LENS: Low-Frequency Eigen Noise Shaping for Efficient Diffusion Sampling

Haewon Jeon **Si-Hyeon Lee**

School of Electrical Engineering
Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, South Korea
{haewon, sihyeon}@kaist.ac.kr

Abstract

Distilled diffusion models accelerate image generation by reducing the number of denoising steps, but often suffer from degraded image quality. To mitigate this trade-off, test-time optimization methods improve quality, yet their iterative nature incurs substantial computational overhead and leads to slow inference, limiting practical usability. Recent hypernetwork-based approaches amortize this process during training, but still require costly noise modulation in high-dimensional latent spaces. In this work, we propose **LENS** (Low-frequency Eigen Noise Shaping), an efficient noise modulation framework that operates in a low-dimensional subspace. Our approach is motivated by the observation that low-frequency components of the noise largely determine the global structure and visual fidelity of generated images. Based on this observation, we provide a theoretical justification for restricting modulation to the low-frequency subspace and derive a principled training objective. Building on this, LENS employs a lightweight, standalone network to selectively modulate these components, enabling efficient and targeted noise modulation. Extensive experiments demonstrate that LENS achieves competitive image quality while reducing FLOPs by 400–700×, model parameters by 25–75×, and inference-time overhead by 10–20× compared to prior methods. The project page is available at https://jeonhaewon.github.io/LENS-project/.

1 Introduction

Diffusion models have achieved remarkable success in high-quality text-to-image generation [32, 52]. These models generate images by iteratively denoising an initial noise sample, but achieving high fidelity typically requires a large number of sampling steps, resulting in high computational cost and slow inference [6, 36]. To improve efficiency, distilled diffusion models have been developed to enable generation in a single or a few steps [38, 48, 51]. However, this efficiency often comes at the expense of generation quality, leading to an inherent trade-off between speed and fidelity [47]. This motivates methods that enhance generation quality in distilled models without sacrificing their efficiency.

Recent work has explored inference-time techniques that improve generation quality by performing additional computations during inference while keeping model parameters fixed [40, 41]. Among these, a prominent direction is to optimize the initial noise, motivated by the observation that it governs the global structure of the generated image [2, 22, 25, 26, 50]. Conventional approaches perform test-time optimization, iteratively refining the noise for each sample [3, 7, 10, 18, 27, 28, 39, 43], which incurs substantial computational overhead.

To mitigate this, recent methods amortize the optimization by learning a prompt-conditioned mapping from noise to improved noise [1, 8, 23, 55]. HyperNoise [8] is a representative approach that

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

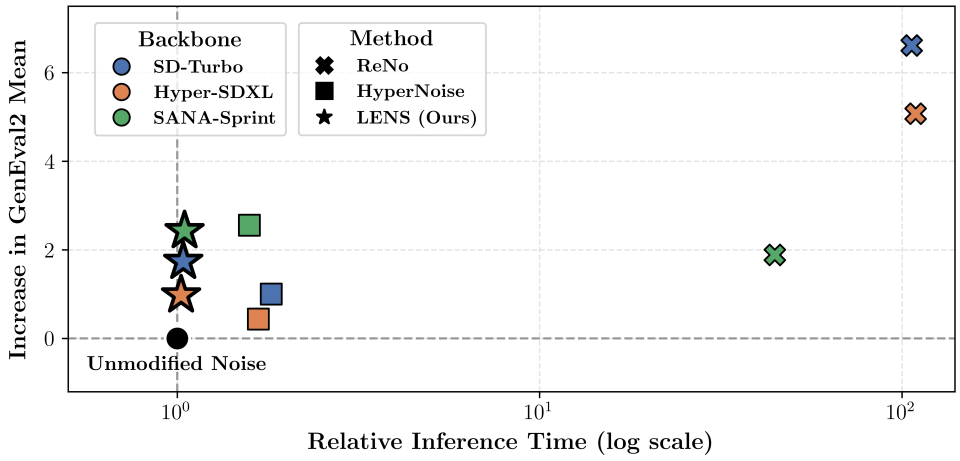


Figure 1: **Performance vs. inference time relative to unmodified noise input.** The x-axis shows inference time normalized by the case where the input noise is sampled directly from a standard Gaussian distribution without any modification, plotted in log scale. The y-axis shows the increase in GenEval2 [17] mean over this reference. Hence, the point $(1, 0)$ corresponds to this unmodified noise setting. We compare three approaches: ReNo [7], which performs iterative test-time optimization of the input noise; HyperNoise [8], which learns a noise modulation network using the diffusion backbone; and the proposed LENS. Results are shown across three distilled diffusion backbones: SD-Turbo [35], Hyper-SDXL [31], and SANA-Sprint [5]. LENS achieves comparable or improved quality relative to HyperNoise with reduced inference time, while ReNo attains higher quality at a substantially higher computational cost.

predicts noise perturbations by reusing the diffusion backbone augmented with lightweight adaptation modules. Specifically, the initial noise is first passed through the Low-Rank Adaptation (LoRA) [13]-augmented diffusion model to generate a modified noise representation, which is then fed into the original diffusion model for image synthesis. As a result, noise modulation requires executing the diffusion backbone itself, both during training (for learning the perturbation mapping) and at inference time (for each sample), leading to non-negligible computational overhead despite improvements over iterative optimization methods. Moreover, this process operates in the full high-dimensional latent space, introducing unnecessary degrees of freedom and further limiting efficiency.

Meanwhile, prior studies suggest that low-frequency components of the noise primarily determine global structure, whereas high-frequency components mainly affect fine details [44]. This insight indicates that effective noise modulation can be achieved in a low-dimensional subspace.

Motivated by these observations, we propose **Low-frequency Eigen Noise Shaping (LENS)**, a framework that performs efficient noise modulation by operating on low-frequency coefficients of the latent noise. By restricting modulation to this low-dimensional subspace, LENS enables the use of a lightweight network that is decoupled from the diffusion backbone, significantly reducing computational cost while preserving control over key structural attributes of generated images. As illustrated in Figure 1, LENS achieves a favorable performance–efficiency trade-off, maintaining competitive generation quality with negligible additional inference cost.

Our contributions are summarized as follows:

- We propose LENS, a novel framework for efficient noise modulation in distilled diffusion models via optimization in a low-frequency subspace. By eliminating the need to reuse the diffusion backbone and reducing the effective degrees of freedom during training, LENS enables more efficient and stable learning.
- We provide theoretical justification for focusing on low-frequency components and derive a principled training objective in the corresponding coefficient space.
- We demonstrate that LENS achieves comparable or improved generation quality while significantly reducing computational cost, with 400–700$\times$ fewer FLOPs, 25–75$\times$ fewer parameters, and 10–20$\times$ faster noise modulation.

2 Related Work

Distilled Diffusion Models. Diffusion-based generative models produce high-quality images by iteratively denoising an initial noise sample $z_0 \sim \mathcal{N}(0, I)$ [12, 37]. However, this process typically requires many sampling steps, resulting in high computational cost [34]. To address this limitation, recent advances in diffusion model distillation, including SD-Turbo [35], Hyper-SDXL [31], and SANA-Sprint [5], have drastically reduced the number of sampling steps. Despite these improvements, such efficiency gains often come at the expense of generation quality, with distilled models typically underperforming their full diffusion counterparts. This trade-off motivates inference-time techniques that improve generation quality while retaining the efficiency of distilled models.

Noise Hypernetworks. Recently, a novel approach has been proposed to improve generation quality by modulating the initial noise via a noise hypernetwork, enabling faster inference than existing iterative test-time optimization methods [3, 7, 10, 18]. Given a pretrained generator G_θ that maps an initial noise sample z_0 to an output image x, i.e., $x = G_\theta(z_0)$, let $r(x)$ be a reward function that quantifies the quality of generated images. HyperNoise [8] considers a reward-tilted noise distribution $p^\star$ that biases sampling toward noise realizations yielding high-reward outputs:

$$p^\star(z_0) \propto p(z_0) \exp\big(r(G_\theta(z_0))\big), \tag{1}$$

where p denotes the original noise distribution. HyperNoise trains a noise hypernetwork f_ϕ with parameters ϕ such that the distribution of the modulated noise sample $z_0 + f_\phi(z_0)$, denoted by p^ϕ, is close to the target distribution $p^\star$. Accordingly, f_ϕ is trained to minimize the KL divergence $D_{\mathrm{KL}}(p^\phi \| p^\star)$, which can be approximated (up to a constant) under mild conditions by the following loss [8]:

$$\mathcal{L}(\phi) = \mathbb{E}_{z_0 \sim p}\left[\frac{1}{2}\|f_\phi(z_0)\|^2 - r(G_\theta(z_0 + f_\phi(z_0)))\right], \tag{2}$$

which consists of a regularization term that penalizes deviation from the original noise distribution and a reward maximization term that encourages high-quality outputs. This formulation amortizes the search for high-reward noise into a learned mapping.

However, noise modulation remains computationally inefficient in both training and inference, since each forward pass requires executing a large-scale diffusion backbone. Furthermore, operating in the full latent noise space introduces unnecessary degrees of freedom, leading to reduced efficiency.

Patch-wise PCA Analysis of Diffusion Noise. Recently, an intriguing result has demonstrated a strong correlation between the PCA components of initial noise in diffusion models and the structure of the generated outputs [44]. In this study, the noise is decomposed into $p \times p$ patches and projected onto a PCA basis derived from natural image statistics. It is shown that the top principal components (i.e., low-frequency components) largely determine key attributes of the generated images, such as global structure and object layout. In contrast, modifying high-frequency components has minimal impact on these structural properties. This observation motivates a more efficient noise modulation strategy that focuses solely on shaping the low-frequency components.

3 Method

We propose a novel framework, **Low-frequency Eigen Noise Shaping (LENS)**, for efficient diffusion sampling via noise modulation, which operates on the leading PCA subspace of the initial noise.[1] LENS employs a lightweight network operating on low-frequency noise coefficients to predict modulation, without relying on the diffusion backbone. It improves generation quality while maintaining low computational overhead in both training and inference.

3.1 Theoretical Foundations and Objective

In the following, we introduce our method and provide theoretical justification based on prior findings that the low-frequency components of the initial noise predominantly determine the structural

[1] Following prior observations that leading PCA components of natural-image patches tend to exhibit smoother, lower-spatial-frequency patterns [16, 24], we refer to this subspace as "low-frequency," while not implying that the PCA basis itself is a Fourier basis. We further verify this interpretation for the PCA bases used by LENS in Appendix D.1.

properties of the generated outputs [44]. Building on the reward-tilted distribution and KL-based learning framework of HyperNoise [8], we extend the formulation to noise modulation restricted to the low-frequency PCA subspace. We consider a pretrained diffusion generator G_θ that maps an initial noise sample $z_0 \in \mathbb{R}^{C\times H\times W}$ to an output image x, i.e., $x = G_\theta(z_0)$, where C, H, and W denote the number of channels, height, and width, respectively. We assume $z_0 \sim \mathcal{N}(0, I)$. Let $r(x)$ be a reward function that quantifies the quality of generated images. In practice, we also condition on a text prompt, and $r(x)$ reflects the alignment of x with the given prompt.

Noise Reparameterization in the PCA Coefficient Space. We partition the initial latent noise z_0 into N non-overlapping patches $\{s_i\}_{i=1}^N$, where each patch $s_i \in \mathbb{R}^d$ with $d = Cs^2$, is obtained by extracting a $s \times s$ spatial region across all channels and vectorizing it. Hence, $N = \frac{HW}{s^2}$.

Each patch is projected onto a PCA basis $V \in \mathbb{R}^{d\times d}$ computed from natural image statistics, resulting in the coefficient vector $w_i = V^\top s_i \in \mathbb{R}^d$. Since V is an orthonormal matrix, this transformation preserves the Gaussian prior, i.e., $w_i \sim \mathcal{N}(0, I)$. A detailed proof is provided in Appendix A.2. We further decompose the coefficient vector as $w_i = (w_{i,L}, w_{i,H})$, where $w_{i,L} \in \mathbb{R}^k$ denotes the low-frequency components and $w_{i,H} \in \mathbb{R}^{d-k}$ denotes the high-frequency components, with $k < d$. Let $w \in \mathbb{R}^{N\times d}$, $w_L \in \mathbb{R}^{N\times k}$, and $w_H \in \mathbb{R}^{N\times(d-k)}$ denote the collections $\{w_i\}_{i=1}^N$, $\{w_{i,L}\}_{i=1}^N$, and $\{w_{i,H}\}_{i=1}^N$, respectively. Then, their corresponding prior distributions, denoted by $q(w)$, $q_L(w_L)$, and $q_H(w_H)$, all follow isotropic Gaussian distributions $\mathcal{N}(0, I)$ with appropriate dimensions, as shown in Appendix A.3.

Low-frequency-only Tilted Distribution. Let $\textbf{Recon}(w)$ denote the reconstruction operator that maps patch-wise PCA coefficients to the full latent noise. The generator can be expressed in the coefficient space as $g(w) := G_\theta(\textbf{Recon}(w))$. Following the reward-tilted formulation of HyperNoise [8], we define the target distribution in the PCA coefficient space as

$$q^\star(w) \propto q(w) \exp\big(r(g(w))\big). \tag{3}$$

Prior empirical results from patch-wise PCA analysis [44] show that perturbations applied to the low-frequency components of the noise have a dominant effect on the composition of generated images. Based on this observation, we make the following assumption, which is empirically validated in Section 4.2 and Appendix D.2.

Assumption 1. *For any (w_L, w_H),*

$$\big|r\big(g(w_L, w_H)\big) - \bar{r}(w_L)\big| \le \epsilon, \tag{4}$$

where $\bar{r}(w_L) := \mathbb{E}_{w_H\sim q_H}[\, r(g(w_L, w_H))\,]$ denotes the expected reward conditioned on the low-frequency coefficients and $\epsilon > 0$ is a small constant that bounds the contribution of the high-frequency components to the reward.

Assumption 1 motivates restricting modulation to the low-frequency coefficients w_L while leaving w_H unchanged. The key challenge is to establish whether such a low-frequency-only restriction can still approximate the full reward-tilted target distribution $q^\star(w_L, w_H)$. Although the Gaussian prior factorizes as $q(w_L, w_H) = q_L(w_L)q_H(w_H)$, as shown in Appendix A.4, the reward $r(g(w_L, w_H))$ generally couples w_L and w_H, and hence $q^\star(w_L, w_H)$ does not generally factorize. Therefore, leaving w_H distributed according to its original prior q_H is not, by itself, guaranteed to preserve the full target distribution. To justify this restriction, we define the low-frequency-only tilted distribution as

$$\tilde{q}^\star(w) := \tilde{q}^\star_L(w_L)\, q_H(w_H), \quad \tilde{q}^\star_L(w_L) \propto q_L(w_L) \exp(\bar{r}(w_L)). \tag{5}$$

The following proposition shows that, under Assumption 1, this low-frequency-only distribution remains close to the full target distribution $q^\star(w)$ in KL divergence. The proof is provided in Appendix A.5.

Proposition 3.1. *The KL divergence between the full tilted distribution $q^\star(w)$ and its low-frequency-only tilted distribution $\tilde{q}^\star(w)$ is bounded as*

$$D_{\mathrm{KL}}(q^\star \parallel \tilde{q}^\star) \le 2\epsilon. \tag{6}$$

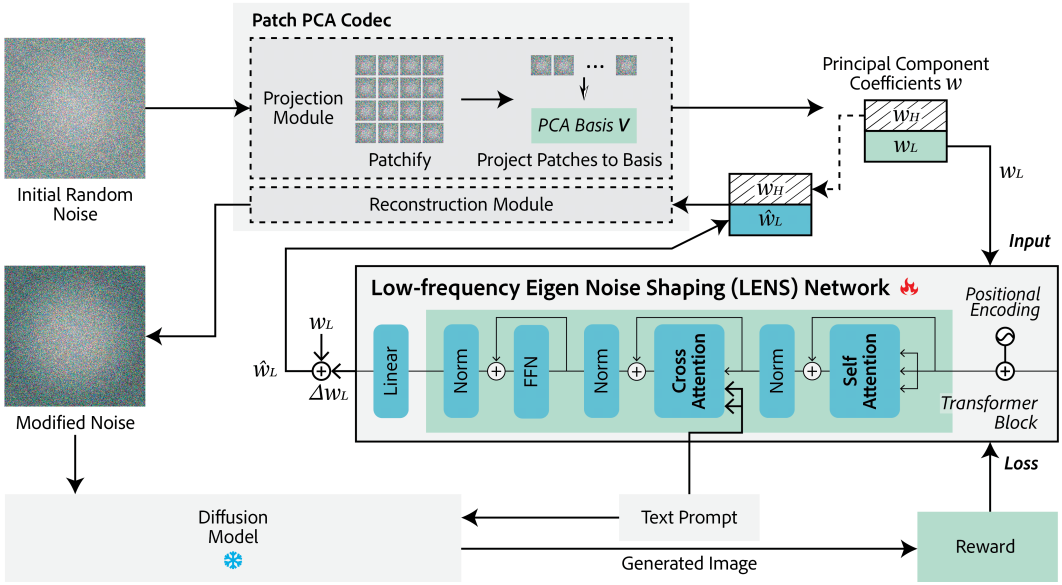


Figure 2: Overview of the proposed **Low-frequency Eigen Noise Shaping (LENS)** framework. The model operates on low-frequency noise coefficients that capture structurally meaningful components. These coefficients are processed via self-attention, while the text prompt is incorporated through cross-attention for conditioning, and the network predicts coefficient updates.

Objective. We train a neural network h_ϕ with parameters ϕ that takes w_L as input and outputs an additive perturbation to w_L. In practice, h_ϕ is also conditioned on a text prompt, but we omit this dependence for notational simplicity. Let q_L^ϕ denote the distribution of the modulated low-frequency coefficients $w_L + h_\phi(w_L)$. The network h_ϕ is trained so that q_L^ϕ approximates the target distribution $\tilde{q}_L^\star$. To this end, we minimize the KL divergence $D_{\mathrm{KL}}(q_L^\phi \| \tilde{q}_L^\star)$. The following proposition shows that this objective admits a tractable reformulation. The proof is provided in Appendix A.6.

Proposition 3.2 (Objective)**.** *Assume that h_ϕ is continuously differentiable with $\|J_{h_\phi}(w_L)\|_2 \leq M < 1$ for any w_L, where $J_{h_\phi}(w_L) := \frac{\partial h_\phi(w_L)}{\partial w_L}$ denotes the Jacobian matrix, $\|\cdot\|_2$ is the spectral norm, and M is a constant. Then minimizing the KL divergence $D_{\mathrm{KL}}(q_L^\phi \| \tilde{q}_L^\star)$ is equivalent (up to a constant) to minimizing*

$$D_{\mathrm{KL}}(q_L^\phi \| q_L) - \mathbb{E}_{w_L \sim q_L^\phi}[\bar{r}(w_L)], \tag{7}$$

which can be approximated (up to ϵ) by

$$\mathcal{L}(\phi) := \mathbb{E}_{w \sim q}\left[\frac{1}{2}\|h_\phi(w_L)\|^2 - r\big(g(w_L + h_\phi(w_L), w_H)\big)\right]. \tag{8}$$

3.2 Implementation

We present the practical implementation of the proposed framework. The framework comprises two components: a Patch PCA Codec and a LENS Network. The Patch PCA Codec projects noise onto a PCA basis to obtain coefficients and reconstructs them back to the noise space, while the LENS Network models prompt-conditioned modulation in the low-frequency coefficient space.

To construct the PCA basis, we precompute it from natural image datasets [44], such as ImageNetV2 [30]. Images are encoded into latent representations using a pretrained Variational Autoencoder (VAE) [19] of the diffusion model, and the resulting latents are partitioned into non-overlapping $s \times s$ patches. Each patch is reshaped into a vector in $\mathbb{R}^d$, and PCA is performed by computing the covariance matrix over all patches, yielding an orthonormal basis $V \in \mathbb{R}^{d \times d}$ whose columns correspond to eigenvectors.

The overall architecture and training pipeline are illustrated in Figure 2. We next describe the Patch PCA Codec and the LENS Network in detail.

Patch PCA Codec. This component consists of two modules: a projection module **Proj** and a reconstruction module **Recon**. The projection module partitions the input latent noise z into non-overlapping patches and projects them onto the precomputed PCA basis to obtain coefficient vectors w, i.e., $w = \textbf{Proj}(z)$. After coefficient modulation, the reconstruction module maps the coefficients w back to the latent noise z via the inverse PCA transformation, i.e., $z = \textbf{Recon}(w)$.

Low-frequency Eigen Noise Shaping (LENS) Network. The LENS network h_ϕ takes low-frequency coefficients w_L and a text prompt c as inputs, and predicts an additive perturbation to w_L that aligns the generated output with the semantics of the prompt. Operating on low-frequency coefficients reduces the dimensionality of the prediction space, enabling a lightweight network decoupled from the diffusion backbone. We implement this network using a transformer architecture [42] to model global dependencies and incorporate text conditioning. The network consists of an input projection layer, sinusoidal positional embeddings, multiple transformer blocks (including self-attention, text cross-attention, and feed-forward networks), and a final linear layer. Self-attention captures global correlations among coefficient tokens across patches, while text cross-attention aligns the modulation with the semantic content of the prompt. Further architectural details are provided in Appendix B.1.

The network is pretrained and used once at inference time to modulate the low-frequency noise. The training procedure is summarized in Algorithm 1. We use Eq. (8) as the training loss $\mathcal{L}(\phi)$, which consists of a regularization term and a reward term. The former acts as an L_2 penalty on the coefficient residual, encouraging proximity to the original Gaussian prior. The latter is a weighted combination of multiple reward models, including CLIP [11], HPSv2.1 [46], ImageReward [49], and PickScore [20]. Additional details on the reward formulation are provided in Appendix B.2.

Algorithm 1 Training of LENS Network

Initialize: LENS network parameters ϕ
while learning **do**
  Sample latent noise $z_0 \sim \mathcal{N}(0, I)$
  $(w_L, w_H) = \textbf{Proj}(z_0)$
  $\hat{w} \leftarrow (w_L + h_\phi(w_L, c),\ w_H)$
  $x \leftarrow G_\theta(\textbf{Recon}(\hat{w}))$
  Compute reward $r(x, c)$ and loss $\mathcal{L}(\phi)$
  Update ϕ using gradient descent
end while
return trained parameters ϕ

3.3 Complexity Analysis

We analyze the computational complexity of our transformer-based noise modulation framework, LENS. The computational complexity of a standard transformer is $\mathcal{O}(n^2 r + n r^2)$, where n is the number of tokens and r is the representation dimension [42]. The first term corresponds to self-attention, while the second term accounts for the feed-forward network. In LENS, the number of tokens is N (the number of patches), and each token has dimension k (the number of low-frequency PCA coefficients). As a result, the computational complexity becomes $\mathcal{O}(N^2 k + N k^2)$. We omit the cost of cross-attention, which scales with the prompt length, as it is negligible compared to the self-attention term given that the prompt is significantly shorter than the number of tokens.

Compared to operating in the full latent dimension d, this reduces the self-attention and feed-forward complexities by factors of d/k and $(d/k)^2$, respectively, yielding substantial computational savings. The efficiency of our method is experimentally validated in Section 4.2.

4 Experiment

4.1 Experimental Settings

Implementation. We conduct experiments using three distilled text-to-image diffusion models: SD-Turbo [35], Hyper-SDXL [31], and SANA-Sprint [5]. For each model, we precompute a corresponding PCA basis using its pretrained VAE encoder. We set the patch size to $s = 4$. For SD-Turbo and Hyper-SDXL, the latent channel dimension is $C = 4$, resulting in $d = 64$, and we use $k = 32$ low-frequency coefficients for modulation. For SANA-Sprint, the latent channel dimension is $C = 32$, resulting in $d = 512$, and we set $k = 256$. The effects of k and s on performance are analyzed in Appendices D.3 and D.4, respectively. All experiments are conducted on NVIDIA RTX 3090 GPUs. Further implementation details are provided in Appendix C.

Table 1: Results on GenEval2. Higher is better (↑) for all metrics except Time. LENS achieves performance comparable to or better than HyperNoise while enabling faster generation. ReNo is shown in gray, as it relies on iterative test-time optimization, incurring substantially higher computational cost and making it not directly comparable to feed-forward methods. Standard deviations are reported in Appendix D.11.

Model	Time(s) ↓	Object ↑	Attribute ↑	Count ↑	Position ↑	Verb ↑	Mean ↑
SD-Turbo	0.107	72.27	42.65	21.15	28.1	5.60	33.95
+ ReNo	11.371	81.02	50.42	31.91	32.58	6.90	40.57
+ HyperNoise	0.194	74.94	43.00	23.95	27.67	5.23	34.96
+ LENS (Ours)	**0.112**	**75.07**	**43.08**	**26.08**	**28.46**	**5.78**	**35.69**
Hyper-SDXL	0.414	71.94	51.97	26.88	21.14	3.28	35.04
+ ReNo	45.13	80.06	56.79	32.61	27.53	3.59	40.12
+ HyperNoise	0.694	71.97	52.02	27.06	**21.58**	**4.76**	35.48
+ LENS (Ours)	**0.424**	**72.96**	**53.47**	**27.84**	21.45	4.39	**36.02**
SANA-Sprint	0.336	80.23	54.73	33.01	33.31	15.53	43.36
+ ReNo	14.964	83.16	56.57	34.73	38.36	13.44	45.25
+ HyperNoise	0.532	**83.59**	54.18	34.84	36.32	**20.66**	**45.92**
+ LENS (Ours)	**0.352**	82.98	**55.27**	**36.12**	**36.38**	18.21	45.79

Dataset. For PCA basis extraction, we use ImageNetV2 [30]. During training, we use approximately 70k text prompts collected from multiple sources, including Attribute Binding (ABC-6K) [9], the T2I-CompBench training set [15], and Pick-a-Pic v2 [20].

Baselines. We compare our method against three baselines: random noise initialization (unmodified noise input), ReNo [7], and HyperNoise [8]. Random noise initialization samples the initial latent from a standard Gaussian distribution without additional processing. ReNo performs reward-guided test-time optimization, iteratively updating the initial noise via gradient-based optimization to maximize a given reward function. HyperNoise trains a noise hypernetwork by attaching LoRA adapters to the diffusion backbone, using it to predict a modulated initial noise that is then fed into the generator.

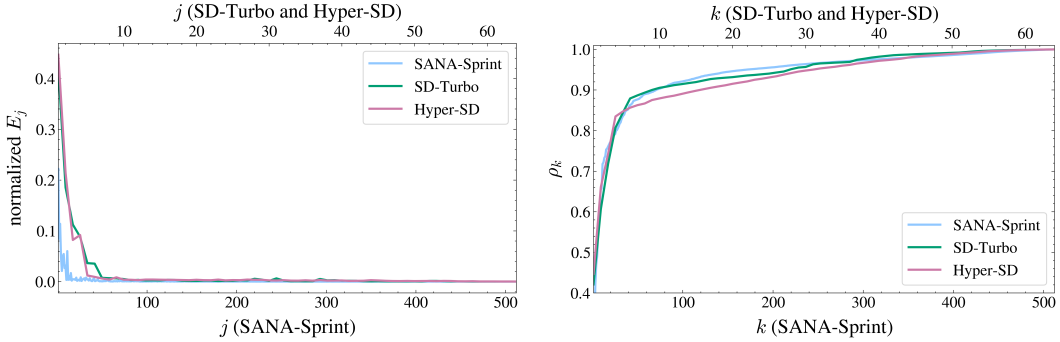
4.2 Experimental Results

Generation Quality. We evaluate the quality of generated images on the GenEval2 benchmark [17], with results reported in Table 1. All images are generated using a single diffusion step. Our method, LENS, achieves comparable or better performance than HyperNoise while reducing the additional time cost by a factor of 10–20. More specifically, HyperNoise incurs additional computational cost of approximately 50%–80% of the image generation cost, as the diffusion backbone must be executed once more during noise modulation. In contrast, our method introduces less than 5% overhead. Although ReNo attains the best performance in many cases, it requires 40–100$\times$ longer generation time than our method, making it impractical for real-world applications. We analyze performance as the number of inference steps increases in Appendix D.6. Additional results on other benchmarks are provided in Appendix D.7.

Computational Efficiency. In Table 2, we evaluate the computational efficiency of our method and HyperNoise [8], a representative approach that learns a noise modulation network. We report FLOPs for the noise-modulation pathway only, excluding full image generation. For HyperNoise, this includes a forward pass through the diffusion backbone (including LoRA branches), whereas for LENS, it includes PCA projection/reconstruction and a forward pass of the LENS network. We also report the number of trainable parameters in the modulation network, counting LoRA parameters for HyperNoise and all parameters of the LENS network for our method. Inference time is measured as the time required for noise modulation, averaged over 100 samples. Our method achieves substantial efficiency gains, reducing FLOPs by 400–700$\times$, the number of parameters by 25–75$\times$, and inference-time latency by 10–20$\times$ across all evaluated models.

Table 2: Noise modulation efficiency comparison between LENS and HyperNoise across different diffusion models. LENS consistently achieves lower FLOPs, fewer parameters, and faster inference time, resulting in significantly improved computational efficiency.

Model	Method	FLOPs	Parameters	Inference-time
SD-Turbo	HyperNoise	997.1 G	136.5 M	70.78 ms
	LENS (Ours)	2.4 G	4.8 M	5.68 ms
Hyper-SDXL	HyperNoise	7.8 T	395.3 M	207.91 ms
	LENS (Ours)	12.3 G	5.3 M	9.83 ms
SANA-Sprint	HyperNoise	1.7 T	314.7 M	173.62 ms
	LENS (Ours)	2.2 G	10.0 M	10.84 ms



(a) Reward gradient energy spectrum in the PCA basis

(b) Cumulative gradient energy ratio

Figure 3: Reward gradient energy distribution in the PCA basis. (a) Normalized energy spectrum $E_j / \sum_j E_j$, showing a rapid decay toward higher-frequency components. (b) Cumulative energy ratio ρ_k, indicating that most energy is concentrated in low-frequency components.

Low-Frequency Concentration of Reward Gradients. In this experiment, we validate the key hypothesis that reward-increasing directions are concentrated in a low-frequency PCA subspace rather than the full high-dimensional noise space. To this end, given an initial noise z_0 and a prompt c, we compute the reward gradient $\nabla_{z_0} r\big(G_\theta(z_0, c)\big)$, which indicates the direction in which the noise should be perturbed to maximally increase the reward. Then, we reshape the reward gradient into patch-wise vectors $\{u_\ell\}$ and project each onto the PCA basis $V = [v_1, \ldots, v_d] \in \mathbb{R}^{d \times d}$, where v_j denotes the j-th eigenvector, yielding coefficients $\alpha_{j,\ell} = v_j^\top u_\ell$. The importance of each direction is quantified by the expected gradient energy $E_j = \mathbb{E}_\ell[\alpha_{j,\ell}^2]$, where the expectation is taken over spatial patches and 100 sampled images. The cumulative energy ratio captured by the top-k components is then defined as $\rho_k = \frac{\sum_{j=1}^{k} E_j}{\sum_{j=1}^{d} E_j}$.

Figure 3(a) shows that the normalized E_j values decay rapidly as j increases, indicating that high-frequency components contribute little to the overall gradient energy. Figure 3(b) further shows that ρ_k quickly approaches a high value even for small k. These results confirm that the reward gradient is largely concentrated in a low-dimensional PCA subspace.

Generation Diversity. To evaluate generation diversity, we generate images for 300 prompts from the GenEval2 benchmark using 50 different random noise seeds, and compute LPIPS [53] and DINOv2 [29] scores. LPIPS measures perceptual differences between images, where higher values indicate greater visual diversity. DINO, on the other hand, captures feature-level similarity, where lower scores correspond to lower semantic similarity and thus higher semantic diversity.

Table 3 reports LPIPS and DINOv2 scores for unmodified noise, HyperNoise, and LENS. The results show that LENS achieves lower LPIPS than unmodified noise, indicating a moderate reduction in pixel-level diversity, while still maintaining higher diversity than HyperNoise. Under the DINO

Table 3: Generation diversity measured by LPIPS and DINO. LENS achieves intermediate LPIPS between random noise and HyperNoise, while attaining the lowest DINO score among all methods. Values are reported as mean ± standard deviation.

Method	LPIPS ↑	DINO ↓
SD-Turbo	**0.6249 ± 0.0427**	0.7531 ± 0.1107
+ HyperNoise	0.5406 ± 0.0562	0.8120 ± 0.1047
+ LENS (Ours)	0.5887 ± 0.0404	**0.7337 ± 0.1221**

Table 4: Out-of-distribution prompt evaluation on OneIG-Bench. LENS achieves the best performance on both Style and Reasoning subsets without any retraining or adaptation.

Method	Style ↑	Reasoning ↑
SD-Turbo	0.3360	0.1907
+ HyperNoise	0.3190	0.1862
+ LENS (Ours)	**0.3474**	**0.1946**

metric, LENS attains lower scores than unmodified noise, suggesting improved semantic diversity. These results demonstrate that LENS preserves a favorable level of diversity while enhancing semantic variation.

Out-of-Distribution Prompt Evaluation. To evaluate the out-of-distribution (OOD) generalization of LENS, we additionally test on the Anime & Stylization and Knowledge & Reasoning subsets of OneIG-Bench [4], which contain prompts substantially different from those used to train the modulation network. Using SD-Turbo, we compare unmodified noise, HyperNoise, and LENS under the standard OneIG-Bench evaluation protocol [4]. In this experiment, both the PCA basis and the trained modulation network are applied without any retraining or adaptation.

As shown in Table 4, LENS achieves the highest scores on both the Style and Reasoning subsets, outperforming unmodified noise and HyperNoise. These results suggest that the learned modulation generalizes well to the evaluated OOD prompts and remains effective even when applied to prompt categories that differ substantially from those seen during training. This further indicates that the benefit of LENS is not limited to the training prompt distribution.

Qualitative Results. Figure 4 illustrates one-step generation results from SD-Turbo and SANA-Sprint. All samples for each prompt are generated from the same initial noise to ensure a fair comparison. Without noise modification, the generated images often exhibit poor alignment with the given prompts. In contrast, both HyperNoise and LENS improve semantic consistency and overall visual quality. However, HyperNoise occasionally exhibits inconsistencies with key prompt attributes, whereas LENS generates images that remain more faithful to the intended semantics. LENS further enhances structural clarity and overall generation quality, demonstrating more reliable high-quality generation in the one-step setting.

More Experiments. We provide additional experiments in Appendix D. Specifically, Appendices D.3 and D.4 present ablation studies on the number of low-frequency coefficients k and the patch size s, justifying our design choices. Appendix D.5 examines the effect of the number of transformer layers in the LENS network, showing that increasing model depth does not necessarily improve performance. Appendix D.6 analyzes performance as the number of inference steps increases, complementing the one-step focus of the main paper, and shows that the distilled models are optimized for few-step generation. Appendix D.7 reports results on additional benchmarks, confirming the robustness of our method across different settings. Appendix D.8 evaluates aesthetic quality, demonstrating favorable overall visual quality. Appendices D.9 and D.10 provide ablations on reward components and PCA dataset sensitivity, showing that the full reward combination performs

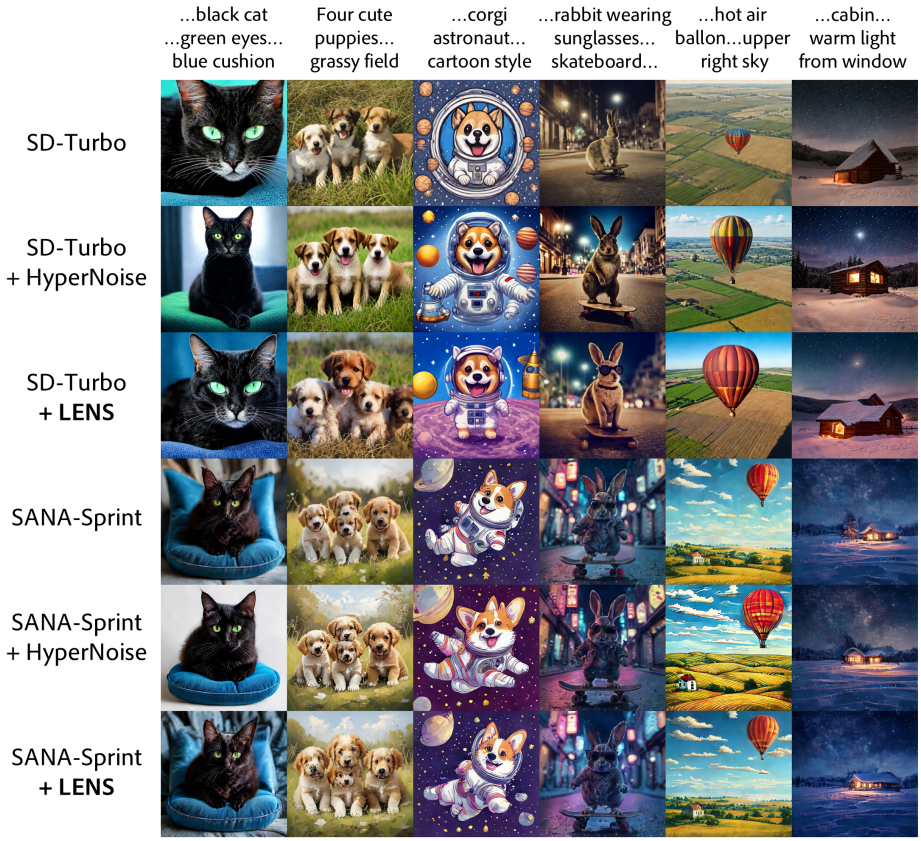


Figure 4: One-step generation results using SD-Turbo and SANA-Sprint. Starting from the same initial noise, we compare outputs generated with unmodified noise, HyperNoise, and LENS.

best and that LENS remains effective across different PCA basis datasets. Finally, standard deviations for the results in Table 1 are provided in Appendix D.11.

5 Conclusion

In this paper, we proposed LENS, a principled and efficient framework for noise modulation in diffusion models. By leveraging the insight that low-frequency components of the noise largely determine the global structure of generated images, we restrict modulation to a low-dimensional subspace and derive a corresponding training objective. This design enables effective improvement in generation quality while significantly reducing computational overhead. Extensive experiments across multiple distilled diffusion backbones demonstrate that LENS consistently achieves competitive or improved image quality with minimal inference-time overhead. These results establish low-dimensional noise shaping as an effective approach for improving diffusion-based generation, offering a practical pathway toward more efficient and scalable generative models. A discussion of the broader impact is provided in Appendix E.

Limitations and Future Work. Our approach relies on a fixed PCA basis computed from a reference dataset, which may not fully capture data distributions that differ significantly from the training domain. Adapting the basis to new domains or learning it jointly with the modulation network could further improve generalization. In addition, LENS focuses on low-frequency components of the noise, which are primarily responsible for global structure. Although this design enables efficient modulation, it may limit the ability to refine fine-grained details governed by high-frequency components. Extending the framework to selectively incorporate higher-frequency information in a controlled manner is an interesting direction for future work.

Acknowledgments and Disclosure of Funding

This work was supported in part by the IITP (Institute of Information & Communications Technology Planning & Evaluation)-ITRC (Information Technology Research Center) grant funded by the Korea government (MSIT) (IITP-2026-RS-2020-II201787, contribution rate 10%), in part by the IITP through the Next Generation Semantic Communication Network Research Center grant funded by Korean Government (MSIT) (RS-2024-00398948, contribution rate 20%), and in part by the IITP grant funded by the Korea Media and Communications Commission (KMCC) (No. RS-2026-25514346, Persona AI Model Technology Based on Accumulated Media Consumption Histories, contribution rate 70%).

References

[1] Donghoon Ahn, Jiwon Kang, Sanghyun Lee, Jaewon Min, Minjae Kim, Wooseok Jang, Hyoungwon Cho, Sayak Paul, SeonHwa Kim, Eunju Cha, et al. A noise is worth diffusion guidance. In *International Conference on Learning Representations*, 2026.

[2] Yuanhao Ban, Ruochen Wang, Tianyi Zhou, Boqing Gong, Cho-Jui Hsieh, and Minhao Cheng. The crystal ball hypothesis in diffusion models: Anticipating object positions from initial noise. In *International Conference on Learning Representations*, 2025.

[3] Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-flow: Differentiating through flows for controlled generation. In *International Conference on Machine Learning*, 2024.

[4] Jingjing Chang, Yixiao Fang, Peng Xing, Shuhan Wu, Wei Cheng, Rui Wang, Xianfang Zeng, Gang Yu, and Hai-Bao Chen. OneIG-Bench: Omni-dimensional nuanced evaluation for image generation. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[5] Junsong Chen, Shuchen Xue, Yuyang Zhao, Jincheng Yu, Sayak Paul, Junyu Chen, Han Cai, Song Han, and Enze Xie. SANA-Sprint: One-step diffusion with continuous-time consistency distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16185–16195, 2025.

[6] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, 2024.

[7] Luca Eyring, Shyamgopal Karthik, Karsten Roth, Alexey Dosovitskiy, and Zeynep Akata. Reno: Enhancing one-step text-to-image models through reward-based noise optimization. In *Advances in Neural Information Processing Systems*, volume 37, pages 125487–125519, 2024.

[8] Luca Eyring, Shyamgopal Karthik, Alexey Dosovitskiy, Nataniel Ruiz, and Zeynep Akata. Noise hypernetworks: Amortizing test-time compute in diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pages 95429–95462, 2025.

[9] Weixi Feng, Xuehai He, Tsu-Jui Fu, Varun Jampani, Arjun Reddy Akula, Pradyumna Narayana, Sugato Basu, Xin Eric Wang, and William Yang Wang. Training-free structured diffusion guidance for compositional text-to-image synthesis. In *International Conference on Learning Representations*, 2023.

[10] Xiefan Guo, Jinlin Liu, Miaomiao Cui, Jiankai Li, Hongyu Yang, and Di Huang. Initno: Boosting text-to-image diffusion models via initial noise optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9380–9389, 2024.

[11] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7514–7528, 2021.

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

[13] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[14] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.

[15] Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. In *Advances in Neural Information Processing Systems*, volume 36, pages 78723–78747, 2023.

[16] Aapo Hyvärinen, Jarmo Hurri, and Patrik O. Hoyer. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision*, volume 39. Springer Science & Business Media, 2009.

[17] Amita Kamath, Kai-Wei Chang, Ranjay Krishna, Luke Zettlemoyer, Yushi Hu, and Marjan Ghazvininejad. GenEval 2: Addressing benchmark drift in text-to-image evaluation. *arXiv preprint arXiv:2512.16853*, 2025.

[18] Korrawe Karunratanakul, Konpat Preechakul, Emre Aksan, Thabo Beeler, Supasorn Suwajanakorn, and Siyu Tang. Optimizing diffusion noise can serve as universal motion priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1334–1345, 2024.

[19] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014.

[20] Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. In *Advances in Neural Information Processing Systems*, volume 36, pages 36652–36663, 2023.

[21] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*, 128(7):1956–1981, 2020.

[22] Shuangqi Li, Hieu Le, Jingyi Xu, and Mathieu Salzmann. Enhancing compositional text-to-image generation with reliable random seeds. In *International Conference on Learning Representations*, 2025.

[23] Zeming Li, Xiangyue Liu, Xiangyu Zhang, Ping Tan, and Heung-Yeung Shum. NoiseAR: Autoregressing initial noise prior for diffusion models. *arXiv preprint arXiv:2506.01337*, 2025.

[24] Artem Lukoianov, Chenyang Yuan, Justin Solomon, and Vincent Sitzmann. Locality in image diffusion models emerges from data statistics. In *Advances in Neural Information Processing Systems*, volume 38, pages 95121–95157, 2025.

[25] Yongzhe Lyu, Yu Wu, Yutian Lin, and Bo Du. Is-diff: Improving diffusion-based inpainting with better initial seed. *IEEE Transactions on Image Processing*, 2026.

[26] Jiafeng Mao, Xueting Wang, and Kiyoharu Aizawa. The lottery ticket hypothesis in denoising: Towards semantic-driven initialization. In *European Conference on Computer Vision*, pages 93–109. Springer, 2024.

[27] Boming Miao, Chunxiao Li, Xiaoxiao Wang, Andi Zhang, Rui Sun, Zizhe Wang, and Yao Zhu. Noise diffusion for enhancing semantic faithfulness in text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23575–23584, 2025.

[28] Zachary Novack, Julian McAuley, Taylor Berg-Kirkpatrick, and Nicholas J Bryan. Ditto: Diffusion inference-time t-optimization for music generation. *arXiv preprint arXiv:2401.12179*, 2024.

[29] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[30] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.

[31] Yuxi Ren, Xin Xia, Yanzuo Lu, Jiacheng Zhang, Jie Wu, Pan Xie, Xing Wang, and Xuefeng Xiao. Hyper-SD: Trajectory segmented consistency model for efficient image synthesis. In *Advances in Neural Information Processing Systems*, volume 37, pages 117340–117362, 2024.

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

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[34] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

[35] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pages 87–103. Springer, 2024.

[36] Vikash Sehwag, Xianghao Kong, Jingtao Li, Michael Spranger, and Lingjuan Lyu. Stretching each dollar: Diffusion training from scratch on a micro-budget. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28596–28608, 2025.

[37] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[38] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, pages 32211–32252. PMLR, 2023.

[39] Zhiwei Tang, Jiangweizhi Peng, Jiasheng Tang, Mingyi Hong, Fan Wang, and Tsung-Hui Chang. Tuning-free alignment of diffusion models with direct noise optimization. In *ICML 2024 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2024.

[40] Masatoshi Uehara, Xingyu Su, Yulai Zhao, Xiner Li, Aviv Regev, Shuiwang Ji, Sergey Levine, and Tommaso Biancalani. Reward-guided iterative refinement in diffusion models at test-time with applications to protein and DNA design. In *International Conference on Machine Learning*, 2025.

[41] Masatoshi Uehara, Yulai Zhao, Chenyu Wang, Xiner Li, Aviv Regev, Sergey Levine, and Tommaso Biancalani. Inference-time alignment in diffusion models with reward-guided generation: Tutorial and review. *arXiv preprint arXiv:2501.09685*, 2025.

[42] 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*, volume 30, 2017.

[43] Bram Wallace, Akash Gokul, Stefano Ermon, and Nikhil Naik. End-to-end diffusion latent optimization improves classifier guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7280–7290, 2023.

[44] Qingsong Wang, Zhengchao Wan, Mikhail Belkin, and Yusu Wang. Seeds of structure: Patch PCA reveals universal compositional cues in diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pages 116960–116993, 2025.

[45] Haoning Wu, Zicheng Zhang, Weixia Zhang, Chaofeng Chen, Liang Liao, Chunyi Li, Yixuan Gao, Annan Wang, Erli Zhang, Wenxiu Sun, Qiong Yan, Xiongkuo Min, Guangtao Zhai, and Weisi Lin. Q-Align: Teaching LMMs for visual scoring via discrete text-defined levels. In *International Conference on Machine Learning*, 2024.

[46] Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023.

[47] Qingsong Xie, Zhenyi Liao, Zhijie Deng, Haonan Lu, et al. Tlcm: Training-efficient latent consistency model for image generation with 2-8 steps. *arXiv preprint arXiv:2406.05768*, 2024.

[48] Sirui Xie, Zhisheng Xiao, Diederik P Kingma, Tingbo Hou, Ying N Wu, Kevin Murphy, Tim Salimans, Ben Poole, and Ruiqi Gao. Em distillation for one-step diffusion models. In *Advances in Neural Information Processing Systems*, volume 37, pages 45073–45104, 2024.

[49] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. ImageReward: Learning and evaluating human preferences for text-to-image generation. In *Advances in Neural Information Processing Systems*, volume 36, pages 15903–15935, 2023.

[50] Katherine Xu, Lingzhi Zhang, and Jianbo Shi. Good seed makes a good crop: Discovering secret seeds in text-to-image diffusion models. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3024–3034. IEEE, 2025.

[51] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6613–6623, 2024.

[52] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, In So Kweon, and Junmo Kim. Text-to-image diffusion models in generative ai: A survey. *arXiv preprint arXiv:2303.07909*, 2023.

[53] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[54] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2017.

[55] Zikai Zhou, Shitong Shao, Lichen Bai, Shufei Zhang, Zhiqiang Xu, Bo Han, and Zeke Xie. Golden noise for diffusion models: A learning framework. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17688–17697, 2025.

A Theoretical Derivations

A.1 Preliminaries

Gaussian Distribution. We denote $\mathcal{N}(\mu, \Sigma)$ as a multivariate Gaussian distribution with mean μ and covariance matrix Σ. In particular, let $\phi_d(x)$ be the probability density function (PDF) of the d-dimensional standard Gaussian distribution $\mathcal{N}(0, I_d)$, defined as:

$$\phi_d(x) = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}\|x\|^2\right). \tag{9}$$

A fundamental property of Gaussian distributions is their closure under affine transformations. Specifically, if a random vector x follows a Gaussian distribution $x \sim \mathcal{N}(\mu, \Sigma)$, then for any matrix $A \in \mathbb{R}^{m\times d}$ and vector $b \in \mathbb{R}^m$, the transformed variable $y = Ax + b$ is also Gaussian:

$$y \sim \mathcal{N}(A\mu + b, A\Sigma A^\top). \tag{10}$$

Furthermore, the standard Gaussian distribution $\mathcal{N}(0, I_d)$ exhibits *rotational invariance*. If A is an orthogonal matrix (i.e., $A^\top A = AA^\top = I_d$), the covariance structure is preserved under the transformation $y = Ax$, yielding:

$$y \sim \mathcal{N}(0, I_d). \tag{11}$$

This property ensures that the statistical characteristics of the Gaussian distribution remain unchanged under orthogonal transformations, such as rotations or changes of basis.

Exponential Tilting. Given a base distribution $q_0(x)$ and a real-valued function $f(x)$, the exponentially tilted distribution is defined as

$$q(x) = \frac{1}{Z} q_0(x) \exp(f(x)), \tag{12}$$

where $Z = \int q_0(x) \exp(f(x))\, dx$ is the normalization constant, assumed to be finite.

This construction reweights the base distribution q_0 by assigning higher probability mass to regions where $f(x)$ is large. As a result, $q(x)$ is biased toward such regions while preserving the overall structure of the base distribution.

In the context of reward-based modeling, $f(x)$ typically corresponds to a reward function $r(x)$, and the tilted distribution takes the form

$$q(x) \propto q_0(x) \exp(r(x)). \tag{13}$$

This formulation commonly arises in probabilistic inference and energy-based models, where the exponential term softly biases the distribution toward high-reward regions while maintaining stochasticity.

Importantly, exponential tilting only reweights the base distribution q_0 and does not assign probability mass to regions where $q_0(x) = 0$. In other words, if a point x is impossible under q_0, it remains impossible under q. As a result, the support of q is contained within that of q_0. This means that exponential tilting does not introduce new regions in the space, but instead redistributes probability mass within the existing domain. Such a property makes it particularly suitable for guiding generation, as it biases the distribution toward desirable regions while preserving the overall structure of the original distribution.

Kullback–Leibler Divergence. For two probability densities p and q, the Kullback–Leibler (KL) divergence is defined as

$$D_{\mathrm{KL}}(p \,\|\, q) = \mathbb{E}_p\left[\log \frac{p(x)}{q(x)}\right], \tag{14}$$

assuming that p is absolutely continuous with respect to q. It measures the discrepancy between two distributions, although it is not a symmetric distance. The KL divergence is always non-negative and equals zero if and only if $p = q$ almost everywhere.

An equivalent expression is

$$D_{\mathrm{KL}}(p \,\|\, q) = \mathbb{E}_p[\log p(x)] - \mathbb{E}_p[\log q(x)], \tag{15}$$

which shows that minimizing $D_{\mathrm{KL}}(p \,\|\, q)$ encourages q to assign high probability to samples drawn from p.

A.2 Gaussian Prior Preservation under PCA Projection

We show that the orthonormal PCA transformation preserves the Gaussian distribution. Let $z_0 \sim \mathcal{N}(0, I_D)$ and partition it into non-overlapping patches $\{s_i\}_{i=1}^N$, where $s_i \in \mathbb{R}^d$. Define $w_i = V^\top s_i$ with an orthonormal matrix $V \in \mathbb{R}^{d\times d}$. Then $w_i \sim \mathcal{N}(0, I_d)$.

Proof. Let $B_i \in \mathbb{R}^{d\times D}$ denote the selection matrix that extracts the i-th patch from z_0, so that

$$s_i = B_i z_0. \tag{16}$$

Since $z_0 \sim \mathcal{N}(0, I_D)$ and s_i is a linear transformation of z_0, s_i is also Gaussian:

$$s_i \sim \mathcal{N}(B_i 0,\ B_i I_D B_i^\top). \tag{17}$$

Because B_i selects d distinct coordinates, we have $B_i I_D B_i^\top = B_i B_i^\top = I_d$. Hence,

$$s_i \sim \mathcal{N}(0, I_d). \tag{18}$$

Applying the property of Gaussian distribution to $s_i \sim \mathcal{N}(0, I_d)$ with $A = V^\top$ and define $w_i = V^\top s_i$, we obtain

$$w_i \sim \mathcal{N}(V^\top 0,\ V^\top I_d V).$$

Using the orthonormality of V, $V^\top I_d V = V^\top V = I_d$.

Therefore,

$$w_i \sim \mathcal{N}(0, I_d). \tag{19}$$

□

A.3 Gaussian Prior Preservation for the Full Coefficient Collection

We show that the full stacked coefficient vector, obtained from non-overlapping patches via an orthonormal PCA transform, follows a standard Gaussian distribution.

Let

$$w := \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} \in \mathbb{R}^{Nd}, \tag{20}$$

where each $w_i \in \mathbb{R}^d$ denotes the PCA coefficient vector of the i-th patch. For notational convenience, w may also be viewed as an $N \times d$ array, but we treat it as a vector in $\mathbb{R}^{Nd}$ when defining its distribution.

We will show that

$$w \sim \mathcal{N}(0, I_{Nd}), \tag{21}$$

which we denote by $q(w) = \mathcal{N}(0, I)$.

Proof. From Section A.2, for each patch i we have

$$w_i = V^\top s_i \sim \mathcal{N}(0, I_d), \tag{22}$$

where V is an orthonormal PCA basis matrix satisfying $V^\top V = V V^\top = I_d$. To establish the joint distribution of the full collection w, it remains to analyze the stacked patch vector.

Define

$$\mathbf{s} := \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{bmatrix} \in \mathbb{R}^{Nd}. \tag{23}$$

Each patch is obtained from the initial noise $z_0 \sim \mathcal{N}(0, I_D)$ via a linear selection:

$$s_i = B_i z_0, \tag{24}$$

where $B_i \in \mathbb{R}^{d\times D}$ is a selection matrix that extracts the coordinates corresponding to the i-th non-overlapping patch.

Stacking all patches, define

$$B := \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{bmatrix} \in \mathbb{R}^{Nd\times D}, \tag{25}$$

so that

$$\mathbf{s} = Bz_0. \tag{26}$$

Since $z_0 \sim \mathcal{N}(0, I_D)$ and p is a linear transformation of z_0, it follows that

$$\mathbf{s} \sim \mathcal{N}(0, BI_DB^\top) = \mathcal{N}(0, BB^\top). \tag{27}$$

Because the patches are non-overlapping and each E_i selects d distinct coordinates, we have

$$B_iB_i^\top = I_d, \quad B_iB_j^\top = 0 \quad (i \neq j). \tag{28}$$

Thus,

$$BB^\top = \begin{bmatrix} I_d & 0 & \cdots & 0 \\ 0 & I_d & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_d \end{bmatrix} = I_{Nd}, \tag{29}$$

which implies

$$\mathbf{s} \sim \mathcal{N}(0, I_{Nd}). \tag{30}$$

Next, define the block-diagonal matrix

$$\bar{V} := \begin{bmatrix} V^\top & 0 & \cdots & 0 \\ 0 & V^\top & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & V^\top \end{bmatrix} \in \mathbb{R}^{Nd\times Nd}. \tag{31}$$

Then the full coefficient vector can be written as

$$w = \bar{V}\mathbf{s}. \tag{32}$$

Since $\mathbf{s} \sim \mathcal{N}(0, I_{Nd})$ and $\bar{V}$ is orthonormal (as each block $V^\top$ is orthonormal), we have

$$w \sim \mathcal{N}(0, \bar{V}I_{Nd}\bar{V}^\top) = \mathcal{N}(0, \bar{V}\bar{V}^\top) = \mathcal{N}(0, I_{Nd}). \tag{33}$$

Therefore, the full stacked coefficient vector follows a standard Gaussian distribution:

$$w \sim \mathcal{N}(0, I_{Nd}). \tag{34}$$

This completes the proof. □

A.4 Factorization of the Gaussian Prior over Low- and High-Frequency Components

We show that the standard Gaussian prior factorizes over the low- and high-frequency components.

Let

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} \in \mathbb{R}^{N\times d}, \tag{35}$$

where each $w_i \in \mathbb{R}^d$, and decompose

$$w = (w_L, w_H), \tag{36}$$

with $w_L \in \mathbb{R}^{N\times k}$ and $w_H \in \mathbb{R}^{N\times(d-k)}$.

Then the Gaussian prior factorizes as

$$q(w) = q_L(w_L)\, q_H(w_H). \tag{37}$$

Proof. From Appendix A.3, the full coefficient collection satisfies

$$w \sim \mathcal{N}(0, I). \tag{38}$$

Under the decomposition $w = (w_L, w_H)$, the covariance matrix can be written in block form as

$$\mathrm{Cov}(w) = \begin{bmatrix} \mathrm{Cov}(w_L) & \mathrm{Cov}(w_L, w_H) \\ \mathrm{Cov}(w_H, w_L) & \mathrm{Cov}(w_H) \end{bmatrix}. \tag{39}$$

Since w is standard Gaussian, this covariance equals the identity, implying

$$\mathrm{Cov}(w_L, w_H) = 0. \tag{40}$$

Because (w_L, w_H) is jointly Gaussian with zero cross-covariance, the two variables are independent. Therefore,

$$q(w_L, w_H) = q_L(w_L)\, q_H(w_H). \tag{41}$$

□

A.5 (Proof of Proposition 3.1) Low-Frequency Approximation of the Tilted Distribution

Proof. By definition of the reward-tilted distribution,

$$q^\star(w) \propto q(w) \exp\big(r(g(w_L, w_H))\big). \tag{42}$$

Using the decomposition

$$r(g(w_L, w_H)) = \bar{r}(w_L) + \delta(w), \qquad \delta(w) := r(g(w_L, w_H)) - \bar{r}(w_L),$$

we obtain

$$q^\star(w) \propto q(w) \exp\big(\bar{r}(w_L) + \delta(w)\big), \tag{43}$$

where $|\delta(w)| \le \epsilon$ by assumption.

The low-frequency-only tilted distribution is defined as

$$\tilde{q}^\star(w) \propto q(w) \exp\big(\bar{r}(w_L)\big). \tag{44}$$

Let Z and $\tilde{Z}$ denote the corresponding normalization constants:

$$q^\star(w) = \frac{1}{Z} q(w) \exp\big(\bar{r}(w_L) + \delta(w)\big), \qquad \tilde{q}^\star(w) = \frac{1}{\tilde{Z}} q(w) \exp\big(\bar{r}(w_L)\big).$$

Therefore,

$$\log \frac{q^\star(w)}{\tilde{q}^\star(w)} = \delta(w) + \log \frac{\tilde{Z}}{Z}. \tag{45}$$

Next, using

$$\tilde{q}^\star(w) = \frac{1}{\tilde{Z}} q(w) \exp(\bar{r}(w_L)),$$

we can write

$$Z = \int q(w) \exp(\bar{r}(w_L)) \exp(\delta(w))\, dw \tag{46}$$

$$= \tilde{Z} \int \tilde{q}^\star(w) \exp(\delta(w))\, dw \tag{47}$$

$$= \tilde{Z}\, \mathbb{E}_{\tilde{q}}[\exp(\delta(w))]. \tag{48}$$

Since $|\delta(w)| \le \epsilon$, we have

$$e^{-\epsilon} \le e^{\delta(w)} \le e^{\epsilon},$$

and hence

$$e^{-\epsilon} \le \frac{Z}{\tilde{Z}} \le e^{\epsilon}.$$

Equivalently,

$$\left| \log \frac{\tilde{Z}}{Z} \right| \le \epsilon. \tag{49}$$

Moreover, since $|\delta(w)| \leq \epsilon$, it follows that

$$-\epsilon \leq \mathbb{E}_{q^\star}[\delta(w)] \leq \epsilon. \tag{50}$$

Therefore,

$$\begin{aligned} D_{\mathrm{KL}}(q^\star \| \tilde{q}^\star) &= \mathbb{E}_{q^\star}\left[\log \frac{q^\star(w)}{\tilde{q}^\star(w)}\right] && (51)\\ &= \mathbb{E}_{q^\star}[\delta(w)] + \log \frac{\tilde{Z}}{Z} && (52)\\ &\leq \epsilon + \epsilon && (53)\\ &= 2\epsilon. && (54) \end{aligned}$$

□

A.6 (Proof of Proposition 3.2) Derivation of Tractable Loss Function

Derivation of KL Divergence Objective. We aim to solve the following optimization problem:

$$D_{\mathrm{KL}}(q_L^\phi \,\|\, \tilde{q}_L^\star) = \mathbb{E}_{w_L \sim q_L^\phi}\left[\log \frac{q_L^\phi(w_L)}{\tilde{q}_L^\star(w_L)}\right]. \tag{55}$$

Introducing the normalization constant Z, we can write the low-frequency-only tilted distribution as

$$\tilde{q}_L^\star(w_L) = \frac{1}{Z} q_L(w_L) \exp(\bar{r}(w_L)). \tag{56}$$

Taking the logarithm, we obtain

$$\log \tilde{q}_L^\star(w_L) = \log q_L(w_L) + \bar{r}(w_L) - \log Z. \tag{57}$$

Substituting this into the KL divergence, we obtain

$$D_{\mathrm{KL}}(q_L^\phi \,\|\, \tilde{q}_L^\star) = \mathbb{E}_{w_L \sim q_L^\phi}\left[\log q_L^\phi(w_L) - \log q_L(w_L) - \bar{r}(w_L) + \log Z\right]. \tag{58}$$

Separating the terms, we have

$$D_{\mathrm{KL}}(q_L^\phi \,\|\, \tilde{q}_L^\star) = \mathbb{E}_{w_L \sim q_L^\phi}\left[\log q_L^\phi(w_L) - \log q_L(w_L)\right] - \mathbb{E}_{w_L \sim q_L^\phi}[\bar{r}(w_L)] + \log Z. \tag{59}$$

Thus, the objective can be written as

$$D_{\mathrm{KL}}(q_L^\phi \,\|\, \tilde{q}_L^\star) = D_{\mathrm{KL}}(q_L^\phi \,\|\, q_L) - \mathbb{E}_{w_L \sim q_L^\phi}[\bar{r}(w_L)] + C, \tag{60}$$

where $C = \log Z$ is constant with respect to ϕ, and can therefore be ignored during optimization.

Practical Approximation of the Low-Frequency Reward. In practice, the low-frequency reward $\bar{r}(w_L)$ is intractable to compute, as it requires taking an expectation over the high-frequency components w_H. However, under Assumption (1), $\bar{r}(w_L)$ can be well approximated by $r(g(w))$ for any w, as their discrepancy is uniformly bounded by ϵ. Consequently, we replace $\bar{r}(w_L)$ with $r(g(w))$ in practice, leading to a tractable training objective.

Based on this approximation, we derive the practical training objective from the low-frequency KL formulation:

$$\mathcal{L}(\phi) = D_{\mathrm{KL}}(q_L^\phi \,\|\, q_L) - \mathbb{E}_{w \sim q^\phi}[r(g(w))]. \tag{61}$$

The true objective in the low-frequency space is given by Eq. (60). The proposed objective $\mathcal{L}(\phi)$ serves as a surrogate for this quantity, and their discrepancy is uniformly bounded by ϵ. This leads to the following bound:

$$\mathcal{L}(\phi) - \epsilon \lesssim D_{\mathrm{KL}}(q_L^\phi \| q_L) - \mathbb{E}_{w_L \sim q_L^\phi}[\bar{r}(w_L)] \lesssim \mathcal{L}(\phi) + \epsilon. \tag{62}$$

Exact KL divergence. The proof is based on the approach of [8]. We define the transformation induced by the network in the low-frequency space as

$$T_\phi(w_L) = w_L + h_\phi(w_L), \tag{63}$$

where h_ϕ is a residual predictor. Let $q_L = \mathcal{N}(0, I)$ denote the standard Gaussian prior over w_L, and let $q_L^\phi = (T_\phi)_\# q_L$ denote the pushforward distribution induced by T_ϕ.

To apply the change-of-variables formula, the transformation T_ϕ must be invertible with a continuously differentiable inverse. The following lemma provides a sufficient condition for this property.

Lemma A.1 (Invertibility of the residual map)**.** *Let $h_\phi : \mathbb{R}^{Nk} \to \mathbb{R}^{Nk}$ be a continuously differentiable function. If there exists a constant M such that*

$$\|J_{h_\phi}(w_L)\|_2 \le M < 1 \quad \textit{for all } w_L \in \mathbb{R}^{Nk}, \tag{64}$$

then T_ϕ is a bijection from $\mathbb{R}^{Nk}$ onto $\mathbb{R}^{Nk}$, and its inverse is continuously differentiable.

Proof. Since h_ϕ is continuously differentiable and satisfies Eq. (64), it follows from the mean value theorem that h_ϕ is M-Lipschitz:

$$\|h_\phi(x) - h_\phi(y)\| \le M\|x - y\| \qquad \forall x, y \in \mathbb{R}^{Nk}. \tag{65}$$

(i) Injectivity. For any $x, y \in \mathbb{R}^{Nk}$,

$$\begin{aligned} \|T_\phi(x) - T_\phi(y)\| &= \|(x - y) + (h_\phi(x) - h_\phi(y))\| \\ &\ge \|x - y\| - \|h_\phi(x) - h_\phi(y)\| \\ &\ge (1 - M)\|x - y\|. \end{aligned} \tag{66}$$

Since $1 - M > 0$, it follows that $T_\phi(x) = T_\phi(y)$ implies $x = y$. Hence, T_ϕ is injective.

(ii) Surjectivity. Fix any $z \in \mathbb{R}^{Nk}$ and define

$$\Psi_z(x) := z - h_\phi(x). \tag{67}$$

Then for any x, y,

$$\|\Psi_z(x) - \Psi_z(y)\| = \|h_\phi(x) - h_\phi(y)\| \le M\|x - y\|. \tag{68}$$

Thus, Ψ_z is a contraction with constant $L < 1$. By the Banach fixed-point theorem, there exists a unique $x^\star$ such that

$$x^\star = \Psi_z(x^\star) = z - h_\phi(x^\star). \tag{69}$$

Rearranging gives

$$z = x^\star + h_\phi(x^\star) = T_\phi(x^\star). \tag{70}$$

Since z is arbitrary, T_ϕ is surjective.

(iii) Differentiability of the inverse. The Jacobian of T_ϕ is

$$J_{T_\phi}(w_L) = I + J_{h_\phi}(w_L). \tag{71}$$

we have Eq. (64), hence, the spectral radius satisfies

$$\rho(J_{h_\phi}(w_L)) \le \|J_{h_\phi}(w_L)\|_2 < 1. \tag{72}$$

Therefore, for every eigenvalue λ of $J_{h_\phi}(w_L)$, we have $|\lambda| < 1$, so

$$1 + \lambda \ne 0.$$

It follows that $I + J_{h_\phi}(w_L)$ is invertible for all w_L, i.e.,

$$\det(J_{T_\phi}(w_L)) \ne 0 \qquad \forall w_L.$$

By the inverse function theorem, T_ϕ admits a locally defined inverse around every point, and this inverse is continuously differentiable. Since Parts (i) and (ii) establish that T_ϕ is globally bijective, these local inverses are consistent and define a global inverse T_ϕ^{-1}, which is also continuously differentiable. □

Under the condition of Lemma A.1 (i.e., $\|J_{h_\phi}(w_L)\|_2 \le M < 1$), we can use change of variables formula. The density of q_L^ϕ is given by

$$q_L^\phi(\hat{w}_L) = q_L(w_L) \left| \det\left(\frac{\partial T_\phi(w_L)}{\partial w_L} \right) \right|^{-1}, \tag{73}$$

where $\hat{w}_L = T_\phi(w_L)$. Since

$$\frac{\partial T_\phi(w_L)}{\partial w_L} = I + J_{h_\phi}(w_L),$$

this yields

$$q_L^\phi(\hat{w}_L) = q_L(w_L) \left| \det\big(I + J_{h_\phi}(w_L)\big) \right|^{-1}. \tag{74}$$

Using the definition of KL divergence,

$$D_{\mathrm{KL}}(q_L^\phi \,\|\, q_L) = \mathbb{E}_{\hat{w}_L \sim q_L^\phi} \left[\log \frac{q_L^\phi(\hat{w}_L)}{q_L(\hat{w}_L)} \right], \tag{75}$$

we substitute the expression of $q_L^\phi(\hat{w}_L)$ to obtain

$$\log \frac{q_L^\phi(\hat{w}_L)}{q_L(\hat{w}_L)} = \log q_L(w_L) - \log q_L(\hat{w}_L) - \log \left| \det\big(I + J_{h_\phi}(w_L)\big) \right|. \tag{76}$$

Since q_L is a standard Gaussian distribution, its log-density is quadratic. Thus,

$$\log q_L(w_L) - \log q_L(\hat{w}_L) = \frac{1}{2}\|\hat{w}_L\|^2 - \frac{1}{2}\|w_L\|^2. \tag{77}$$

Substituting $\hat{w}_L = w_L + h_\phi(w_L)$ and expanding,

$$\begin{aligned} \frac{1}{2}\|\hat{w}_L\|^2 - \frac{1}{2}\|w_L\|^2 &= \frac{1}{2}\|w_L + h_\phi(w_L)\|^2 - \frac{1}{2}\|w_L\|^2 \\ &= \frac{1}{2}\|h_\phi(w_L)\|^2 + \langle w_L, h_\phi(w_L)\rangle. \end{aligned} \tag{78}$$

The first term corresponds to the magnitude of the residual update, while the second term captures the interaction between the input w_L and the update direction.

To simplify the second term, we apply Stein's lemma under the Gaussian base measure $w_L \sim \mathcal{N}(0, I)$. Assume that h_ϕ is continuously differentiable and satisfies

$$\mathbb{E}\big[\|h_\phi(w_L)\|\big] < \infty, \qquad \mathbb{E}\big[\|J_{h_\phi}(w_L)\|_2\big] < \infty, \tag{79}$$

so that integration by parts applies and the boundary terms vanish under the Gaussian measure. In our setting, these conditions are ensured by the Jacobian bound in Eq. (64), which implies that h_ϕ is Lipschitz and hence has at most linear growth.

Applying Stein's lemma yields

$$\mathbb{E}\big[\langle w_L, h_\phi(w_L)\rangle\big] = \mathbb{E}\big[\mathrm{Tr}(J_{h_\phi}(w_L))\big]. \tag{80}$$

Combining the above results, we obtain the exact expression

$$D_{\mathrm{KL}}(q_L^\phi \,\|\, q_L) = \mathbb{E}_{w_L \sim q_L} \left[\frac{1}{2}\|h_\phi(w_L)\|^2 + \mathrm{Tr}\big(J_{h_\phi}(w_L)\big) - \log\det\big(I + J_{h_\phi}(w_L)\big) \right]. \tag{81}$$

This expression shows that the KL divergence decomposes into a quadratic penalty on h_ϕ and a Jacobian-based correction capturing local geometric distortion.

Approximation of the Jacobian term. The exact KL expression derived above contains a Jacobian-dependent correction term,

$$\mathrm{Tr}(J_{h_\phi}(w_L)) - \log\det\big(I + J_{h_\phi}(w_L)\big). \tag{82}$$

While this term is necessary for exact density transformation, it is computationally expensive, as it requires evaluating both the Jacobian trace and the log-determinant.

To obtain a tractable objective, we analyze this term under the small-Jacobian assumption in Eq. (64) as in Lemma A.1. Let

$$A := J_{h_\phi}(w_L). \tag{83}$$

Since $\|A\|_2 \le M < 1$, its spectral radius satisfies $\rho(A) < 1$, and hence the matrix logarithm admits the convergent expansion

$$\log(I + A) = \sum_{j=1}^{\infty} \frac{(-1)^{j+1}}{j} A^j. \tag{84}$$

Using the identity $\log\det(I + A) = \mathrm{Tr}(\log(I + A))$, we obtain

$$\log\det(I + A) = \sum_{j=1}^{\infty} \frac{(-1)^{j+1}}{j} \mathrm{Tr}(A^j). \tag{85}$$

Therefore,

$$\mathrm{Tr}(A) - \log\det(I + A) = \sum_{j=2}^{\infty} \frac{(-1)^{j}}{j} \mathrm{Tr}(A^j). \tag{86}$$

Taking absolute values and using $|\mathrm{Tr}(A^j)| \le Nk\,\|A\|_2^j \le Nk\,M^j$, we get

$$|\mathrm{Tr}(A) - \log\det(I + A)| \le Nk \sum_{j=2}^{\infty} \frac{M^j}{j} = Nk\big(-\log(1 - M) - M\big). \tag{87}$$

Substituting this bound into the KL expression yields

$$\left| D_{\mathrm{KL}}(q_L^\phi \,\|\, q_L) - \frac{1}{2}\mathbb{E}_{w_L \sim q_L} \|h_\phi(w_L)\|^2 \right| \le Nk\big(-\log(1 - M) - M\big). \tag{88}$$

When $M \ll 1$, the bound admits the second-order approximation

$$-\log(1 - M) - M = \frac{M^2}{2} + O(M^3), \tag{89}$$

which becomes negligible as $M \to 0$. Consequently,

$$D_{\mathrm{KL}}(q_L^\phi \,\|\, q_L) \approx \frac{1}{2}\mathbb{E}_{w_L \sim q_L} \|h_\phi(w_L)\|^2. \tag{90}$$

This shows that, in the small-Jacobian regime, the KL divergence is well approximated by a simple quadratic penalty on the residual update h_ϕ.

Final practical objective. Applying this approximation yields the practical training loss

$$\mathcal{L}(\phi) = \mathbb{E}_{w \sim q}\left[\frac{1}{2}\|h_\phi(w_L)\|^2 - r\big(g(w_L + h_\phi(w_L), w_H)\big)\right]. \tag{91}$$

This objective admits a clear interpretation: the first term acts as a regularizer that penalizes large deviations from the base Gaussian prior in the low-frequency subspace, while the second term encourages updates that improve the reward.

B Additional Details on Model Architecture and Loss Function

B.1 Model Architecture of the LENS Framework

Patch PCA Codec. The Patch PCA Codec transforms latent noise into a patch-wise PCA coefficient space and reconstructs it back to the latent space.

Let $z \in \mathbb{R}^{C\times H\times W}$ denote a latent noise representation, where C, H, and W denote the number of channels, height, and width, respectively. We extract non-overlapping patches using an `unfold` operation with patch size s, producing a sequence of patch vectors:

$$\mathbf{s} = \{s_i\}_{i=1}^{N}, \quad s_i \in \mathbb{R}^d, \tag{92}$$

where $N = \frac{H}{s} \cdot \frac{W}{s}$ and $d = Cs^2$.

Each patch is centered using a precomputed mean $\mu \in \mathbb{R}^d$:

$$\tilde{s}_i = s_i - \mu. \tag{93}$$

Since μ is fixed, centering only shifts the mean of the PCA coefficients while preserving their covariance. For the theoretical analysis in Section 3.1, we absorb this deterministic mean shift into the coefficient coordinates and use re-centered variables with a standard Gaussian prior.

We project the centered patches onto a truncated PCA basis $V' \in \mathbb{R}^{d\times k}$, whose columns are orthonormal (i.e., ${V'}^\top V' = I$):

$$w_{i,L} = \tilde{s}_i V', \tag{94}$$

where $w_{i,L} \in \mathbb{R}^k$ are the retained low-frequency PCA coefficients and $k \ll d$.

The high-frequency component is defined as:

$$y_i = \tilde{s}_i - w_{i,L}{V'}^\top, \tag{95}$$

where y_i lies in the orthogonal complement of the subspace spanned by V'.

Collecting all patches, we obtain:

$$w_L = \{w_{i,L}\}_{i=1}^{N} \in \mathbb{R}^{N\times k}, \quad y = \{y_i\}_{i=1}^{N} \in \mathbb{R}^{N\times d}. \tag{96}$$

Given modified coefficients $\hat{w}_{i,L}$, reconstruction is performed as:

$$\hat{s}_i = \mu + \hat{w}_{i,L}{V'}^\top + y_i. \tag{97}$$

The reconstructed patches are merged back into the latent space using a `fold` operation.

This formulation is mathematically equivalent to performing a full PCA projection using an orthonormal basis $V \in \mathbb{R}^{d\times d}$, obtaining full coefficients $w_i = \tilde{s}_i V$, modifying only the first k coefficients, and reconstructing via $w_i V^\top$.

Specifically, let $V = [V', V'']$, where V' contains the first k principal components and V'' contains the remaining components. Then, since V is orthonormal, we can decompose:

$$\tilde{s}_i = w_{i,L}{V'}^\top + w_{i,H}{V''}^\top, \tag{98}$$

where $w_{i,H}$ denotes the remaining high-frequency coefficients.

In our formulation, the orthogonal component y_i corresponds exactly to the contribution of the high-frequency components, i.e., the component of $\tilde{s}_i$ that lies in the subspace orthogonal to the span of the top-k principal components V':

$$y_i = w_{i,H}{V''}^\top. \tag{99}$$

Therefore, modifying only $w_{i,L}$ while keeping y_i fixed is equivalent to modulating only the first k PCA coefficients in the full PCA space.

LENS Network. The LENS Network is a transformer-based model [42] that predicts residual updates to the PCA coefficients.

The input to the network is a sequence of low-frequency PCA coefficients:

$$w_L \in \mathbb{R}^{N \times k}, \tag{100}$$

where N is the number of patches and k is the number of low-frequency coefficients used for modulation. Each token corresponds to a patch and contains its k-dimensional PCA coefficient vector. Each coefficient vector is first projected into a hidden dimension h:

$$\mathbf{h}^{(0)} = \mathrm{Linear}(w_L) + \mathbf{P}, \quad \mathbf{h}^{(0)} \in \mathbb{R}^{N \times h}, \tag{101}$$

where $\mathbf{P}$ is a sinusoidal positional embedding.

Given a text prompt embedding c, two conditioning paths are used: (1) a pooled text condition that is projected and added to all patch tokens (bias injection), and (2) token-wise text embeddings used in cross-attention.

Each transformer block applies self-attention, text cross-attention, and a feed-forward network (FFN), with residual connections and layer normalization.

After passing through L transformer blocks, the final hidden representation is projected back to the coefficient space:

$$\Delta w_L = \mathrm{Linear}(\mathrm{LN}(\mathbf{h}^{(L)})), \tag{102}$$

where $\Delta w_L \in \mathbb{R}^{N \times k}$ represents the predicted coefficient updates.

B.2 Composition of the Reward-based Loss Function

We employ a composite reward function to guide the optimization of the coefficient modulation network. The overall reward is defined as a weighted combination of multiple complementary scoring functions that capture semantic alignment, human preference, and perceptual quality.

Given a generated image x conditioned on a text prompt c, the total reward is defined as

$$r(x, c) = \sum_i \lambda_i \, r_i(x, c), \tag{103}$$

where $\{r_i\}$ denotes individual reward components and $\{\lambda_i\}$ are weighting coefficients. The proposed reward combines complementary signals: CLIP [11] provides a strong signal for semantic alignment, HPSv2.1 (Human Preference Score) [46] captures absolute perceptual quality, ImageReward [49] promotes visual fidelity by penalizing generative artifacts, and PickScore [20] models relative human preference. This combination yields a more stable and well-balanced optimization signal compared to relying on a single reward model.

In practice, we combine four reward functions:

$$r(x, c) = \lambda_{\text{CLIP}} r_{\text{CLIP}}(x, c) + \lambda_{\text{HPS}} r_{\text{HPS}}(x, c) + \lambda_{\text{IR}} r_{\text{IR}}(x, c) + \lambda_{\text{Pick}} r_{\text{Pick}}(x, c). \tag{104}$$

We set the weights as $\lambda_{\text{CLIP}} = 0.01$, $\lambda_{\text{HPS}} = 5.0$, $\lambda_{\text{IR}} = 1.0$, and $\lambda_{\text{Pick}} = 0.05$.

CLIP Score. The CLIP reward [11] measures semantic alignment by computing the cosine similarity between image and text embeddings in a shared latent space. It encourages the generated image to remain consistent with the input prompt.

HPSv2.1. HPSv2.1 [46] is a model fine-tuned on human preference data to predict aesthetic quality. It captures perceptual attributes such as composition, lighting, and detail.

ImageReward. ImageReward [49] is trained on large-scale human feedback to evaluate overall image quality conditioned on text prompts. It provides a scalar reward reflecting both visual fidelity and semantic alignment.

PickScore. PickScore [20] is trained on pairwise human preference data using a Bradley–Terry objective. It predicts relative preference between images for the same prompt, capturing fine-grained ranking signals.

C Experimental Details

C.1 Implementation Details

PCA Basis Extraction. Before training each model, we precompute the PCA basis for latent patches. To this end, we use ImageNetV2 (MIT License) [30], a dataset designed to evaluate generalization performance. ImageNetV2 preserves the same 1,000-class structure as ImageNet [33] while consisting of 10 newly collected images per class, resulting in a total of 10,000 images without overlap with the original training data. By covering a wide range of classes and visual variations, it provides a reasonable approximation of natural image statistics while mitigating overfitting to a specific dataset distribution.

The PCA basis is computed as follows. First, images are encoded into latent representations using the VAE of the corresponding model. The latent features are then divided into patches according to a predefined patch size and vectorized. Next, the mean vector of the patches is computed, and each patch is centered by subtracting the mean. Based on these centered vectors, a covariance matrix is constructed, from which eigenvectors are obtained. These eigenvectors are used as the PCA basis during both training and inference for each model.

Models and Baselines. In this work, we employ three distilled diffusion models: SD-Turbo[2] [35], Hyper-SDXL[3] [31], and SANA-Sprint[4] [5]. For Hyper-SDXL, we use the 1-step SDXL UNet checkpoint provided in the official repository.

As baselines, we employ ReNo (MIT License)[5] [7] and HyperNoise (MIT License)[6] [8].

ReNo is a test-time optimization method that refines latent representations during inference to better align generated samples with the target distribution. Specifically, it performs iterative optimization at test time without modifying the model parameters. In our experiments, we set the number of optimization iterations to 20 and follow the official implementation for all other hyperparameters. For SANA-Sprint, we reimplement ReNo based on the original codebase to ensure compatibility with this model.

HyperNoise is a hypernetwork-based approach that learns to generate input-conditioned noise modulations to steer the denoising process toward better quality samples. We follow the official implementation for all hyperparameters, including setting the LoRA rank to 128. Since the original implementation does not support Hyper-SDXL, we reimplement HyperNoise for this model based on the provided codebase.

Settings. In LENS, the base generator is pretrained and kept frozen. In the trainable LENS network, most layers use the default PyTorch initialization, while the final output projection is zero-initialized, and the cross-attention gate logits are initialized to -2.0 (gain ≈ 0.119), resulting in near-zero initial perturbations under the default modulation setting. This design is intended to stabilize training by starting from minimal perturbations, with updates initially focusing on the output layer and gradually propagating to the entire network, thereby reducing variance and improving convergence in reward-based learning.

Across all experiments, the guidance scale is set to 0 during both training and evaluation. For all three models—SD-Turbo, Hyper-SDXL, and SANA-Sprint—the LENS network is trained at a resolution of 512×512. This choice is primarily motivated by computational efficiency and resource constraints, while maintaining adequate text–image alignment. During evaluation, images are generated at resolutions of 512, 1024, and 1024, respectively. The number of generation steps is fixed at 1 for both training and evaluation, except for the ablation experiments in the Appendix where the number of steps is varied.

When evaluating on GenEval2 [17], we adopt Soft-TIFA AM as the scoring metric and generate images using different seeds for each prompt to mitigate dependence on random seed selection. To

[2] https://huggingface.co/stabilityai/sd-turbo

[3] https://huggingface.co/ByteDance/Hyper-SD

[4] https://huggingface.co/Efficient-Large-Model/Sana_Sprint_0.6B_1024px_diffusers

[5] https://github.com/ExplainableML/ReNO

[6] https://github.com/ExplainableML/HyperNoise

further reduce statistical variance, we repeat the experiments with three different base seeds and report the averaged results, where each base seed defines a distinct set of per-prompt seeds.

All training and experiments were conducted using Python 3.11.0 on NVIDIA RTX 3090 GPUs, with a 64-core Intel Xeon Gold 6226R CPU. The training data type was set to bfloat16 (bf16) by default. To increase the effective batch size without significantly increasing VRAM usage, gradient accumulation was employed. Additionally, gradient checkpointing was applied to both the text encoder and parts of the reward model to improve memory efficiency. When using two RTX 3090 GPUs, training each LENS network typically required approximately 4 to 6 days, although the exact duration varied slightly depending on the model architecture and overall GPU utilization.

C.2 Hyperparameters

The hyperparameters used to train our LENS Network for each model are summarized in Table 5.

Table 5: Hyperparameters for training the LENS network

	Low-frequency Eigen Noise Shaping (LENS)		
Model	SD-Turbo	Hyper-SDXL	SANA-Sprint
Patch size	4	4	4
Number of low-frequency coefficients (k)	32	32	256
Number of layers (blocks)	4	4	8
Number of attention heads	4	4	8
Gradient norm clipping	1.0	1.0	1.0
Learning rate	$2e{-}5$	$5e{-}6$	$1e{-}3$
Optimizer	AdamW	AdamW	AdamW
Batch size	2	2	4
Accumulation steps	15	8	6
Training epochs	≈ 15	≈ 20	≈ 20
Number of training prompts	$\approx 70k$	$\approx 70k$	$\approx 70k$

D Additional Experimental Results

D.1 Spatial-Frequency Characteristics of PCA Components

To verify the use of the term "low-frequency" for the leading PCA components, we analyze the spatial-frequency characteristics of the PCA bases used by LENS. For each PCA eigenvector, we reshape it into its latent-patch representation, compute its 2D Fourier transform, and measure the spectral centroid of its power spectrum. We then compare the average spectral centroid of the leading k components used by LENS with that of the remaining $d - k$ components, using $k = 32$ for SD-Turbo and Hyper-SDXL and $k = 256$ for SANA-Sprint, following the PCA component settings used in our experiments. We additionally report the Spearman correlation between PCA rank and spectral centroid. Table 6 summarizes the results.

Table 6: Spatial-frequency characteristics of the PCA components used by LENS

Backbone	**Leading k spectral centroid mean**	**Remaining $d - k$ spectral centroid mean**	**Spearman correlation**
SD-Turbo	0.3456	0.4491	0.4293
Hyper-SDXL	0.3670	0.4277	0.2960
SANA-Sprint	0.3260	0.4689	0.8751

Across all three backbones, the leading k components exhibit lower average spectral centroids than their complementary components. Moreover, the positive Spearman correlations indicate an overall tendency for spatial frequency to increase with PCA rank. These results support our use of "low-frequency" as a shorthand for the empirically lower-frequency characteristics of the leading PCA subspace, rather than as an exact identification with the Fourier basis.

D.2 Empirical Validation of Assumption 1

To examine the empirical plausibility of Assumption 1, we evaluate the sensitivity of the reward to perturbations in the low-frequency PCA coefficients w_L and the complementary coefficients w_H. Using SD-Turbo, we construct 250 prompt–noise pairs from 25 prompts and 10 anchor noise samples. For each pair, we perform 30 independent resamplings under two conditions: (i) varying only w_L while fixing w_H, and (ii) varying only w_H while fixing w_L. The reward is computed using the same weighted combination of CLIP, HPS, ImageReward, and PickScore used during training. We quantify reward sensitivity using the mean absolute deviation and the standard deviation across repeated resamplings. We additionally report the H/L ratio, defined as the reward variation induced by perturbing w_H divided by that induced by perturbing w_L. Thus, an H/L ratio below 1 indicates that the complementary coefficients have a smaller effect on the reward than the low-frequency coefficients. The results are summarized in Table 7.

Table 7: Reward sensitivity to perturbations of coefficients

	w_H **varied**	w_L **varied**	H/L **ratio** $\downarrow$
Mean absolute deviation	0.2152	0.4925	0.4371
Mean standard deviation	0.2729	0.6211	0.4393

The two sensitivity measures yield highly consistent results. In both cases, the H/L ratio is approximately 0.44, substantially below 1, indicating that perturbing the complementary coefficients produces less than half the reward variation caused by perturbing the low-frequency coefficients. These results provide empirical evidence supporting the plausibility of Assumption 1 and are consistent with the motivation for restricting modulation to the low-frequency PCA subspace.

D.3 Effect of the Number of Low-frequency Coefficients

We conduct an ablation study on the number of low-frequency coefficients k using the SD-Turbo model. The patch size is fixed at $s = 4$, and only k is varied during training. For SD-Turbo, the latent channel dimension is 4, resulting in a full dimension of $d = 64$. Images are generated using the LENS framework and evaluated on the GenEval2 benchmark. Table 8 summarizes the results for each metric, and Figure 5 compares the GenEval2 mean scores.

As shown, the best performance is achieved at $k = 32$. Increasing k toward the full dimension d degrades performance, as it involves modifying an unnecessarily large number of components. This suggests that effective noise modulation benefits from focusing on a restricted subspace, where limiting the degrees of freedom leads to more stable and efficient optimization.

Table 8: Effect of the number of PCA coefficients k on GenEval2 results

k	**Object** $\uparrow$	**Attribute** $\uparrow$	**Count** $\uparrow$	**Position** $\uparrow$	**Verb** $\uparrow$
8	73.26	42.06	24.59	26.36	3.24
16	73.50	40.83	23.91	28.17	1.53
32	**75.07**	**43.08**	**26.08**	**28.46**	**5.78**
64	74.02	42.55	25.05	28.21	5.33

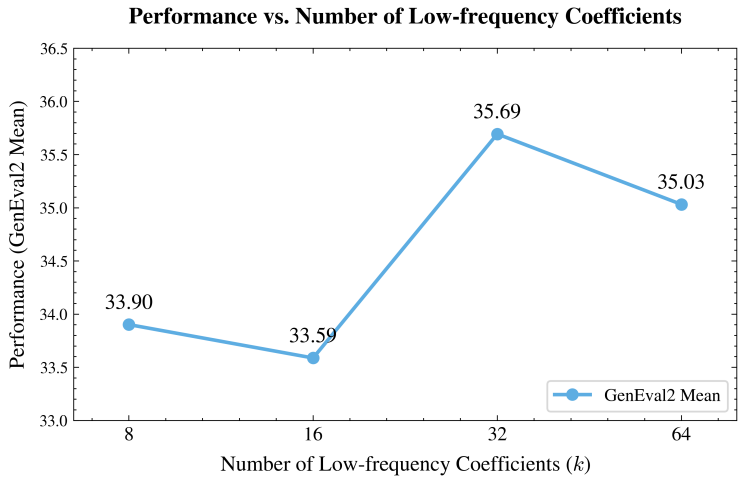


Figure 5: Effect of the number of low-frequency coefficients k on GenEval2 mean

D.4 Effect of the Patch Size

We conduct an ablation study on the patch size s using the SD-Turbo model. The number of low-frequency coefficients k is fixed, and only s is varied. Images are generated using the LENS framework and evaluated on the GenEval2 benchmark. Table 9 summarizes the results for each metric, and Figure 6 compares the GenEval2 mean scores.

The results show that the best performance is achieved at $s = 4$. This suggests that a moderate patch size strikes a good balance between preserving local details and capturing global structure, leading to improved performance.

Table 9: Effect of the patch size s on GenEval2 results

s	**Object** ↑	**Attribute** ↑	**Count** ↑	**Position** ↑	**Verb** ↑
2	73.71	42.37	23.16	28.24	4.35
4	**75.07**	**43.08**	**26.08**	**28.46**	**5.78**
8	73.59	41.06	23.44	28.27	3.67
16	72.65	40.41	22.30	28.05	3.68

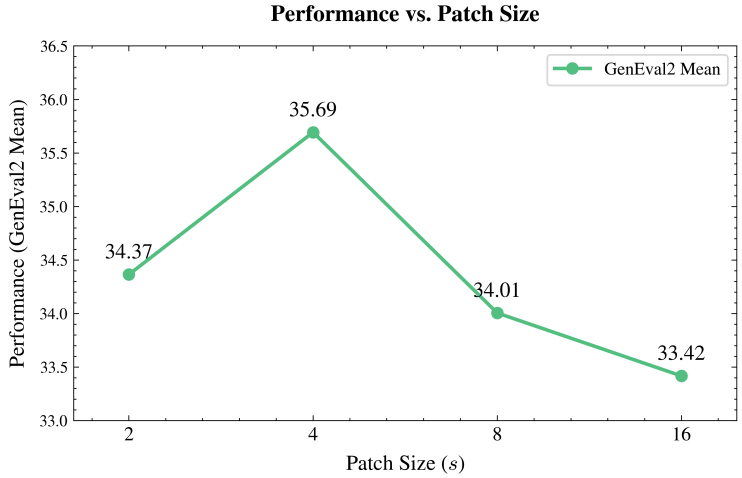


Figure 6: Effect of the patch size s on GenEval2 mean

D.5 Effect of the Number of Transformer Layers

Table 10 presents the performance of LENS with varying numbers of transformer layers for the SD-Turbo model. As the number of layers increases, the GenEval2 mean score decreases from 35.70 to 33.10. This indicates that deeper architectures do not necessarily yield better performance. Since LENS operates on low-dimensional representations, increasing depth may introduce redundant transformations or hinder effective feature utilization across layers. Overall, these results highlight the importance of selecting an appropriate model depth, rather than naively increasing the number of layers, in the design of the LENS architecture.

Table 10: Effect of the number of layers

SD-Turbo + LENS	GenEval2 Mean ↑
4 Layers	**35.70**
8 Layers	34.80
12 Layers	33.10

D.6 Multi-step Analysis

While the main paper focuses on one-step generation, we further analyze performance across varying numbers of inference steps. Table 11 shows the performance trends as the number of generation steps increases. LENS consistently outperforms both random noise initialization and HyperNoise in terms of GenEval2 mean across all settings, except at 4 and 16 steps, where it remains comparable to HyperNoise while still surpassing random noise.

Notably, LENS achieves the largest performance gain in the one-step setting, attaining the highest score of 35.69. As the number of inference steps increases, performance slightly degrades. We attribute this trend to the design of the underlying model, SD-Turbo, which is specifically optimized through distillation for one-step generation. Consequently, while LENS remains competitive in multi-step settings, it achieves the best performance and efficiency when used with fewer inference steps.

Table 11: Step-wise performance

SD-Turbo	GenEval2 Mean ↑
One-step	33.95
+ HyperNoise	34.96
+ LENS (Ours)	**35.69**
Two-step	30.92
+ HyperNoise	32.95
+ LENS (Ours)	**33.29**
Four-step	29.82
+ HyperNoise	**31.72**
+ LENS (Ours)	31.00
Eight-step	29.16
+ HyperNoise	30.58
+ LENS (Ours)	**30.61**
Sixteen-step	29.10
+ HyperNoise	**30.57**
+ LENS (Ours)	30.10

D.7 Results on Other Benchmarks

T2I-CompBench. Table 12 presents quantitative evaluation results on T2I-CompBench [15]. LENS achieves competitive performance, closely matching HyperNoise with only a marginal gap, while significantly outperforming the Random Noise baseline. Notably, despite its substantially faster generation speed, LENS maintains comparable performance to HyperNoise, resulting in a more favorable performance–efficiency trade-off.

These results demonstrate that LENS provides an effective and resource-efficient solution for ensuring semantic alignment in text-to-image generation.

Table 12: Results on T2I-CompBench

Model	Color ↑	Shape ↑	Texture ↑	Spatial ↑	Numeracy ↑	Mean ↑
SD-Turbo	0.56	0.43	0.53	0.14	0.50	0.43
+ HyperNoise	**0.58**	**0.46**	**0.57**	**0.19**	**0.53**	**0.47**
+ LENS (Ours)	0.57	**0.46**	0.55	0.18	0.52	0.45

DPG-Bench. Table 13 presents the DPG-Bench [14] scores for the SD-Turbo model. LENS achieves highly competitive performance, closely matching HyperNoise with only a marginal gap. Notably, despite its substantially faster generation speed, LENS maintains comparable semantic alignment to the strongest baseline, highlighting its superior practical efficiency.

Table 13: Results on DPG-Bench

Method	DPG-Bench Score ↑
SD-Turbo	70.94
+ HyperNoise	**72.40**
+ LENS (Ours)	72.33

D.8 Aesthetic Quality Evaluation

To further evaluate perceptual detail and aesthetic quality, we conduct an additional evaluation using Q-Align [45], an LMM-based perceptual metric aligned with human visual judgments. Q-Align Quality measures overall perceptual quality, while Q-Align Aesthetic evaluates aesthetic attributes such as color, lighting, and composition. We randomly sampled 100 prompts from PartiPrompts and generated images using SD-Turbo with unmodified noise, HyperNoise, and LENS.

Table 14: Q-Align evaluation

Method	Q-Align Quality ↑	Q-Align Aesthetic ↑
SD-Turbo	4.3197	**3.0155**
+ HyperNoise	4.4339	3.0054
+ LENS	**4.4672**	3.0030

As shown in Table 14, LENS achieves the highest Q-Align Quality score, indicating improved perceptual quality. Its Q-Align Aesthetic score remains very close to those of unmodified noise and HyperNoise, differing by only 0.0125 and 0.0024, respectively. These results indicate that LENS improves perceptual quality while maintaining comparable aesthetic quality.

D.9 Reward Component Ablation

To analyze the contribution of each reward component, we conduct a leave-one-out ablation study on the four reward terms used by LENS. For each ablation, we remove one component while keeping all other training settings unchanged and train a new modulation network. All variants are evaluated on GenEval2 using the same prompt–seed pairs.

Table 15: Leave-one-out ablation of the reward components

Method	Object ↑	Attribute ↑	Count ↑	Position ↑	Verb ↑	Mean ↑	Mean drop
LENS (w/ all rewards)	74.34	42.56	23.98	28.39	7.81	35.42	–
LENS (w/o CLIP)	73.89	42.22	23.76	28.39	3.95	34.44	0.98
LENS (w/o HPS)	72.92	39.96	23.99	28.36	5.05	34.06	1.36
LENS (w/o ImageReward)	73.55	41.43	24.03	28.56	7.26	34.97	0.45
LENS (w/o PickScore)	74.22	41.96	24.26	27.29	8.52	35.25	0.17

As shown in Table 15, using all four reward components achieves the best overall GenEval2 performance. Removing HPS results in the largest drop in the mean score, followed by CLIP, ImageReward, and PickScore. Although the contribution of each component varies across individual categories, the full reward yields the highest overall performance, suggesting that the four reward terms play complementary roles.

D.10 Sensitivity to the PCA Basis Dataset

To evaluate the sensitivity of LENS to the dataset used for constructing the PCA basis, we additionally compute PCA bases from Open Images v7 [21] and Places365 [54], each using 10,000 uniformly sampled images, in addition to the ImageNetV2 basis used in the main experiments. For each basis, we train a separate LENS modulation network using the same SD-Turbo training configuration and evaluate all variants on GenEval2 with identical prompt–seed pairs.

Table 16: Sensitivity of LENS to the dataset used for constructing the PCA basis

Method	Object ↑	Attribute ↑	Count ↑	Position ↑	Verb ↑	Mean ↑
SD-Turbo	72.14	42.72	20.42	29.50	5.82	34.12
+ LENS (ImageNetV2 basis)	74.34	42.56	23.98	28.39	7.81	35.42
+ LENS (Open Images basis)	74.90	44.12	24.50	29.97	7.36	36.17
+ LENS (Places365 basis)	75.12	41.34	24.51	27.59	6.59	35.03

As shown in Table 16, LENS improves the overall GenEval2 mean over the SD-Turbo baseline for all three PCA bases. Although the scores vary slightly across individual categories, the overall improvement is preserved regardless of the dataset used to construct the PCA basis. These results indicate that LENS is not strongly sensitive to the choice of PCA basis dataset.

D.11 Standard Deviations of GenEval2 Scores

Table 1 reports GenEval2 results obtained by averaging over three runs with different base seeds to mitigate dependence on specific random seeds. Table 17 provides the corresponding standard deviations.

Table 17: Standard deviations of GenEval2 results in Table 1

Model	Object	Attribute	Count	Position	Verb	Mean
SD-Turbo	0.29	0.30	0.21	0.32	0.21	0.24
+ ReNo	0.38	0.42	0.33	0.37	0.27	0.35
+ HyperNoise	0.40	0.48	0.39	0.36	0.25	0.33
+ LENS (Ours)	0.37	0.40	0.36	0.40	0.23	0.35
Hyper-SDXL	0.40	0.43	0.32	0.29	0.20	0.32
+ ReNo	0.45	0.43	0.35	0.37	0.16	0.35
+ HyperNoise	0.40	0.44	0.38	0.39	0.25	0.32
+ LENS (Ours)	0.42	0.41	0.36	0.33	0.21	0.35
SANA-Sprint	0.47	0.44	0.41	0.46	0.56	0.42
+ ReNo	0.44	0.43	0.42	0.52	0.53	0.47
+ HyperNoise	0.43	0.44	0.38	0.50	0.74	0.45
+ LENS (Ours)	0.41	0.43	0.43	0.49	0.61	0.48

E Broader Impact

Generative models have been widely studied and increasingly adopted across various domains, leading to substantial societal impact. This work contributes to improving the quality and computational efficiency of such models, enabling higher-quality generation with reduced computational cost. In this regard, our approach may facilitate broader accessibility to generative modeling, particularly in resource-constrained settings.

At the same time, we do not expect our work to introduce fundamentally new societal risks beyond those already associated with existing generative models. However, inherent risks such as the potential misuse for generating misleading or synthetic content, including misinformation, remain relevant. These concerns are not unique to our method but are shared across the broader class of generative modeling techniques. Responsible deployment and appropriate usage guidelines are therefore important to mitigate such risks.

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: We clearly state the main claims including the contributions and scope of the work in the abstract and introduction and ensure that these claims are consistent with the results in the paper.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We clearly discuss the limitations of our work in the conclusion section of the paper.

   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 clearly state all assumptions for our theoretical results and provide complete proofs in the main paper and the appendices, with appropriate references throughout.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We clearly describe the proposed method, datasets, and experimental procedures for reproducibility, and provide detailed settings in Appendix C.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We provide the code related to the experiments in the supplementary material and plan to release it publicly after the review period.

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: Detailed experimental settings, including hyperparameters, training configurations, and implementation details, are provided in Appendix C.

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: [Yes]

Justification: For quantitative evaluations using GenEval2, results are averaged over three base seeds to reduce variance, and the corresponding standard deviations are reported in Appendix D.11.

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: Details of the compute resources used for the experiments are provided in Appendix C.

   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 confirm that this work complies 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: The broader societal impacts of our work 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: This work builds on existing open-source models and is not expected to pose significant risks of misuse.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We properly cite all existing assets, including datasets and models.

    Guidelines:

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

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

13. **New assets**

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

    Answer: [Yes]

    Justification: We will provide the code in accordance with the submission guidelines and provide clearly documented materials alongside the assets.

    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 work 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 work does not involve any research with human subjects and therefore does not require IRB approval.

    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: This work does not involve the use of LLMs as part of the core methodology. LLMs were used only for text editing and coding assistance.

    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.