2006.11239`.

[2] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *arXiv preprint arXiv:2105.05233*, 2021. URL `https://arxiv.org/abs/2105.05233`.

[3] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. URL `https://www.computer.org/csdl/proceedings-article/cvpr/2022/694600k0674/1H1iFsO7Zuw`.

[4] William S. Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4195–4205, 2023. URL https://arxiv.org/abs/2212.09748.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. URL https://papers.nips.cc/paper/7181-attention-is-all-you-need.

[6] 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 (ICCV)*, pages 23197–23206, 2023. URL https://arxiv.org/abs/2303.14389.

[7] Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar. Fast training of diffusion models with masked transformers. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=vTBjBtGioE.

[8] Rui Zhu, Yingwei Pan, Yehao Li, Ting Yao, Zhenglong Sun, Tao Mei, and Chang Wen Chen. Sd-dit: Unleashing the power of self-supervised discrimination in diffusion transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8435–8445, 2024. URL https://www.computer.org/csdl/proceedings-article/cvpr/2024/530000i435/20hMDJsvBTO.

[9] 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. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=DJSZGGZYVi.

[10] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. URL https://openreview.net/forum?id=kR95DuwwXHZ.

[11] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*, 2022. URL https://arxiv.org/abs/2202.07800.

[12] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=JroZRaRw7Eu.

[13] David Raposo, Sam Ritter, Blake A. Richards, Timothy P. Lillicrap, Peter C. Humphreys, and Adam Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*, 2024. URL https://arxiv.org/abs/2404.02258.

[14] Felix Krause, Timy Phan, Ming Gui, Stefan Andreas Baumann, Vincent Tao Hu, and Björn Ommer. Tread: Token routing for efficient architecture-agnostic diffusion training. *arXiv preprint arXiv:2501.04765*, 2025. URL https://arxiv.org/abs/2501.04765.

[15] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL https://jmlr.org/papers/v15/srivastava14a.html.

[16] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision (ECCV)*, pages 646–661, 2016. URL https://arxiv.org/abs/1603.09382.

[17] Soufiane Hayou and Fadhel Ayed. Regularization in resnet with stochastic depth. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/82ba9d6eee3f026be339bb287651c3d8-Abstract.html.

[18] Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo J. Kim, and Sung-Hoon Yoon. Perception prioritized training of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11472–11481, 2022. URL https://arxiv.org/abs/2204.00227.

[19] Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via min-snr weighting strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7441–7451, 2023. URL https://arxiv.org/abs/2303.09556.

[20] Jingfeng Yao, Cheng Wang, Wenyu Liu, and Xinggang Wang. Fasterdit: Towards faster diffusion transformers training without architecture modification. *arXiv preprint arXiv:2410.10356*, 2024. URL https://arxiv.org/abs/2410.10356.

[21] Kai Wang, Mingjia Shi, Yukun Zhou, Zekai Li, Zhihang Yuan, Yuzhang Shang, Xiaojiang Peng, Hanwang Zhang, and Yang You. A closer look at time steps is worthy of triple speed-up for diffusion model training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12934–12944, 2025. URL https://arxiv.org/abs/2405.17403.

[22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848. URL https://doi.org/10.1109/CVPR.2009.5206848.

[23] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 2256–2265, 2015. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

[24] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

[25] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

[26] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 8162–8171, 2021. URL https://proceedings.mlr.press/v139/nichol21a.html.

[27] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

[28] Nick Stracke, Stefan Andreas Baumann, Kolja Bauer, Frank Fundel, and Björn Ommer. Cleandift: Diffusion features without noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 117–127, 2025. URL https://www.computer.org/csdl/proceedings-article/cvpr/2025/436400a117/2999Z5KyfEA.

[29] Xingyu Zhou, Qifan Li, Xiaobin Hu, Hai Chen, and Shuhang Gu. Guiding a diffusion transformer with the internal dynamics of itself. *arXiv preprint arXiv:2512.24176*, 2025. URL https://arxiv.org/abs/2512.24176.

[30] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

[31] Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022. URL https://arxiv.org/abs/2209.15571.

[32] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[33] Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas Matthew 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 (ECCV)*, 2024. URL https://arxiv.org/abs/2401.08740.

[34] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W. Battaglia. Generating images with sparse representations. *arXiv preprint arXiv:2103.03841*, 2021. URL https://arxiv.org/abs/2103.03841.

[35] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016. URL https://arxiv.org/abs/1606.03498.

[36] Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lučić, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. *arXiv preprint arXiv:1806.00035*, 2018. URL https://arxiv.org/abs/1806.00035.

[37] 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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. URL https://arxiv.org/abs/1706.08500.

[38] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. URL https://arxiv.org/abs/2207.12598.

[39] Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018. URL https://arxiv.org/abs/1812.06162.

[40] Zihao Sun, Yu Sun, Longxing Yang, Shunong Lu, Jilin Mei, Wenxiao Zhao, and Yu Hu. Unleashing the power of gradient signal-to-noise ratio for zero-shot nas. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5740–5750, 2023. URL https://ieeexplore.ieee.org/document/10378069.

Appendix

Contents

A Additional Implementation Details **15**

A.1 EDM training and sampling. 15

A.2 Implementation of noise-aware routing. 15

A.3 Network Details 16

B Derivation of Route-Sensitivity Decomposition **16**

C Token-Selection Sensitivity Estimation **17**

D Fine-tuning without Routing **17**

E Generalization across Training Horizons and Tasks **18**

F Additional Statements and Limitations **18**

F.1 Limitations 18

F.2 Statement on Broader Impacts 18

F.3 Statement on Asset Usage, Licenses, and Terms of Use 18

G Additional Uncurated Samples **19**

A Additional Implementation Details

A.1 EDM training and sampling.

Our implementation follows the EDM [27] preconditioning formulation. During training, we sample $\log \sigma \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$, construct $x_\sigma = x + \sigma\epsilon$, and optimize the weighted denoising loss

$$\mathcal{L}_{\text{EDM}} = w(\sigma) \left\| D_\theta(x_\sigma, \sigma) - x \right\|_2^2, \qquad w(\sigma) = \frac{\sigma^2 + \sigma_{\text{data}}^2}{(\sigma \sigma_{\text{data}})^2}.$$

We use $P_{\text{mean}} = -1.2$, $P_{\text{std}} = 1.2$, and $\sigma_{\text{data}} = 0.5$ unless otherwise stated.

For sampling, we use the EDM sampler with a second-order Heun correction. The noise schedule is discretized as

$$t_i = \left(\sigma_{\max}^{1/\rho_{\text{EDM}}} + \frac{i}{N-1} \left(\sigma_{\min}^{1/\rho_{\text{EDM}}} - \sigma_{\max}^{1/\rho_{\text{EDM}}} \right) \right)^{\rho_{\text{EDM}}}, \qquad i = 0, \ldots, N-1,$$

followed by $t_N = 0$. We set $N = 40$, $\rho_{\text{EDM}} = 7$, $\sigma_{\max} = 80$, and $\sigma_{\min} = 0.002$. Here ρ_{EDM} denotes the EDM sampler exponent and should not be confused with the routing ratio ρ used in the main text. Furthermore, we use Heun's method as the ODE solver for EDM sampling. The 40-step EDM sampler provides an efficient evaluation protocol and has been shown to achieve FID scores [37] comparable to much longer DDPM [1] sampling schedules, such as 250 sampling steps. Therefore, compared with the ADM-style setup used in the original DiT training [4], the EDM formulation offers a simpler preconditioned denoising objective together with a more efficient sampling procedure.

A.2 Implementation of noise-aware routing.

Three-bin noise-aware routing. Noise-aware routing is implemented as a sample-wise routing-ratio schedule inside each mini-batch. Given the sampled noise level σ_m of sample m, we assign it to a noise region according to $\log \sigma_m$. Specifically, for region proportions $(\pi_{\text{low}}, \pi_{\text{mid}}, \pi_{\text{high}})$, we compute two quantile thresholds

$$\tau_1 = P_{\text{mean}} + P_{\text{std}} \Phi^{-1}(\pi_{\text{low}}), \qquad \tau_2 = P_{\text{mean}} + P_{\text{std}} \Phi^{-1}(\pi_{\text{low}} + \pi_{\text{mid}}),$$

where Φ^{-1} is the inverse CDF of the standard normal distribution. The sample is assigned to the low-, middle-, or high-noise region according to

$$b(m) = \begin{cases} \text{low}, & \log \sigma_m < \tau_1, \\ \text{mid}, & \tau_1 \le \log \sigma_m < \tau_2, \\ \text{high}, & \log \sigma_m \ge \tau_2. \end{cases}$$

The routing ratio is then selected as $\rho_m = \rho_{b(m)}$. In our default configuration, we use $(\pi_{\text{low}}, \pi_{\text{mid}}, \pi_{\text{high}}) = (0.3, 0.4, 0.3)$ and $(\rho_{\text{low}}, \rho_{\text{mid}}, \rho_{\text{high}}) = (0.5, 0.6, 0.7)$.

For a mini-batch with B samples and N latent tokens per sample, the implementation constructs a vector of sample-wise routing ratios $(\rho_1, \ldots, \rho_B)$. For sample m, the number of bypassed tokens is $\lfloor \rho_m N \rfloor$, and the number of active tokens processed by the routed segment is $N - \lfloor \rho_m N \rfloor$. Since different samples in the same mini-batch may have different routing ratios, their active-token counts may differ. To keep the computation batched, we gather active tokens up to the maximum active-token count in the mini-batch and use a valid-token mask to ignore padded positions.

At the start layer of a route $r_{i \to j}$, active tokens are gathered according to a random token permutation, while bypassed tokens are stored in the original token sequence. The routed blocks process only the gathered active tokens. At the end layer, the processed active tokens are scattered back to their original positions, while bypassed tokens are restored from the saved full token sequence. This implementation preserves the original inference architecture: routing is enabled only during training, and all tokens follow the full DiT path at inference time.

Continuous noise-aware routing. We additionally consider a continuous routing policy that maps the sampled noise level directly to a routing ratio:

$$\rho(\sigma) = \rho_{\min} + (\rho_{\max} - \rho_{\min}) \Phi \left(\frac{\log \sigma - P_{\text{mean}}}{P_{\text{std}}} \right), \tag{7}$$

where Φ denotes the standard normal CDF, $\rho_{\min} = 0.5$, and $\rho_{\max} = 0.7$. Under the log-normal noise distribution $\log \sigma \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$, the expected routing ratio remains $\mathbb{E}[\rho(\sigma)] = 0.6$, matching the discrete allocation strategies.

A.3 Network Details

Table 5 summarizes the backbone configurations used in our experiments. Noise-aware routing does not introduce any additional learnable parameters. It only changes the training-time routing gate sampling according to the noise condition, while keeping the backbone architecture, parameter count, and inference computation path unchanged.

Table 5: **Network details.** Noise-aware routing introduces no additional learnable parameters. The parameter counts of DiT [4] and DiT with noise-aware routing are identical for all backbone sizes.

Backbone	DiT Params	DiT + Noise-Aware Routing Params
DiT-S/2	33M	33M
DiT-B/2	130M	130M
DiT-L/2	458M	458M
DiT-XL/2	675M	675M

B Derivation of Route-Sensitivity Decomposition

In this section, we provide the formal derivation of the route-sensitivity decomposition presented in Eq. (5).

Problem Setup. For a fixed training sample and trajectory condition λ, let $J \in \mathbb{R}^{d\times N}$ denote the Jacobian matrix $J_{\mathbf{1},\lambda}$. The bypass indicator vector $S = (S_1, \ldots, S_N)^\top \in \{0,1\}^N$ consists of i.i.d. Bernoulli random variables such that $S_k \sim \mathrm{Bernoulli}(\rho)$. Consequently, the first and second moments of S are given by:

$$\mathbb{E}[S] = \rho\mathbf{1}, \quad \text{and} \quad \mathrm{Cov}(S) = \rho(1-\rho)I_N. \tag{8}$$

Derivation. We aim to evaluate the expected squared norm of the linearized discrepancy, $\mathbb{E}_S[\|JS\|_2^2]$. Using the property that $\|v\|_2^2 = v^\top v$ and the linearity of expectation, we have:

$$\mathbb{E}_S[\|JS\|_2^2] = \mathbb{E}_S[S^\top J^\top JS]. \tag{9}$$

By applying the property of the trace operator, $\mathbb{E}[x^\top Ax] = \mathrm{Tr}(A\mathbb{E}[xx^\top])$, and substituting $A = J^\top J$, we obtain:

$$\mathbb{E}_S[S^\top J^\top JS] = \mathrm{Tr}\left(J^\top J\,\mathbb{E}[SS^\top]\right). \tag{10}$$

The second moment matrix $\mathbb{E}[SS^\top]$ can be expanded via the relation $\mathbb{E}[SS^\top] = \mathrm{Cov}(S) + \mathbb{E}[S]\mathbb{E}[S]^\top$:

$$\mathbb{E}[SS^\top] = \rho(1-\rho)I_N + \rho^2\mathbf{1}\mathbf{1}^\top. \tag{11}$$

Substituting this back into the trace expression and utilizing the linearity of the trace, we arrive at:

$$\begin{aligned}\mathbb{E}_S[\|JS\|_2^2] &= \mathrm{Tr}\left(J^\top J\left[\rho(1-\rho)I_N + \rho^2\mathbf{1}\mathbf{1}^\top\right]\right)\\ &= \rho(1-\rho)\mathrm{Tr}(J^\top J) + \rho^2\mathrm{Tr}(J^\top J\mathbf{1}\mathbf{1}^\top).\end{aligned} \tag{12}$$

To simplify the terms in Eq. (12), we recall two standard matrix norm identities:

1. **Frobenius Norm:** $\mathrm{Tr}(J^\top J) = \sum_{i,j} J_{i,j}^2 = \|J\|_F^2$.
2. **Cyclic Property:** For the second term, the cyclic property of the trace ($\mathrm{Tr}(ABC) = \mathrm{Tr}(CAB)$) implies:

$$\mathrm{Tr}(J^\top J\mathbf{1}\mathbf{1}^\top) = \mathrm{Tr}(\mathbf{1}^\top J^\top J\mathbf{1}) = (J\mathbf{1})^\top(J\mathbf{1}) = \|J\mathbf{1}\|_2^2.$$

Combining these results, the final decomposition is:

$$\mathbb{E}_S[\|J_{\mathbf{1},\lambda}S\|_2^2] = \rho^2\|J_{\mathbf{1},\lambda}\mathbf{1}\|_2^2 + \rho(1-\rho)\|J_{\mathbf{1},\lambda}\|_F^2, \tag{13}$$

which recovers the result in Eq. (5).

C Token-Selection Sensitivity Estimation

To directly estimate the route-sensitivity term in our analysis, we use a Hutchinson estimator with Rademacher probes. For any Jacobian J, if v satisfies $\mathbb{E}[vv^\top] = I$, then

$$\mathbb{E}_v \|Jv\|_2^2 = \mathbb{E}_v \left[v^\top J^\top J v\right] = \mathrm{Tr}(J^\top J) = \|J\|_F^2.$$

We use Rademacher probes $v \sim \mathrm{Unif}\{-1, +1\}$, which satisfy this condition and provide an unbiased trace estimator.

In our setting, $J_{\mathbf{1},\lambda}$ denotes the Jacobian of the predictor with respect to the token-selection gate at the full-path gate $G = \mathbf{1}$. Instead of explicitly forming $J_{\mathbf{1},\lambda}$, we estimate the Jacobian-vector product by finite differences:

$$J_{\mathbf{1},\lambda} v \approx \frac{g_\theta(x_\lambda, \lambda; \mathbf{1} + \delta v) - g_\theta(x_\lambda, \lambda; \mathbf{1})}{\delta}.$$

This gives the loss-normalized token-selection sensitivity estimator

$$\widehat{\mathcal{S}}_{\mathrm{tok}} = \mathbb{E}_{x,\lambda,v}\left[w(\lambda) \frac{\|g_\theta(x_\lambda, \lambda; \mathbf{1} + \delta v) - g_\theta(x_\lambda, \lambda; \mathbf{1})\|_2^2}{\delta^2 d_{\mathrm{out}}}\right],$$

which estimates

$$\mathbb{E}_{x,\lambda}\left[w(\lambda) \frac{\|J_{\mathbf{1},\lambda}\|_F^2}{d_{\mathrm{out}}}\right].$$

We compute this estimate on DiT [4] checkpoints saved at 8k, 9k, and 10k training steps, and average the results across checkpoints. The resulting estimate is

$$\widehat{\mathcal{S}}_{\mathrm{tok}} = 0.0162.$$

This value is on the same order as the effective route-regularization coefficient obtained from the training-loss-gap fit in Section 4.4, where $\Delta\mathrm{loss} = 0.0212\rho$. Although the two quantities are not expected to match exactly, since the loss-gap coefficient is measured from independently trained fixed-ratio routing models whereas $\widehat{\mathcal{S}}_{\mathrm{tok}}$ is a local fixed-parameter sensitivity estimate, their comparable magnitude provides additional support for the route-sensitivity regularization view.

D Fine-tuning without Routing

Routing-based training can slightly affect the model's response to classifier-free guidance, since the model is optimized with a restricted training-time computation path. Following TREAD [14], we therefore apply an additional fine-tuning stage without routing to improve guided generation quality. During this stage, all tokens follow the full DiT path, while the model is fine-tuned with AdamW [32]using a batch size of 1,024, learning rate 1×10^{-5}, and $(\beta_1, \beta_2) = (0.9, 0.999)$.

Additionally, we report the FID [37] trajectory of DiT-B/2 [4] + Ours evaluated during training with CFG=1.5, without applying any post-training fine-tuning.

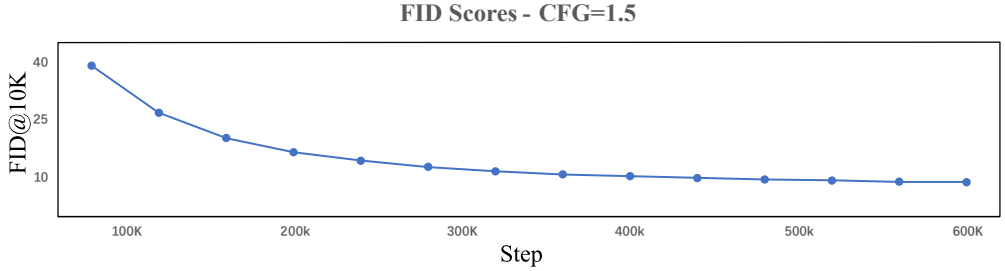


Figure 3: The model achieves an FID of 10.45 at 400K training steps and further improves to 8.90 at 600K steps, which is better than the reported CFG [38] FID of REPA [14] at 800K steps (9.04).

E Generalization across Training Horizons and Tasks

We provide additional evaluations to examine the effectiveness of noise-aware routing across extended training horizons and different generation tasks.

Table 6: **Generalization across training horizons and tasks.** (a) Extended DiT-XL/2 training on ImageNet-256 without routing-free fine-tuning. (b) Text-to-image generation on MS-COCO using DiT-B/2 trained for 400K iterations.

(a) Extended training

Metric	TREAD	Ours
Epochs to FID 4.96	160	**126**
Epochs to FID 4.77	240	**190**
FID at 240 epochs	4.77	**4.68**

(b) Text-to-image generation

Method	FID↓
Vanilla DiT	35.68
+TREAD	20.55
+Ours	**19.13**

Extended training. We extend DiT-XL/2 training from 160 to 240 epochs without routing-free fine-tuning. As shown in Table 6(a), our method requires approximately 21% fewer epochs to reach matched FID thresholds. At 240 epochs, it also achieves a lower FID than TREAD (4.68 vs. 4.77), indicating that the convergence advantage persists under longer training budgets.

Text-to-image generation. We further evaluate our method on MS-COCO text-to-image generation using DiT-B/2 trained for 400K iterations. As shown in Table 6(b), our method achieves an FID of 19.13, outperforming TREAD (20.55) and vanilla DiT (35.68). This result suggests that the benefits of noise-aware routing extend beyond ImageNet class-conditional generation.

F Additional Statements and Limitations

F.1 Limitations

Our work has several limitations. The proposed regularization view is based on a local first-order approximation and is not intended as a globally exact characterization of routing dynamics. Our experiments are mainly conducted on class-conditional ImageNet with DiT/SiT-style backbones, so broader validation remains future work. In addition, the current noise-aware routing schedule relies on predefined noise partitions and manually chosen routing ratios, and repeated-run evaluation is still limited by the high cost of full diffusion training.

F.2 Statement on Broader Impacts

Our work improves the training efficiency of diffusion transformers, thereby reducing computational cost and lowering the barrier to generative model research. This can make experimentation with large-scale generative models more accessible to a broader range of researchers. At the same time, improving the efficiency of image generation models may also facilitate harmful uses such as deepfake generation, synthetic misinformation, or large-scale production of deceptive visual content. In this work, we do not release a high-risk pretrained generative model, and we encourage responsible use consistent with existing safety policies and usage restrictions.

F.3 Statement on Asset Usage, Licenses, and Terms of Use

Our experiments use several existing public assets. We summarize the main assets, their roles, and their usage information below.

- **ImageNet** [22]. We use class-conditional ImageNet as the main benchmark dataset for training and evaluation. We use it only for research purposes and follow its official terms of use.
- **Pretrained VAE** [3]. Following the standard DiT setup, all models are trained in the latent space of a pretrained VAE. We use the publicly available pretrained VAE specified in our implementation and follow its original license and usage conditions.

- **DiT / SiT / TREAD baselines** [4, 33, 14]. We compare against and build on publicly available diffusion transformer baselines, including DiT, SiT, and TREAD, and credit their original papers and public resources in the main paper and bibliography. We follow the corresponding licenses and terms of use of these assets.
- **ADM TensorFlow evaluation suite** [2]. We compute evaluation metrics using the ADM TensorFlow evaluation suite and follow its original usage conditions.

We credit the original creators of all these assets in the main paper and bibliography, and we use these public datasets, models, and code resources in accordance with their respective licenses and terms of use.

G Additional Uncurated Samples



Figure 4: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.



Figure 5: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.



Figure 6: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.

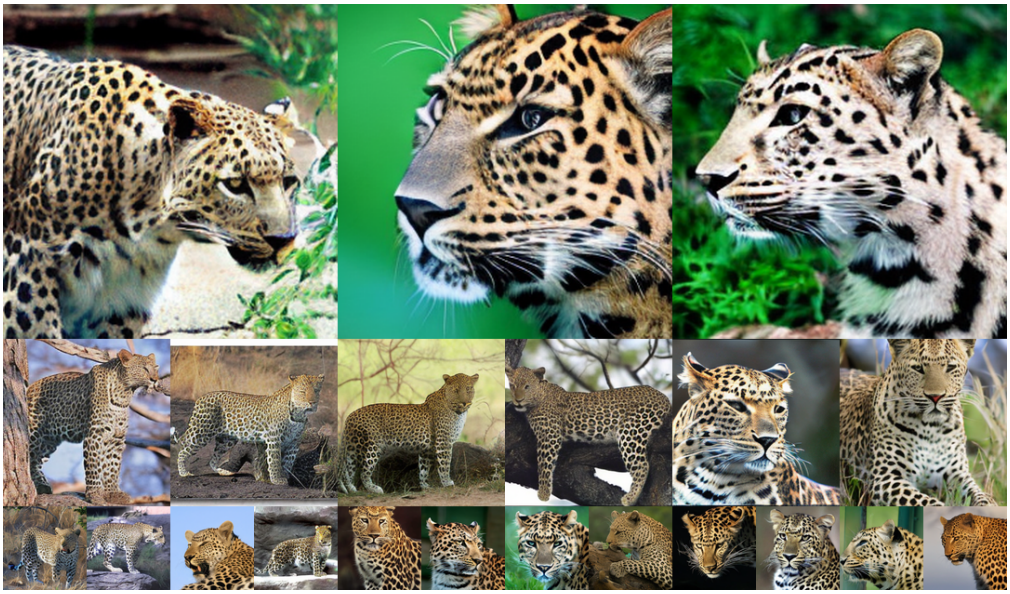


Figure 7: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.



Figure 8: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.



Figure 9: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.



Figure 10: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.



Figure 11: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.



Figure 12: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.



Figure 13: **Uncurated generation results of DiT-XL/2 with Noise-Aware Routing (F).** We use classifier-free guidance with $\omega = 4.0$.

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 abstract and introduction accurately reflect the paper's theoretical and experimental contributions.

   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: The limitations of the proposed method are discussed in Appendix E.

   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: [Yes]

   Justification: We provide detailed assumptions and derivations in the Method section.

   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: The paper provides the main model, training, and evaluation details needed to reproduce the reported results.

   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: [No]

   Justification: We do not release the implementation code, but we provide sufficiently detailed descriptions of the method, training setup, and evaluation protocol to enable reproduction of the main results.

   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: The paper specifies datasets, model settings, optimization details, and evaluation metrics.

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: Training large diffusion or flow models with multiple random seeds exceeds our computational resources and is uncommon in this research setting. We nevertheless repeated experiments on a subset of models and found that, under the same configuration with FID computed from 50k generated samples, the variation stayed within 0.1 and did not affect the main conclusions.

   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: [Yes]

   Justification: The paper reports the hardware and compute cost of the main experiments.

   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 are committed to complying with the NeurIPS Code of Ethics.

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: [Yes]

Justification: Potential positive and negative societal impacts are discussed in Appendix E.

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: [N/A]

Justification: We do not release a generative model or dataset as part of our submission.

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: The existing datasets, models, and code assets used in this work, together with their usage information, are described in Appendix E.

    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: [N/A]

    Justification: This paper does not release new assets as part of the submission.

    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: This paper does not involve crowdsourcing or 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: This paper does not involve crowdsourcing or 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: We do not use LLMs as an important, original, or non-standard component of the core methodology.

    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.