Alleviating Distortion in Image Generation via Multi-Resolution Diffusion Models and Time-Dependent Layer Normalization

Qihao Liu[1,2*], Zhanpeng Zeng[1,3*], Ju He[1,2*], Qihang Yu[1], Xiaohui Shen[1], Liang-Chieh Chen[1]

[1] ByteDance [2] Johns Hopkins University [3] University of Wisconsin-Madison

* equal contribution

https://qihao067.github.io/projects/DiMR

Figure 1: (Top) Randomly sampled 512×512 images generated by the proposed DiMR. (Bottom) Random samples of the **low visual fidelity** 256×256 images generated by DiMR and DiT [41]. To detect low visual fidelity images for both models, a classifier-based rejection model is employed (with the same rejection rate). DiMR generates images with higher fidelity and less distortion than DiT.

Abstract

This paper presents innovative enhancements to diffusion models by integrating a novel multi-resolution network and time-dependent layer normalization. Diffusion models have gained prominence for their effectiveness in high-fidelity image generation. While conventional approaches rely on convolutional U-Net architectures, recent Transformer-based designs have demonstrated superior performance and scalability. However, Transformer architectures, which tokenize input data (via "patchification"), face a trade-off between visual fidelity and computational complexity due to the quadratic nature of self-attention operations concerning token length. While larger patch sizes enable attention computation efficiency, they

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-4254

struggle to capture fine-grained visual details, leading to image distortions. To address this challenge, we propose augmenting the **Di**ffusion model with the **M**ulti-**R**esolution network (DiMR), a framework that refines features across multiple resolutions, progressively enhancing detail from low to high resolution. Additionally, we introduce Time-Dependent Layer Normalization (TD-LN), a parameter-efficient approach that incorporates time-dependent parameters into layer normalization to inject time information and achieve superior performance. Our method's efficacy is demonstrated on the class-conditional ImageNet generation benchmark, where DiMR-XL variants surpass previous diffusion models, achieving FID scores of 1.70 on ImageNet 256×256 and 2.89 on ImageNet 512×512. Our best variant, DiMR-G, further establishes a state-of-the-art 1.63 FID on ImageNet 256×256.

1 Introduction

Diffusion and score-based generative models [23, 53, 55, 19, 54] have demonstrated promising results for high-fidelity image generation [7, 38, 44, 46, 48]. These models generate images through an iterative process of gradually denoising Gaussian random noise to create realistic samples Central to this process is a neural network, tasked with denoising the inputs through a mean squared error loss function. Traditionally, U-Net architectures [47] (enhanced with residual blocks [15] and self-attention blocks [59] at lower resolution) have been prevalent. However, recent advancements have introduced Transformer-based designs [59, 8], offering superior performance and scalability.

In practice, Transformer-based architectures face the challenge of balancing visual fidelity and computational complexity, primarily stemming from the self-attention operation and the patchification process employed for downsampling inputs [8] (*i.e.*, a smaller patch size results in better visual fidelity at the cost of a longer token length and thus more computational complexity by the self-attention operation). The quadratic complexity inherent in self-attention concerning token length necessitates larger patch sizes to facilitate more efficient attention computations. However, the adoption of large patch sizes inevitably compromises the model's capacity to capture finer visual details, resulting in image distortion (*i.e.*, low visual fidelity). This dilemma prompts DiT [41] to conduct a systematic study on the impact of patch size on image distortion, as depicted in Fig. 7 of their paper. Consequently, they settled on a patch size of 2 for their final design. Similarly, U-ViT [2] opted for a patch size of 2 for input sizes of 256×256 and a patch size of 4 for 512×512 images, effectively balancing the token length for different image sizes. Despite these meticulous adjustments, the generated results still exhibit discernible image distortion, as illustrated in Fig. 1.

One simplistic solution to mitigate image distortion in Transformer-based architectures is adopting a patch size of 1, but this significantly increases computational complexity. Instead, inspired by the success of *image cascade* [20, 48] which generate images at increasing resolutions, we propose a *feature cascade* approach that progressively upsamples lower-resolution features to higher resolutions, alleviating distortion in image generation. In this study, we present DiMR, which enhances the **Di**ffusion model with a **M**ulti-**R**esolution network. DiMR tackles the challenge of balancing visual detail capture and computational complexity through improvements in the denoising backbone architecture. We employ a multi-resolution network design that comprises multiple branches to progressively refine features from low to high resolution, preserving intricate details within the input data. Specifically, the first branch handling the lowest resolution incorporates Transformer blocks [59], leveraging the superior performance and scalability observed in prior works [2, 41], while the remaining branches utilize ConvNeXt blocks [35], which are efficient for high resolution features. The network processes inputs progressively from the lowest resolution, with additional features from the preceding resolution. The last branch refines features at the same spatial resolution as the input, effectively mitigating image distortion arising from the patchification.

Additionally, we observe that existing time conditioning mechanisms [42, 25, 7], such as adaptive layer normalization (adaLN) [41], are parameter-intensive. In contrast, we propose a more efficient approach, Time-Dependent Layer Normalization (TD-LN), that integrates time-dependent parameters directly into layer normalization [1], achieving superior performance with fewer parameters.

To demonstrate its effectiveness, we evaluate DiMR on the class-conditional ImageNet generation benchmark [6]. On ImageNet 64×64, DiMR-M (133M parameters) and DiMR-L (284M), without classifier-free guidance [18], achieve FID scores of 3.65 and 2.21, respectively, outperforming the

https://doi.org/10.52202/079017-4254

Transformer-based U-ViT-M/4 and U-ViT-L/4 by 2.20 and 2.05 FID. On ImageNet 256×256, DiMR-XL (505M) achieves FID scores of 4.50 without classifier-free guidance and 1.70 with classifier-free guidance. Meanwhile, DiMR-G (1.06B) further improves the FID scores to 3.56 without classifier-free guidance and 1.63 with classifier-free guidance. On ImageNet 512×512, DiMR-XL (525M) achieves FID scores of 7.93 and 2.89, without and with classifier-free guidance, respectively. These results demonstrate superior performance compared to all previous methods, despite having similar or smaller model sizes, establishing a new state-of-the-art performance. In summary, our main contributions are as follows:

1. We develop effective strategies for integrating multi-resolution networks into diffusion models, introducing the novel feature cascade approach that captures visual details and reduces image distortions in high-fidelity image generation.
2. We propose TD-LN, a simple yet effective parameter-efficient method that explicitly encodes crucial temporal information into the diffusion model for enhanced performance.
3. We introduce DiMR, a novel architecture that enhances diffusion models with the proposed multi-resolution network and the TD-LN. DiMR demonstrates superior performance on the class-conditional ImageNet generation benchmark compared to existing methods.

2 Related Work

Diffusion models. Diffusion [53, 19] and score-based generative models [23, 55], centered around a denoising network trained to progressively produce denoised variants of the input data. They have driven significant advances across various domains [34, 29, 58, 56, 62, 40, 61], particularly excelling in high-fidelity image generation tasks [38, 44, 46, 48]. Key advancements in diffusion models include the improvements in sampling methodologies [19, 54, 26] and the adoption of classifier-free guidance [18]. Latent Diffusion Models (LDMs) [46, 41, 43, 63] address the challenges of high-resolution image generation by conducting diffusion in the lower-resolution latent space via a pre-trained autoencoder [28]. In this study, our focus lies on designing the denoising network within diffusion models and examining its applicability across both pixel diffusion models and LDMs.

Architecture for diffusion models. Early diffusion models employed convolutional U-Net architectures [47] as the denoising network, which were subsequently strengthened through explorations of either computing attention [7, 39] or performing diffusion directly at multiple scales [20, 13]. Recently, Transformer-based architectures [2, 41, 14] along with other explorations [64, 57, 27] have emerged as promising alternatives, showcasing superior performance and scalability. Specifically, for Transformer-based architectures, U-ViT [2] treats all inputs, including time, condition, and noisy image patches, as tokens and employs long-skip connections between shallow and deep transformer layers inspired by U-Net. Similarly, DiT [41] leverages Vision Transformers (ViTs) [8] to systematically explore the design space under the Latent Diffusion Models (LDMs) framework, demonstrating favorable properties such as scalability, robustness, and efficiency. In this study, we introduce the Multi-Resolution Network as a new denoising architecture for diffusion models, featuring a multi-branch design where each branch is dedicated to processing a specific resolution.

Time conditioning mechanisms. Following the widespread usage of adaptive normalization [42] in GANs [3, 25], diffusion models similarly explore adaptive group normalization (AdaGN) [7] and adaptive layer normalization (AdaLN) [41] to encode the time information. These methods share the similarity in requiring computing a linear projection of the timestep, which significantly increases the parameter of the model. Recently, U-ViT [2] introduces a new strategy to simply treat time as a token and process with Transformer blocks. Even though effective, it is not feasible to treat time as input for other blocks (*e.g.*, ConvNeXt blocks [35]). In this study, we introduce Time-Dependent Layer Normalization (TD-LN), a parameter-efficient approach that explicitly encodes temporal information by incorporating time-dependent parameters into layer normalization [1].

3 Preliminary

Diffusion models [53, 19] are characterized by a forward process that gradually injects noises to destroy data $\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0)$, and a reverse process that inverts the forward process corruptions. Formally,

https://doi.org/10.52202/079017-4254

the noise injection process is formulated as a Markov chain:

$$q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}),$$

where $\boldsymbol{x}_t$ for $t \in [1 : T]$ is a family of random variables obtained by progressively injecting Gaussian noise into the data $\boldsymbol{x}_0$, and $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t|\sqrt{\alpha_t}\boldsymbol{x}_{t-1}, \beta_t\boldsymbol{I})$ represents the noise injection schedule such that $\alpha_t + \beta_t = 1$. In the reverse process, a Gaussian model $p(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}|\boldsymbol{\mu}_t(\boldsymbol{x}_t), \sigma_t^2\boldsymbol{I})$ is learned to approximate the ground truth reverse transition $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$. This step is equivalent to predicting the denoised variant of the input $\boldsymbol{x}_t$, and thus the learning objective can be further simplified to predicting the noise $\boldsymbol{\epsilon}_t$ via a noise prediction network (with parameters $\boldsymbol{\theta}$), *i.e.*, $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t)$: $\min_{\boldsymbol{\theta}} \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}_t}\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t)\|_2^2$. The condition information c can be incorporated into the learning objective when the diffusion process is guided by the class condition [7], *i.e.*, $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, c)$: $\min_{\boldsymbol{\theta}} \mathbb{E}_{t,\boldsymbol{x}_0,c,\boldsymbol{\epsilon}_t}\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, c)\|_2^2$. Traditionally, learning this objective relies on the U-Net [47], with the condition c encoded into the U-Net through various methods [2, 7, 41, 46].

Classifier-free guidance [18], an effective approach to generating high-fidelity samples, combines the score estimates from a conditional diffusion model and a jointly trained unconditional diffusion model. Formally, the classifier-free guidance encourages the sampled $\boldsymbol{x}$ to have high $p(\boldsymbol{x}|c)$ by setting: $\hat{\epsilon}_\theta(\boldsymbol{x}_t, c) = \epsilon_\theta(\boldsymbol{x}_t, \emptyset) + s \cdot \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}|c) \propto \epsilon_\theta(\boldsymbol{x}_t, \emptyset) + s \cdot (\epsilon_\theta(\boldsymbol{x}_t, c) - \epsilon_\theta(\boldsymbol{x}_t, \emptyset))$, where s is the scale of guidance, $s \geq 1$, and setting $s = 1$ becomes the standard sampling. Following prior arts [2, 41], we also exploit classifier-free guidance.

4 Method

In this section, we begin by introducing the proposed Multi-Resolution Network (Sec. 4.1), which progressively refines features from low to high resolution. Next, we detail the proposed Time-Dependent Layer Normalization (Sec. 4.2). We then discuss several micro-level design enhancements (Sec. 4.3). Finally, we present the DiMR model variants, scaled for different model sizes (Sec. 4.4).

4.1 Multi-Resolution Network

Motivation. There is a trade-off between generation quality and computational complexity as depicted in the ablation study in Fig. 7 of DiT [41]. Their careful study revealed that Transformer-based diffusion models with smaller patch sizes operate at higher feature resolutions and produce better generation quality but incur higher computational costs due to the increased input size.

We conjecture that the distortion in U-ViT [2] and DiT [41] arises from their oversimplified upsampling module, where lower-resolution feature maps are upsampled directly to the target size of the generated images via a simple linear layer (for increasing channels) and pixel shuffling upsampling [52]. Inspired by *image cascade* [20, 48]—a method for generating high-resolution images by using multiple cascaded diffusion models to produce images of progressively increasing resolution—we propose *feature cascade*, which progressively upsample lower-resolution features to higher resolutions to alleviate distortion in image generation. The *feature cascade* is implemented through the proposed Multi-Resolution Network, deployed as the denoising network in diffusion models.

Overview of multi-branch design. The proposed Multi-Resolution Network comprises R branches, where each branch is dedicated to process a specific resolution. For the r-th branch ($r \in \{1, \cdots, R\}$), the input features are processed by a convolution with a kernel size of $2^{R-r} \times 2^{R-r}$ and a stride of 2^{R-r}, which effectively patchifies the input for different resolutions. The first branch (*i.e.*, $r = 1$) downsamples the input features by a factor of 2^{R-1}, and subsequently handles the lowest resolution features via the Transformer blocks [59], which enjoys the superior performance and scalability of self-attention operations [2, 41]. For higher resolution features, the remaining branches utilize the ConvNeXt blocks [35], which leverages the efficiency of large kernel depthwise-convolution operations [22, 50]. Intermediate features from the previous branch (last block's output) are upsampled and added with the inputs for the current branch. Following U-ViT [2], all branches employ the long skip connections and an additional 3×3 convolution in the end. The final branch (*i.e.*, $r = R$) refines features at the same spatial resolution as the input.

https://doi.org/10.52202/079017-4254

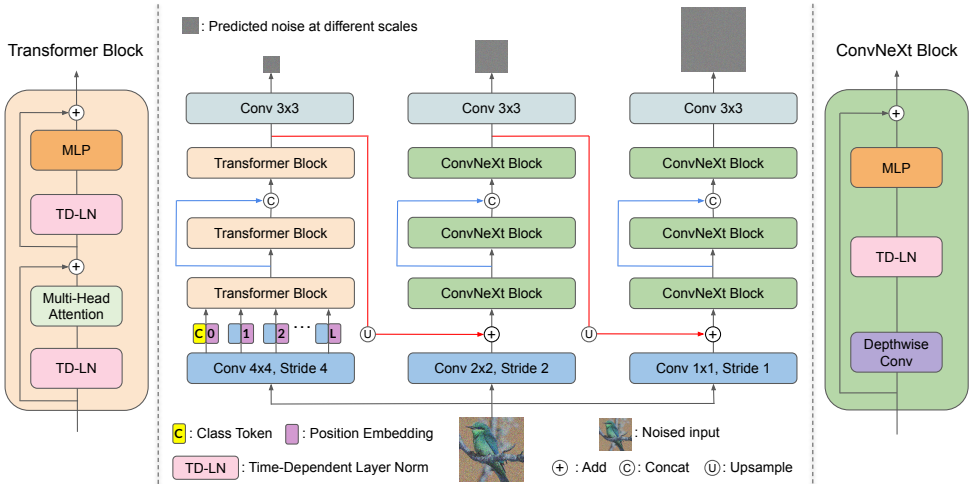

Figure 2: **Model overview.** We propose DiMR that enhances **D**iffusion models with a **M**ulti-**R**esolution Network. In the figure, we present the Multi-Resolution Network with three branches. The first branch processes the lowest resolution (4 times smaller than the input size) using powerful Transformer blocks, while the other two branches handle higher resolutions (2 times smaller than the input size and the same size as the input, respectively) using effective ConvNeXt blocks. The network employs a *feature cascade* framework, progressively upsampling lower-resolution features to higher resolutions to reduce distortion in image generation. The Transformer and ConvNeXt blocks are further enhanced by the proposed Time-Dependent Layer Normalization (TD-LN), detailed in Fig. 4.

Design details. For $r \in \{1, \cdots, R\}$, we define the r-th branch as a function $f_{\boldsymbol{\theta},r}$ as follows:

$$\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, c, r), \boldsymbol{y}_r = f_{\boldsymbol{\theta},r}(\boldsymbol{x}_t, \boldsymbol{y}_{r-1}, t, c), \tag{1}$$

where the function $f_{\boldsymbol{\theta},r}$, parameterized by $\boldsymbol{\theta}$ and r, takes as input the input features $\boldsymbol{x}_t$ and the features from previous resolution $\boldsymbol{y}_{r-1}$ (also time t and condition c). The outputs of $f_{\boldsymbol{\theta},r}$ contain the intermediate features $\boldsymbol{y}_r$ (last block's output, before the final 3×3 convolution) and the predicted noise $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, c, r)$ for the resolution specific to r-th branch.

To process the inputs, the function $f_{\boldsymbol{\theta},r}$ first patchifies the input features $\boldsymbol{x}_t$, and adds it with the upsampled features $\boldsymbol{y}_{r-1}$ from the previous resolution $r-1$. The resulting features are then processed by either a stack of Transformer blocks (when $r = 1$) or ConvNeXt blocks (when $r \neq 1$) with another 3×3 convolution added in the end. Formally, we have:

$$f_{\boldsymbol{\theta},r}(\boldsymbol{x}_t, \boldsymbol{y}_{r-1}, t, c) = \text{Conv}_{3\times3}(g_{\boldsymbol{\theta},r}(\text{Patchify}(\boldsymbol{x}_t) + \text{Upsample}(\boldsymbol{y}_{r-1}), t, c)), \tag{2}$$

where $\text{Conv}_{3\times3}$ is 3×3 convolution, Patchify is patchification instantiated via a convolution with a kernel size of $2^{R-r} \times 2^{R-r}$ and a stride of 2^{R-r}, Upsample is the pixel shuffling upsampling operation [52], and $g_{\boldsymbol{\theta},r}$ is a stack of Transformer blocks or ConvNeXt blocks, depending on r, augmented with the long skip connections [2]. For the first branch (*i.e.*, $r = 1$), $\boldsymbol{y}_0$ is set to zero. The noise prediction $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, c, R)$ at the last branch (*i.e.*, $r = R$) is used for the iterative diffusion process. We illustrate the proposed Multi-Resolution Network with three branches in Fig. 2. Note that the input features can be either raw image pixels or latent features after VAE [28], where the latent features facilitate efficient high-resolution image generation [46].

4.2 Time-Dependent Layer Normalization

Motivation. Time conditioning plays a crucial role in the diffusion process. While the ConvNeXt blocks in the Multi-Resolution Network efficiently process high-resolution features, they also present a new challenge: *How do we inject time information into ConvNeXt blocks?* To address this, we carry out a systematic ablation study (details in Tab. 2), starting with the U-ViT architecture [2], which encodes time information via an in-context conditioning mechanism, Time-Token (*i.e.*, treating time as input token to Transformer). Unlike Transformer blocks, however, it is not feasible to add a time token directly to ConvNeXt blocks, which can only process 2D features. As an alternative, we explored the adaptive normalization mechanism, particularly the adaptive layer normalization

https://doi.org/10.52202/079017-4254

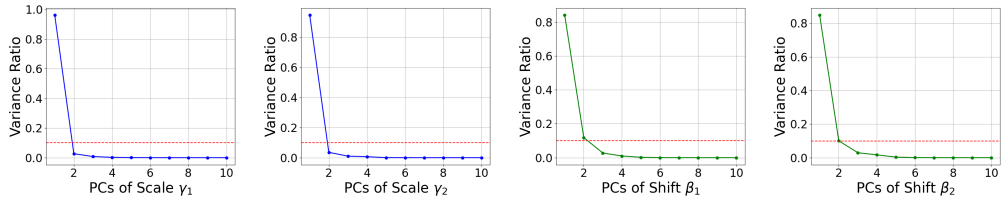

Figure 3: **Principal Component Analysis (PCA) of learned scale and shift parameters in adaLN-Zero [41].** We conduct PCA on the learned scale (γ_1, γ_2) and shift (β_1, β_2) parameters obtained from a parameter-heavy MLP in adaLN-Zero using a pre-trained DiT-XL/2 [41] model. The vertical axis represents the explained variance ratio of the corresponding Principal Components (PCs). Our observations reveal that the learned parameters can be largely explained by two principal components, suggesting the potential to approximate them by a simpler function.

AdaLN-Zero [41]. Interestingly, we found AdaLN-Zero to be more effective than Time-Token on the ImageNet 64×64 benchmark, contradicting U-ViT's findings on CIFAR-10 [30] (See Fig. 2(b) in [2]). However, AdaLN-Zero significantly increases model parameters (from 130.9M to 202.4M) due to the Multi-Layer Perceptron (MLP) used to adaptively learn the scale and shift parameters.

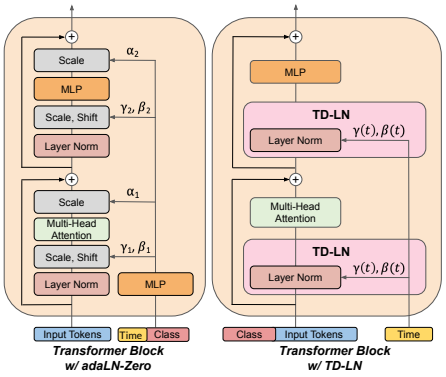

Figure 4: **Time conditioning mechanisms.** (Left) adaLN-Zero [41] learns scale and shift parameters (γ_i, β_i, α_i, $i = \{1, 2\}$) using parameter-heavy MLPs. (Right) The proposed Time-Dependent Layer Normalization (TD-LN) formulates the LN statistics as functions of time ($\gamma(t)$, $\beta(t)$), making it parameter-efficient.

To understand how time information is utilized in adaLN-Zero, we conducted Principal Component Analysis (PCA) on the learned scale (γ_1, γ_2) and shift (β_1, β_2) parameters from a parameter-heavy MLP in adaLN-Zero using a pre-trained DiT-XL/2 [41] model, as shown in Fig. 3. Intriguingly, we observed that the learned parameters can be largely explained by two principal components, suggesting that a parameter-heavy MLP might be unnecessary and that a simpler function could suffice. To address the increase in parameters, we introduce Time-Dependent Layer Normalization (TD-LN), a straightforward and lightweight method to inject time into layer normalization. We detail the designs below.

adaLN design. Building on layer normalization [1], adaLN additionally learns the scale parameter γ_1 and shift parameter β_1 via an MLP from the sum of the embedding vectors of time t and class condition c. Formally, given the input $\boldsymbol{x}$ (ignoring the dependency on t for simplicity), we have:

$$\gamma_1, \beta_1 = \text{MLP}(\text{Embed}(t) + \text{Embed}(c)), \tag{3}$$

$$\boldsymbol{z} = \gamma_1 \cdot \text{LN}(\boldsymbol{x}, \gamma, \beta) + \beta_1, \tag{4}$$

where γ_1 and β_1 scale and shift the output from the layer normalization LN, the function Embed generates the embedding vectors for time t and class condition c, and $\boldsymbol{z}$ is the output. The LN has its own learnable affine transform parameters γ and β. adaLN-Zero [41] introduces another scale parameter α_1, obtained from the same MLP for zero initialization of a residual block [12]. We note that DiT employs two sets of $(\gamma_1, \beta_1, \alpha_1)$ and $(\gamma_2, \beta_2, \alpha_2)$ in a Transformer block, as shown in Fig. 4.

TD-LN design. In contrast, our proposed method, Time-Dependent Layer Normalization (TD-LN), directly incorporates time t into layer normalization by formulating LN's learnable affine transform parameters γ and β as functions of t. Motivated by the observation that the learned parameters of adaLN-Zero can be largely explained by two principal components, we propose to model this through the linear interpolation of two learnable parameters p_1 and p_2. Formally,

$$s(t) = \text{Sigmoid}(w \cdot t + b), \tag{5}$$

$$\gamma(t) = s(t) \cdot p_1 + (1 - s(t)) \cdot p_2, \tag{6}$$

where $s(t)$ is a transformation of time t, w and b are the learnable weight and bias, and Sigmoid is the sigmoid activation function. The other affine transform parameter, $\beta(t)$, is formulated similarly with

another two parameters p_3 and p_4. Consequently, the proposed TD-LN is represented as follows:

$$\boldsymbol{z} = \text{LN}(\boldsymbol{x}, \gamma(t), \beta(t)). \tag{7}$$

Unlike adaLN, which learns additional re-scaling γ_1 and re-centering β_1 variables, TD-LN directly incorporates the time-dependent $\gamma(t)$ and $\beta(t)$ into layer normalization, eliminating the need for a parameter-heavy MLP. Furthermore, TD-LN is a versatile mechanism, enabling the injection of time information into both Transformer blocks and ConvNeXt blocks. In DiMR, we replace all layer normalizations with the proposed TD-LN, and treat the class condition c as input token for the Transformer blocks.

4.3 Micro-Level Design

In addition to the major architectural modifications discussed earlier, we also explore several micro-level design changes to enhance model performance.

Multi-scale loss. The proposed Multi-Resolution Network comprises R branches, each dedicated to processing features at a specific resolution, naturally producing multi-scale outputs. To leverage this, we explore training the network with a multi-scale loss $\mathcal{L}_{multi}$, which is a weighted sum of mean squared error loss at each resolution. Formally, the multi-scale loss is defined as follows:

$$\mathcal{L}_{multi} = \sum_{r=1}^{R} \alpha_r \cdot \mathbb{E}_{t,\boldsymbol{x}_0,c,\boldsymbol{\epsilon}_t} \|\text{Downsample}(\boldsymbol{\epsilon}_t, r) - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, c, r)\|_2^2, \tag{8}$$

where α_r is the loss weight for the r-th branch, and $\text{Downsample}(\boldsymbol{\epsilon}_t, r)$ downsamples the target noise $\boldsymbol{\epsilon}_t$ by a factor of 2^{R-r} using average pooling (the R-th branch, containing no downsampling, is our final output). We set $\alpha_r = 1/(2^{R-r} \times 2^{R-r})$, motivated by the prior work [21] which found that the signal to noise ratio increases by a factor of k^2 when the noised input is average-pooled with a $k \times k$ kernel. Intuitively, our target output (the R-th branch) has a loss weight $\alpha_R = 1$, and the loss weights for the intermediate outputs are scaled down quadratically based on the downsampling factor.

Gated linear unit. In the proposed Multi-Resolution Network, both Transformer and ConvNeXt blocks include an MLP block, consisting of two linear transformations with GeLU activation [16] in between. We also explore replacing the first linear layer with GeGLU [51], an enhanced version of the Gated Linear Unit (GLU) [5] that has $2\times$ expansion rate.

4.4 DiMR Model Variants

We now introduce the DiMR model variants, scaled appropriately for different model sizes. We present four sizes: DiMR-M (medium, 133M parameters), DiMR-L (large, 284M parameters), DiMR-XL (extra-large, around 500M parameters) and DiMR-G (giant, 1.06B parameters). Three hyperparameters—R (number of branches), N (number of layers per branch), and D (hidden size per branch)—define each DiMR variant. Specifically, R determines the number of branches in the multi-resolution network. We append 2R or 3R to the model name to indicate whether two or three branches are used. The number of layers N in the multi-resolution network is represented as a tuple of R numbers, where the r-th number specifies the number of layers in the r-th branch. Similarly, the hidden size D is also a tuple of R numbers. We follow a straightforward scaling rule: most layers are stacked in the first branch, which is processed by Transformer blocks, while the remaining branches use only half the number of layers of the first branch. Additionally, when the resolution is doubled, the hidden size is reduced by a factor of two. The model variants details are presented in Tab. 4 in Sec. B in the Appendix.

5 Experimental Results

5.1 Experimental Setup

Datasets. We consider class-conditional image generation tasks at 64×64, 256×256, and 512×512 resolutions on ImageNet-1K [6]. For images at 64×64, we train DiMR on pixel space. For images at 256×256 and 512×512, following the baselines [2, 41], we utilize an off-the-shelf pre-trained variational autoencoder [28] from Stable Diffusion [46] to extract the latent representations sized at 32×32 and 64×64, respectively. Then we train our DiMR to model these latent representations.

https://doi.org/10.52202/079017-4254

Table 1: **Class-conditional image generation on ImageNet** 256×256 **and ImageNet** 512×512**.** We report training epochs, number of parameters (#Params), GFLOPs, and FID-50K with and without Classifier-Free Guidance (CFG). Best results are marked in **bold**.

(a) **ImageNet** 256×256

Model	Epoch	#Params.	Gflops	FID (w/o CFG)↓	FID↓
ADM-U [7]	396	608M	742	7.49	3.94
LDM-4 [46]	166	400M	104	10.56	3.60
U-ViT-H/2 [2]	400	501M	133	6.58	2.29
DiT-XL/2 [41]	1399	675M	119	9.62	2.27
DiMR-XL/2R (Ours)	400	505M	160	4.87	1.77
DiMR-XL/2R (Ours)	800	505M	160	4.50	1.70
DiMR-G/2R (Ours)	800	1.06B	331	**3.56**	**1.63**

(b) **ImageNet** 512×512

Model	Epoch	#Params.	Gflops	FID (w/o CFG)↓	FID↓
ADM [7]	-	422M	-	23.24	7.72
ADM-U	1081	731M	2813	9.96	3.85
U-ViT-L/4 [2]	400	287M	77	-	4.67
U-ViT-H/4 [2]	400	501M	133	15.70	4.05
DiT-XL/2 [41]	599	675M	525	12.03	3.04
DiMR-XL/3R (Ours)	400	525M	206	8.56	3.23
DiMR-XL/3R (Ours)	800	525M	206	**7.93**	**2.89**

Evaluation. We measure the model's performance using Fréchet Inception Distance (FID) [17]. We report FID on 50K generated samples to measure the image quality (*i.e.*, FID-50K). To ensure fair comparisons, we follow the same evaluation suite as the baseliness [7, 41] to compute the FID scores. We also report Inception Score [49] and Precision/Recall [31] in Sec. D as secondary metrics.

Implementation details. We use AdamW optimizer [36] with a constant learning rate of 2×10^{-4} for most experiments, except for the 64×64 models where we use 3×10^{-4}. A batch size of 1024 is used for most architectures. For a fair comparison with DiT [41], we train the 256×256 and 512×512 models for 1M iterations, and also report results for 500K iterations to compare with U-ViT [2]. We train the 64×64 models for 300K iterations, following the U-ViT protocol. *Our training hyperparameters are almost entirely retained from U-ViT [2]. We did not tune learning rates, decay/warm-up schedules, Adam* β_1/β_2 *values, or weight decays.* Further details on hyperparameters and configurations are provided in Sec. C in the Appendix.

5.2 State-of-the-Art Diffusion Models

We compare DiMR with state-of-the-art diffusion models on ImageNet 256×256 and 512×512 in Tab. 1, and provide more comparisons with other types of generative models in Tab. 7 and Tab. 8 in the Appendix. Results on ImageNet 64×64 are reported in Tab. 6 in the Appendix. More random samples of the generated images are also presented in Fig. 7 to Fig. 18 in the Appendix.

ImageNet 256×256**.** From Tab. 1a, we observe that our DiMR-XL/2R outperforms all previous diffusion-based models and achieves a state-of-the-art FID-50K score of 1.70. Specifically, with a comparable model size and equal or fewer training epochs, our model surpasses previous state-of-the-art transformer-based diffusion models, including U-ViT[2] (1.77 *vs*. 2.29 with Classifier-Free Guidance [18] (CFG) and 4.87 *vs*. 6.58 without CFG) and DiT [41] (1.70 *vs*. 2.27 with CFG and 4.50 *vs*. 9.62 without CFG). Our best model, DiMR-G/2R, scales up to the billion-parameter level, setting a new state-of-the-art with an FID of 1.63 with CFG and 3.56 without CFG.

ImageNet 512×512**.** Our DiMR outperforms all previous diffusion-based models on ImageNet 512×512 and achieves a state-of-the-art FID-50K score of 2.89 as shown in Tab. 1b. It is worth noting that, although both Gflops and model sizes are critical for improving performance, as discussed in the DiT paper [41], we still outperform it with only 39.2% of the GFLOPs and 77.8% of the model size, improving the FID-50K from 3.04 to 2.89. As transformers and diffusion models have demonstrated good scaling behavior, we believe that further scaling up our DiMR will lead to better performance, which we have left as future work.

5.3 Alleviating Distortion

Transformer-based architectures encounter the challenge of balancing visual fidelity with computational complexity. Despite adopting a small patch size of 2, current models still struggle with distortions. To illustrate the effectiveness of DiMR in alleviating these distortions, we adopt a classifier-based rejection model following previous work [45]. However, we diverge from previous approaches by *solely* using the rejection model to analyze distorted images, rather than filtering out bad images and computing metrics only on selected 'good' images. It is important to note that all metrics in our paper are computed without using the rejection model to ensure fair comparisons.

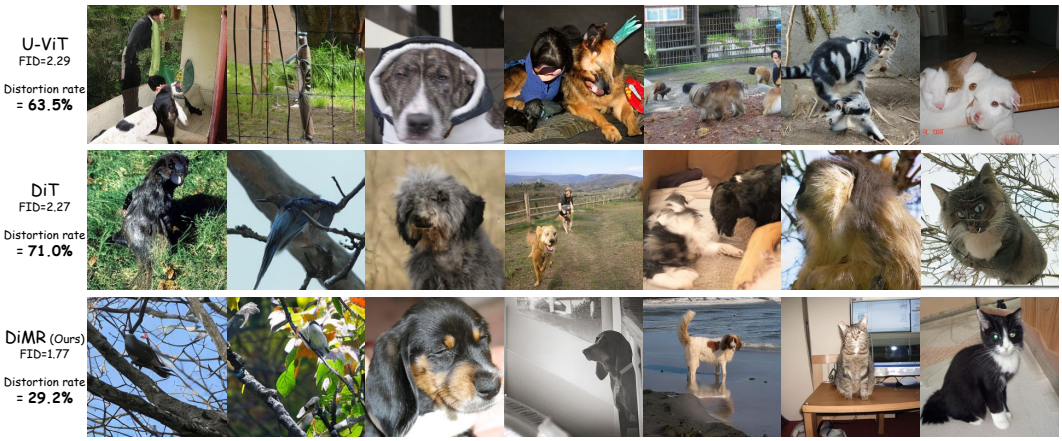

Figure 5: **DiMR alleviates distortions and improves visual fidelity.** In this figure, we randomly visualize the detected low-fidelity images, identified by a pretrained classifier, which are generated by the best models from the baselines and our DiMR. The first column reports both their FID-50K scores and the proportion of distorted images based on human evaluation. DiMR demonstrates better generation performance and lower distortion rates than the baselines.

Table 2: **Ablation study**. Beginning with the baseline, we verify the effectiveness of each component.

	Model	AdaLN-Zero [41]	TD-LN	Multi-branch	GLU [51]	Multi-scale Loss	FID(↓)	# Params.
1	Baseline (U-ViT-M/4 [2])						5.85	130.9M
2		✓					5.44	202.4M
3		✓		✓			7.91	217.9M
4			✓	✓			5.21	154.0M
5			✓	✓	✓		4.86	132.9M
6	DiMR-M/3R (Ours)		✓	✓	✓	✓	3.65	132.9M

Specifically, we randomly generate 80K images for each model and utilize a pretrained Vision Transformer classifier [8] to identify low-fidelity images based on the predicted probabilities. Images with a probability below a threshold of 0.2 are considered low-fidelity or potentially distorted. Fig. 5 shows random samples of low-fidelity images detected by the classifier. However, we find that not all detected images are distorted; many are classified with low probability due to classifier errors. To accurately identify distorted images among those detected by the classifier, we conduct user studies where human evaluators manually assess the images. Images generated by all three methods are merged and presented, along with their corresponding class labels, to human evaluators, who are instructed to determine whether each image is distorted (*i.e.*, identify low-fidelity images). Each image is evaluated by five different human evaluators. We consider the proportion of distorted images generated by different models, *i.e.* distortion rate. We compute three distortion rates, one for each model, from each evaluator based on the images they evaluate. The final distortion rate for each model is obtained by averaging the rates from all evaluators. As reported in Fig. 5, we observe that even among those low-fidelity images, only 29.2% of the images generated by DiMR are distorted, while previous methods yield much higher distortion rates of 63.5% and 71.0%.

5.4 Ablation Studies

We conduct the primary ablation experiments on ImageNet 64×64, progressively building on the baseline U-ViT-M/4 [2] to validate the effectiveness of the proposed designs, leading to our final model, DiMR-M/3R, as presented in Tab. 2. Additionally, we explore alternative design choices on ImageNet 256×256 with DiMR-XL/2R, including adopting a pure convolutional architecture, replacing addition with concatenation in feature cascading, and introducing skip connections between branches, as shown in Tab. 3.

AdaLN-Zero *vs*. TD-LN. Since the time token used in U-ViT cannot be adopted for ConvNeXt blocks, we first apply AdaLN-Zero [41] to the original U-ViT and our multi-branch network. As observed in row 2 of Tab. 2, AdaLN-Zero slightly improves the performance of U-ViT from 5.85 to

https://doi.org/10.52202/079017-4254

Table 3: **Design choices.** We empirically experiment with different design choices in model architecture (Tab 3a), feature cascade (Tab 3b) and skip-connection (Tab 3c). We report FID-50K scores with classifier-free guidance (CFG) after 400 training epochs.

(a) **Pure Conv v.s. Hybrid**

Model	1st branch	FID↓
DiMR-XL/2R	ConvNeXt	2.09
	Transformer	1.77

(b) **Concatenation v.s. Addition**

Model	feature cascade	FID↓
DiMR-XL/2R	Concatenation	2.06
	Addition	1.77

(c) **w/ v.s. w/o Skip-Connection**

Model	skip-connection	FID↓
DiMR-XL/2R	✓	1.96
		1.77

5.44. However, it does not perform well on ConvNeXt blocks and thus decreases the performance from 5.44 to 7.91 (row 3). Additionally, AdaLN-Zero significantly increases the model size from 130.9M to 202.4M. In contrast, our TD-LN is more flexible and parameter-efficient: it efficiently provides time information to both Transformer blocks and ConvNeXt blocks, improving the FID-50K score from 7.91 to 5.21 (row 4), and also reduces the model size from 217.9M to 154.0M.

GLU further reduces model size. In Tab. 2, row 5 shows the improvement of GLU compared with the vanilla MLP block. We observe that using GLU slightly improves the performance from 5.21 to 4.86 and further reduces the model size from 154.0M to 132.9M.

Multi-scale loss is critical for multi-resolution network. Training a multi-resolution network presents additional challenges and can result in sub-optimal results. In Tab. 2, row 6 illustrates that our multi-scale loss significantly enhances the performance, achieving a FID-50K score of 3.65.

Multi-branch design improves visual fidelity and alleviates distortions in image generations. Finally, comparing the multi-branch design in row 6 (incorporating TD-LN, GLU, and multi-scale loss to facilitate training) with the baseline in row 1 reveals a significant improvement in FID-50K, from 5.85 to 3.65, with just a 1.5% increase in model size (130.9M to 132.9M). Additionally, from Fig. 5, it's evident that the multi-branch design generates images with higher fidelity and less distortion.

Transformer is essential for low-resolution processing. As shown in Table 3a, replacing the Transformer blocks in the 1st (lowest-resolution) branch with ConvNeXt blocks results in a DiMR variant that uses only convolutional layers. However, this configuration performs worse compared to combining Transformer blocks with ConvNeXt blocks across different resolutions. This indicates that Transformers are more effective at capturing fine-grained details, while their usage at the lowest resolution maintains a manageable computational cost.

Simple addition suffices for multi-resolution feature cascading. As shown in Table 3b, a straightforward addition operation effectively transfers information from lower-resolution features to higher-resolution features. Replacing addition with concatenation leads to slightly worse results. We also validate the necessity of adding skip-connection between branches. As shown in Table 3c, introducing skip-connection not only degrades performance but also complicates the model architecture. Therefore, we adopt a simple upsampling followed by an addition operation for feature cascading.

6 Conclusion

In this work, we introduce DiMR, which enhances diffusion models through the Multi-Resolution Network, progressively refining features from low to high resolutions and effectively reducing image distortion. Additionally, DiMR incorporates the proposed parameter-efficient Time-Dependent Layer Normalization (TD-LN), further improving image generation quality. The effectiveness of DiMR has been demonstrated on the popular class-conditional ImageNet generation benchmark, outperforming prior methods and setting new state-of-the-art performance on diffusion-style generative models. We hope that DiMR will inspire future designs of both denoising networks and time conditioning mechanisms, paving the way for even more advanced image generation models.

Acknowledgement: We thank Xueqing Deng and Peng Wang for their valuable discussion during Zhanpeng's internship.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *CVPR*, 2023.

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[4] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *CVPR*, 2022.

[5] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *ICML*, 2017.

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

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[9] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021.

[10] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Masked diffusion transformer is a strong image synthesizer. In *ICCV*, 2023.

[11] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Mdtv2: Masked diffusion transformer is a strong image synthesizer. *arXiv preprint arXiv:2303.14389*, 2023.

[12] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[13] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Joshua M Susskind, and Navdeep Jaitly. Matryoshka diffusion models. In *ICLR*, 2023.

[14] Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. Diffit: Diffusion vision transformers for image generation. In *ECCV*, 2024.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[16] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 2017.

[18] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

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

[20] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *JMLR*, 23(47):1–33, 2022.

[21] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *ICML*, 2023.

[22] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

 https://doi.org/10.52202/079017-4254

[23] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *JMLR*, 6(4), 2005.

[24] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. In *CVPR*, 2023.

[25] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

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

[27] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Yuhta Takida, Naoki Murata, Toshimitsu Uesaka, Yuki Mitsufuji, and Stefano Ermon. Pagoda: Progressive growing of a one-step generator from a low-resolution diffusion teacher. *arXiv preprint arXiv:2405.14822*, 2024.

[28] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[29] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

[30] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[31] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *NeurIPS*, 2019.

[32] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *CVPR*, 2022.

[33] Tianhong Li, Dina Katabi, and Kaiming He. Self-conditioned image generation via generating representations. *arXiv preprint arXiv:2312.03701*, 2023.

[34] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *NeurIPS*, 2022.

[35] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022.

[36] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[37] Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *ECCV*, 2024.

[38] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[39] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, 2021.

[40] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. *arXiv preprint arXiv:2205.07460*, 2022.

[41] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

[42] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

[43] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[44] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[45] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *NeurIPS*, 2019.

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

[47] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[48] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 2022.

[49] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *NeurIPS*, 2016.

[50] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

[51] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[52] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016.

[53] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

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

[55] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *NeurIPS*, 2019.

[56] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *NeurIPS*, 2021.

[57] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *arXiv preprint arXiv:2404.02905*, 2024.

[58] Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, Karsten Kreis, et al. Lion: Latent point diffusion models for 3d shape generation. *NeurIPS*, 2022.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

[60] Mark Weber, Lijun Yu, Qihang Yu, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. Maskbit: Embedding-free image generation via bit tokens. *arXiv preprint arXiv:2409.16211*, 2024.

[61] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In *CVPR*, 2023.

[62] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

[63] Zeyue Xue, Guanglu Song, Qiushan Guo, Boxiao Liu, Zhuofan Zong, Yu Liu, and Ping Luo. Raphael: Text-to-image generation via large mixture of diffusion paths. *NeurIPS*, 2023.

[64] Jing Nathan Yan, Jiatao Gu, and Alexander M Rush. Diffusion models without attention. In *CVPR*, 2024.

[65] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

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

 https://doi.org/10.52202/079017-4254

Appendix

In the appendix, we provide additional information as listed below:

- Sec. A provides the dataset information and licenses.
- Sec. B provides the DiMR model variants, scaled appropriately for different model sizes.
- Sec. C provides the implementation details of DiMR.
- Sec. D provides more comparison with other methods for class-conditional image generation on ImageNet 64×64, ImageNet 256×256, and ImageNet 512×512.
- Sec. E provides detailed introduction and more results of the Principal Component Analysis (PCA) on the scale and shift parameters learned in adaLN-Zero.
- Sec. F provides more generated image samples by DiMR.
- Sec. G discusses the limitations of our method.
- Sec. H discusses the positive societal impacts of our method.
- Sec. I discusses the potential risk of our method and safeguards that will be put in place for responsible release of our models.

A Datasets Information and Licenses

ImageNet: The ImageNet [6] dataset, containing 1,281,167 training and 50,000 validation images from 1,000 different classes, is a standard benchmark for image classification and class-conditional image generation. For the task of class-conditional image generation, the images are typically resized to a specified size, *e.g.*, 64×64, 256×256, or 512×512.

License: Custom License, non-commercial. https://image-net.org/accessagreement

Dataset website: https://image-net.org/

B DiMR Model Variants

We introduce the DiMR model variants, scaled appropriately for different model sizes. We present four sizes: DiMR-M (medium, 132.7M parameters), DiMR-L (large, 284.0M parameters), DiMR-XL (extra-large, around 500M parameters), and DiMR-G (giant, 1.06B parameters) Three hyperparameters—R (number of branches), N (number of layers per branch), and D (hidden size per branch)—define each DiMR variant. Specifically, R determines the number of branches in the multi-resolution network. We append 2R or 3R to the model name to indicate whether two or three branches are used. The number of layers N in the multi-resolution network is represented as a tuple of R numbers, where the r-th number specifies the number of layers in the r-th branch. Similarly, the hidden size D is also a tuple of R numbers. We follow a straightforward scaling rule: most layers are stacked in the first branch, which is processed by Transformer blocks, while the remaining branches use only half the number of layers of the first branch. Additionally, when the resolution is doubled, the hidden size is reduced by a factor of two. The model variants are illustrated in Tab. 4.

Table 4: **DiMR family**. The specific configuration of a DiMR variant is determined by the hyperparameters R (number of branches), N (number of layers per branch), and D (hidden size per branch).

model	input size	latent size	#branches R	#layers N	hidden size D	#params
DiMR-M/3R	64×64	-	3	(15, 8, 8)	(768, 384, 192)	133M
DiMR-L/3R	64×64	-	3	(33, 17, 17)	(768, 384, 192)	284M
DiMR-XL/2R	256×256	32×32	2	(39, 20)	(960, 480)	505M
DiMR-XL/3R	512×512	64×64	3	(39, 20, 20)	(960, 480, 240)	525M
DiMR-G/2R	256×256	32×32	2	(57, 29)	(1152, 576)	1.06B

Table 5: **Experimental setup of DiMR.** Experimental settings for all DiMR variants, including model architectures, training hyperparameters, training costs, and sampler information.

Model	DiMR-M/3R	DiMR-L/3R	DiMR-XL/2R	DiMR-XL/3R	DiMR-G/2R
Parameters	133M	284M	505M	525M	1.06B
Image Resolution	64×64	64×64	256×256	512×512	256×256
Latent space	✗	✗	✓	✓	✓
Latent shape	-	-	$32 \times 32 \times 4$	$64 \times 64 \times 4$	$32 \times 32 \times 4$
Image decoder	-	-	sd-vae-ft-ema	sd-vae-ft-ema	sd-vae-ft-ema
Number of branches R	3	3	2	3	2
Blocks in 1st branch	Transformer	Transformer	Transformer	Transformer	Transformer
# Layers	15	33	39	39	57
# Dimensions	768	768	960	960	1152
# Heads	12	12	16	16	16
Resolution	16×16	16×16	16×16	16×16	16×16
Loss Coeffs.	$1/16$	$1/16$	$1/4$	$1/16$	$1/4$
Blocks in 2nd branch	ConvNeXt	ConvNeXt	ConvNeXt	ConvNeXt	ConvNeXt
# Layers	8	17	20	20	29
# Dimensions	384	384	480	480	576
Kernel size	7×7	7×7	7×7	7×7	7×7
Resolution	32×32	32×32	32×32	32×32	32×32
Loss Coeffs.	$1/4$	$1/4$	1	$1/4$	1
Blocks in 3rd branch	ConvNeXt	ConvNeXt	-	ConvNeXt	-
# Layers	8	17	-	20	-
# Dimensions	192	192	-	240	-
Kernel size	7×7	7×7	-	7×7	-
Resolution	64×64	64×64	-	64×64	-
Loss Coeffs.	1	1	-	1	-
Batch size	1024	1024	1024	1024	1024
Training iterations	300K	300K	1M	1M	1M
Warm-up steps	5K	5K	5K	5K	5K
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
learning rate	3×10^{-4}	3×10^{-4}	2×10^{-4}	2×10^{-4}	2×10^{-4}
Weight decay	0.03	0.03	0.03	0.03	0.03
Betas	$(0.99, 0.99)$	$(0.99, 0.99)$	$(0.99, 0.99)$	$(0.99, 0.99)$	$(0.99, 0.99)$
Training devices	8 A100	16 A100	16 A100	16 A100	32 A100
Training time	62 hours	80 hours	172 hours	288 hours	400 hours
Sampler	DPM-Solver	DPM-Solver	DPM-Solver	DPM-Solver	DPM-Solver
Sampling steps	50	50	250	250	250

C Implementation Details

We use the AdamW optimizer [36] with a constant learning rate of 2×10^{-4} for most experiments, except for the 64×64 models where we use 3×10^{-4}. We set the weight decay to 0.03 and the betas to (0.99, 0.99) for all experiments. A batch size of 1024 is used for all architectures. For a fair comparison with DiT [41], we train the 256×256 and 512×512 models for 1M iterations, and we also report results for 500K iterations to compare with U-ViT [2]. We train the 64×64 models for 300K iterations, following the U-ViT protocol. All experiments use 5K steps for warm-up. We present the detailed experimental setup for all DiMR variants in Tab. 5.

D Additional Experimental Results

We present the full results of the proposed DiMR compared to other methods on ImageNet [6] in terms of Fréchet Inception Distance (FID) [17], Inception Score (IS) [49], and Precision/Recall [31]. The comparisons are made on class-conditional image generation without classifier-free guidance on ImageNet 64×64 in Tab. 6, and with classifier-free guidance [18] on ImageNet 256×256 in Tab. 7 and ImageNet 512×512 in Tab. 8.

https://doi.org/10.52202/079017-4254

Table 6: **Class-conditional image generation on ImageNet** 64×64 **(w/o classifier-free guidance)**. Metrics include Fréchet Inception Distance (FID), Inception Score (IS), Precision, and Recall, where "↓" or "↑" indicate whether lower or higher values are better, respectively. "Type": the type of the generative model. "Epoch": the number of epochs trained on ImageNet [6]. "#Params": the number of parameters in the model. "#Gflops": the computational cost. "Diff.": Diffusion models.

Model	Type	Epoch	#Params.	Gflops	FID(↓)	IS(↑)	Precision(↑)	Recall(↑)
U-ViT-M/4 [2]	Diff.	240	131M	35	5.85	33.71	0.69	0.61
U-ViT-L/4 [2]	Diff.	240	287M	77	4.26	40.66	0.71	0.62
DiMR-M/3R (Ours)	Diff.	240	133M	54	3.65	42.41	0.74	0.59
DiMR-L/3R (Ours)	Diff.	240	284M	111	2.21	55.73	0.75	0.60

Table 7: **Class-conditional image generation on ImageNet** 256×256 **(with classifier-free guidance)**. Metrics include Fréchet Inception Distance (FID), Inception Score (IS), Precision, and Recall, where "↓" or "↑" indicate whether lower or higher values are better, respectively. We report results of GAN-based models (GAN), BERT-style masked-prediction models (Mask.), autoregressive models (AR), visual autoregressive models (VAR), and diffusion based models (Diff.). "Type": the type of the generative model. "Epoch": the number of epochs trained on ImageNet [6]. "#Params": the number of parameters in the model. "#Gflops": the computational cost. "-re": the models utilize rejection sampling. "Mask. + Diff.": the models using masked-prediction to improve diffusion models.

Model	Type	Epoch	#Params.	Gflops	FID(↓)	IS(↑)	Precision(↑)	Recall(↑)
BigGAN [3]	GAN	-	112M	-	6.95	224.5	0.89	0.38
GigaGAN [24]	GAN	-	569M	-	3.45	225.5	0.84	0.61
MaskGIT [4]	Mask.	300	227M	-	6.18	182.1	0.80	0.51
MaskGIT-re [4]	Mask.	300	227M	300	4.02	355.6	-	-
RCG [33]	Mask.	200	502M	-	3.49	215.5	-	-
TiTok-S-128 [66]	Mask.	800	287M	-	1.97	281.8	-	-
MDT-G [10]	Mask. + Diff.	1299	676M	119	1.79	283.0	0.81	0.61
MDTv2-G [11]	Mask. + Diff.	919	675M	119	1.58	314.7	0.79	0.65
MaskBit [60]	Mask.	1080	305M	-	1.52	328.6	-	-
VQGAN [9]	AR	100	1.4B	-	15.78	74.3	-	-
VQGAN-re [9]	AR	100	1.4B	-	5.20	280.3	-	-
ViTVQ [65]	AR	100	1.7B	-	4.17	175.1	-	-
ViTVQ-re [65]	AR	100	1.7B	-	3.04	227.4	-	-
RQTran [32]	AR	50	3.8B	-	7.55	134.0	-	-
RQTran-re [32]	AR	50	3.8B	-	3.80	323.7	-	-
VAR-*d*16 [57]	VAR	200	310M	-	3.60	257.5	0.85	0.48
VAR-*d*20 [57]	VAR	250	600M	-	2.95	306.1	0.84	0.53
VAR-*d*24 [57]	VAR	350	1.0B	-	2.33	320.1	0.82	0.57
VAR-*d*30 [57]	VAR	350	2.0B	-	1.97	334.7	0.81	0.61
VAR-*d*30-re [57]	VAR	350	2.0B	-	1.80	356.4	0.83	0.57
ADM-G [7]	Diff.	396	554M	-	4.59	186.7	0.82	0.52
ADM-G, ADM-U [7]	Diff.	208	608M	742	3.94	215.8	0.83	0.53
CDM [20]	Diff.	2158	-	-	4.88	158.7	-	-
LDM-4 [46]	Diff.	166	400M	-	3.60	247.7	-	-
DiT-L/2 [41]	Diff.	1399	458M	81	5.02	167.2	0.75	0.57
DiT-XL/2 [41]	Diff.	1399	675M	119	2.27	278.2	0.83	0.57
U-ViT-L/2 [2]	Diff.	240	287M	77	3.40	219.9	0.83	0.52
U-ViT-H/2 [2]	Diff.	400	501M	133	2.29	263.9	0.82	0.57
DiffuSSM-XL [64]	Diff.	515	673M	280	2.28	259.1	0.86	0.56
SiT-XL [37]	Diff.	1399	675M	119	2.06	270.3	0.82	0.59
DiffiT [14]	Diff.	400	561M	114	1.73	276.5	0.80	0.62
DiMR-XL/2R (Ours)	Diff.	400	505M	160	1.77	285.7	0.79	0.62
DiMR-XL/2R (Ours)	Diff.	800	505M	160	1.70	289.0	0.79	0.63
DiMR-G/2R (Ours)	Diff.	800	1.1B	331	1.63	292.5	0.79	0.63

ImageNet 64×64**.** We follow the exact experimental setup of U-ViT [2] for class-conditional image generation on ImageNet 64×64 without classifier-free guidance to verify the effectiveness of our proposed backbone. Therefore, we focus solely on comparing against U-ViT on this benchmark. As shown in Tab. 6, when both are trained for 240 epochs, the proposed DiMR-M/3R with 133M parameters achieves an FID of 3.65 and an IS of 42.41, improving upon the counterpart U-ViT-M/4 with 131M parameters by 2.20 in FID and 8.70 in IS. For the larger model, DiMR-L/3R with 284M parameters outperforms U-ViT-L/4 with 287M parameters by 2.05 in FID and 15.07 in IS. These

Table 8: **Class-conditional image generation on ImageNet** 512×512 **(with classifier-free guidance)**. Metrics include Fréchet Inception Distance (FID), Inception Score (IS), Precision, and Recall, where "↓" or "↑" indicate whether lower or higher values are better, respectively. We report results of GAN-based models (GAN), BERT-style masked-prediction models (Mask.), autoregressive models (AR), visual autoregressive models (VAR), and diffusion based models (Diff.). "Type": the type of the generative model. "Epoch": the number of epochs trained on ImageNet [6]. "#Params": the number of parameters in the model. "#Gflops": the computational cost.

Model	Type	Epoch	#Params.	Gflops	FID(↓)	IS(↑)	Precision(↑)	Recall(↑)
BigGAN [3]	GAN	-	158M	-	8.43	177.9	0.88	0.29
MaskGIT [4]	Mask.	300	227M	-	7.32	156.0	0.78	0.50
MaskGIT-re [4]	Mask.	300	227M	-	4.46	342.0	-	-
VAR-*d*36-s [57]	VAR	350	2.35B	-	2.63	303.2	-	-
ADM-G [7]	Diff.	-	422M	-	7.72	172.7	0.87	0.42
ADM-G, ADM-U [7]	Diff.	1081	731M	2813	3.85	221.7	0.84	0.53
DiT-XL/2 [41]	Diff.	599	675M	525	3.04	240.8	0.84	0.54
U-ViT-L/4 [2]	Diff.	400	287M	77	4.67	213.3	0.87	0.45
U-ViT-H/4 [2]	Diff.	400	501M	133	4.05	263.8	0.84	0.48
DiffuSSM-XL [64]	Diff.	236	673M	1066	3.41	255.0	0.85	0.49
DiffiT [14]	Diff.	800	561M	-	2.67	252.1	0.83	0.55
DiMR-XL/3R (Ours)	Diff.	400	525M	206	3.23	285.1	0.82	0.54
DiMR-XL/3R (Ours)	Diff.	800	525M	206	2.89	289.8	0.83	0.55

consistent and significant improvements demonstrate the capability of the proposed Multi-Resolution Network and TD-LN in enhancing diffusion models to generate high-fidelity images.

ImageNet 256×256**.** We compare DiMR with state-of-the-art generative models on ImageNet 256×256 with classifier-free guidance in Tab. 7. Compared to U-ViT [2] in a fair setting, our DiMR-XL/2R with 505M parameters, trained for 400 epochs, significantly outperforms U-ViT-H/2 with 501M parameters, also trained for 400 epochs, by 0.52 in FID and 21.8 in IS. In comparison with the recently popular diffusion model DiT [41], DiMR-XL/2R trained for 800 epochs consistently outperforms DiT-L/2 with 458M parameters by 3.32 in FID and 121.8 in IS. DiMR-XL/2R even surpasses the larger variant DiT-XL/2 with 675M parameters by 0.57 in FID and 11.8 in IS. Notably, DiT models require training for 1399 epochs, while DiMR-XL/2R achieves superior performance with only 800 epochs. Furthermore, scaling up to DiMR-G/2R sets a new state-of-the-art for ImageNet 256×256 image generation, achieving an FID of 1.63 and an IS of 292.5.

ImageNet 512×512**.** We compare DiMR with state-of-the-art generative models on ImageNet 512×512 with classifier-free guidance in Tab. 8. Under the same training setting for 400 epochs, DiMR-XL/3R with 525M parameters outperforms U-ViT-H/4 with 501M parameters by 0.82 in FID and 21.3 in IS. Compared to DiT-XL/2 with 675M parameters, DiMR-XL/3R shows slight improvements of 0.15 in FID and 49.0 in IS. Overall, DiMR-XL/3R demonstrates performance comparable to other state-of-the-art generative models for ImageNet 512×512 image generation.

E Additional PCA of Learned Scale and Shift Parameters in adaLN-Zero

We conduct PCA on the learned scale (γ_1, γ_2) and shift (β_1, β_2) parameters obtained from a parameter-heavy MLP in adaLN-Zero using a pre-trained DiT-XL/2 [41] model with 28 layers in depth. To conduct the analysis, we utilize the pre-trained DiT-XL/2 to generate images and collected the scale and shift parameters (tensors) produced by the MLP at different layers along the sampling steps. PCA is then performed on the collected tensors at each layer separately. The results are presented in Fig. 6, where each row of the figure displays the analysis result at different depths, from top to bottom: 7, 14, 21, 28. The vertical axis represents the explained variance ratio of the corresponding Principal Components (PCs). As observed, in most cases, the most important principal component can explain most of the variance, while starting from the 3rd principal component, it usually only accounts for less than 5% of the variance. Our observations reveal that the learned parameters, regardless of whether produced by an MLP at a shallower layer or a deeper layer, can be largely explained by two principal components, suggesting the potential to approximate them by a simpler function, TD-LN, where the linear interpolation of two learnable parameters is learned as introduced in Sec. 4.2.

https://doi.org/10.52202/079017-4254

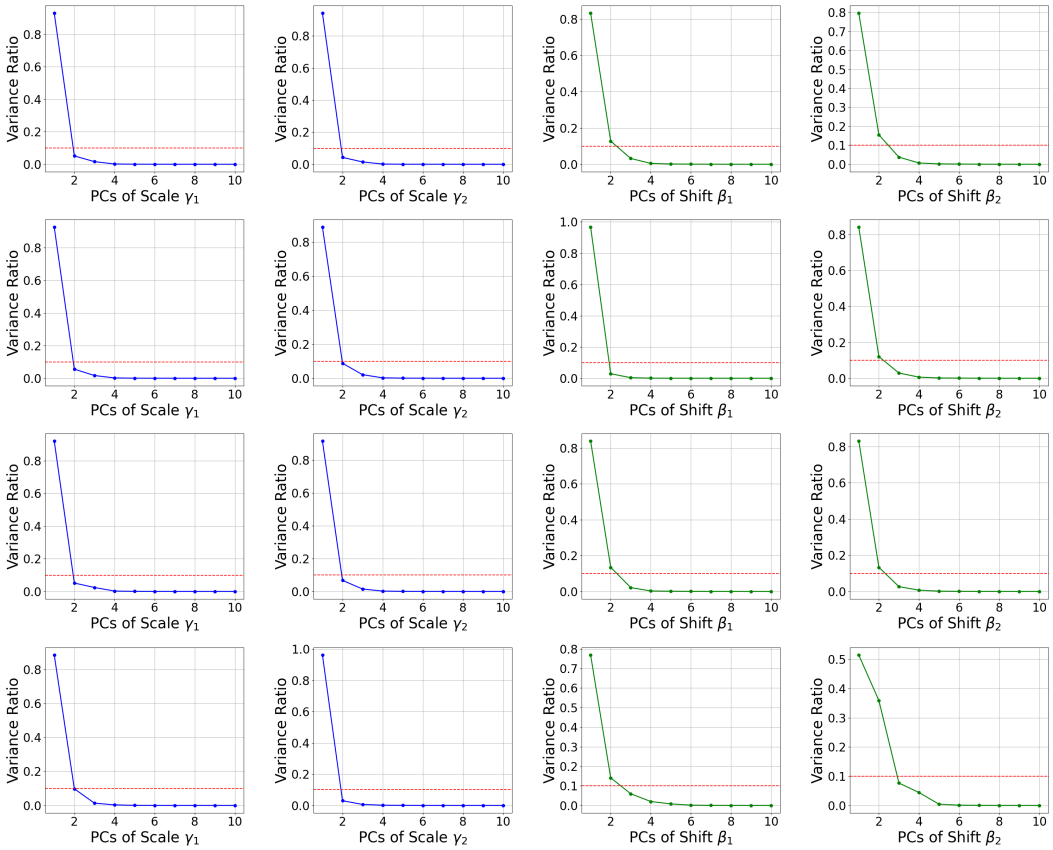

Figure 6: **Principal Component Analysis (PCA) of learned scale and shift parameters in adaLN-Zero [41].** We conduct PCA on the learned scale (γ_1, γ_2) and shift (β_1, β_2) parameters obtained from a parameter-heavy MLP in adaLN-Zero using a pre-trained DiT-XL/2 [41] model with 28 layers in depth. Each row of the figure presents the analysis result at different depths, from top to bottom: 7, 14, 21, 28. The vertical axis represents the explained variance ratio of the corresponding Principal Components (PCs). Our observations reveal that the learned parameters can be largely explained by two principal components, suggesting the potential to approximate them by a simpler function.

F Model Samples

We present samples from our largest variant, DiMR-XL/3R, at 512 × 512 resolution trained for 800 epochs. Fig. 7- 18 display uncurated samples from the model across a range of input class labels with classifier-free guidance. It is worth noting that our generated image samples exhibit high-quality and minimal image distortions.

G Limitations

The proposed DiMR has a few remaining limitations. First, it focuses on class-conditional image generation, rather than full text-to-image generation approaches. Additionally, although DiMR demonstrates great scalability in our experiments, the exploration of DiMR variants stops at the DiMR-XL/3R model with 524.8M parameters due to constraints on computational resources. In contrast, a few recent methods have scaled image diffusion models to billions of parameters. We leave it as future work to further explore the scaling law of our proposed DiMR to enhance its image generation capabilities.

Figure 7: **Uncurated** 512 × 512 **DiMR samples.** Class label = ‘arctic wolf’ (270)

Figure 8: **Uncurated** 512 × 512 **DiMR samples.** Class label = ‘volcano’ (980)

Figure 9: **Uncurated** 512 × 512 **DiMR samples.** Class label = ‘husky’ (250)

https://doi.org/10.52202/079017-4254

Figure 10: **Uncurated** 512×512 **DiMR samples.** Class label = 'sulphur-crested cockatoo' (89)

Figure 11: **Uncurated** 512×512 **DiMR samples.** Class label = 'cliff drop-off' (972)

Figure 12: **Uncurated** 512×512 **DiMR samples.** Class label = 'balloon' (417)

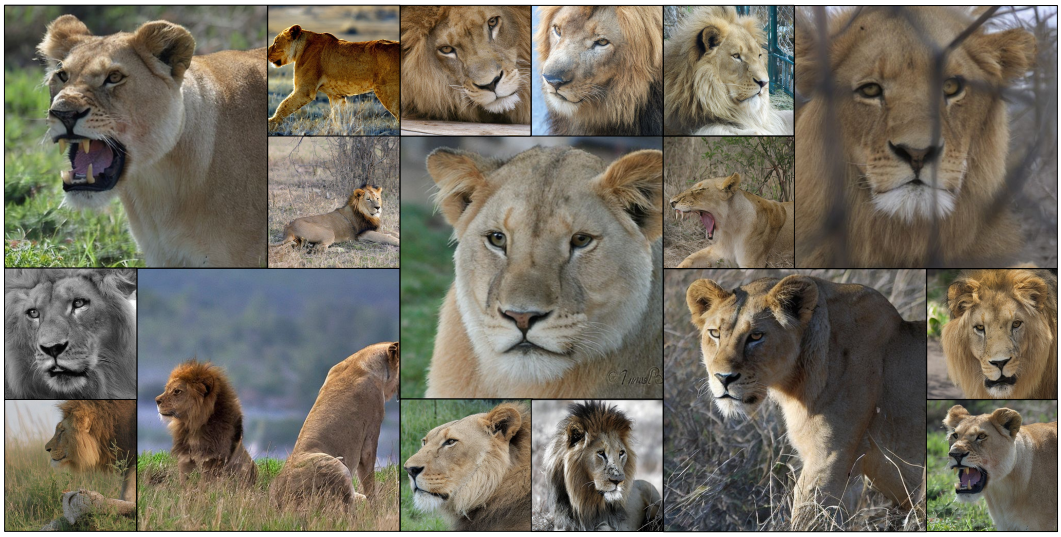

Figure 13: **Uncurated** 512 × 512 **DiMR samples.** Class label = 'lion' (291)

Figure 14: **Uncurated** 512 × 512 **DiMR samples.** Class label = 'otter' (360)

Figure 15: **Uncurated** 512 × 512 **DiMR samples.** Class label = 'red panda' (387)

https://doi.org/10.52202/079017-4254

Figure 16: **Uncurated** 512 × 512 **DiMR samples.** Class label = ‘panda’ (388)

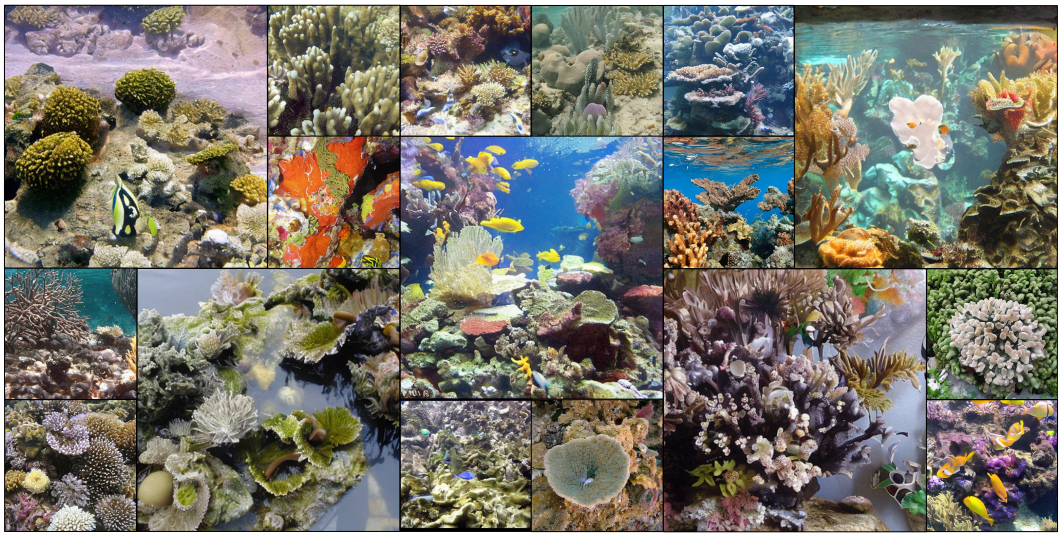

Figure 17: **Uncurated** 512 × 512 **DiMR samples.** Class label = ‘coral reef’ (973)

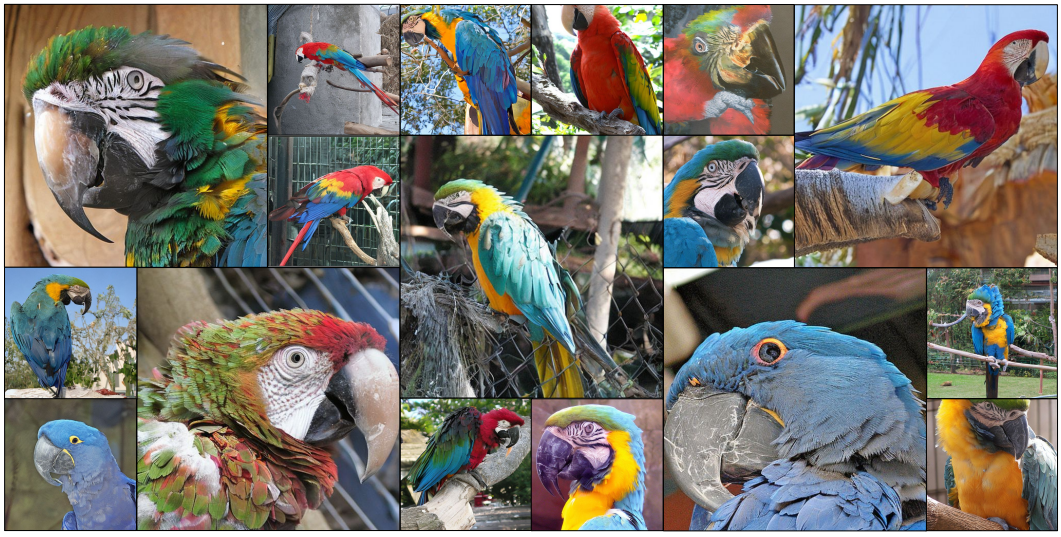

Figure 18: **Uncurated** 512 × 512 **DiMR samples.** Class label = ‘macaw’ (88)

H Broader Impacts

The proposed DiMR has the potential to facilitate numerous fields through its advanced image generation capabilities. In the realm of creative industries, DiMR can enhance the efficiency and creativity of artists and designers by generating high-fidelity images with fewer distortions. The high-quality generated images can also contribute to research on synthetic datasets by creating realistic images, aiding in reducing the annotations required for training vision models. However, with these advancements come ethical considerations, such as the risk of generating deepfakes or other malicious content. It is thus crucial to implement safeguards to minimize potential harms.

I Safety Concerns and Safeguards

Given the powerful capabilities of DiMR, it is essential to implement robust safeguards to address potential safety and ethical concerns. One primary concern is the misuse of generated content, such as the creation of deepfakes, which can lead to misinformation and privacy violations. To mitigate this, it is important to establish strict access controls and usage policies to prevent the misuse of these models when released. Transparency in the training data and model architecture is also critical to ensure accountability and to identify potential biases that could lead to harmful outputs. By prioritizing these safeguards, we can ensure the responsible use of DiMR while minimizing potential risks.

https://doi.org/10.52202/079017-4254

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: See Sec. 1 for the main claims of the paper and Sec. 4 and Sec. 5 for their illustration and validation.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: See Sec. G for the discussion on the limitations.

   Guidelines:

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

3. **Theory Assumptions and Proofs**

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

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental Result Reproducibility**

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

   Answer: [Yes]

   Justification: See Sec. 5.1 and Tab. 5 for all the information needed to reproduce the main experimental results of the paper.

   Guidelines:

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

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

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

https://doi.org/10.52202/079017-4254

Answer: [Yes] .

Justification: We have open-sourced the full code and trained model weights after thorough cleaning and the implementation of necessary safeguards.

Guidelines:

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

6. **Experimental Setting/Details**

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

   Answer: [Yes]

   Justification: See Sec. 5.1 and Tab. 5 for all the training and test details necessary to understand the results.

   Guidelines:

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

7. **Experiment Statistical Significance**

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

   Answer: [No]

   Justification: Following the literature [2, 41, 64, 14, 57], error bars are not reported to provide statistical significance, as the metrics FID, IS, Precision, and Recall inherently account for it by being computed while sampling 50K images.

   Guidelines:

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

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

8. **Experiments Compute Resources**

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

   Answer: [Yes]

   Justification: See Tab. 5 for all the information on the computer resources needed to reproduce the experiments.

   Guidelines:

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

9. **Code Of Ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conform the NeurIPS Code of Ethics in every respect.

   Guidelines:

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

10. **Broader Impacts**

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

    Answer: [Yes]

    Justification: See Sec. H for the discussion on the broader impacts.

    Guidelines:

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

https://doi.org/10.52202/079017-4254

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

11. **Safeguards**

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

    Answer: [Yes]

    Justification: See Sec. I for the discussion on the safeguards.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: See Sec. A for dataset licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

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

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

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

14. **Crowdsourcing and Research with Human Subjects**

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

    Answer: [NA]

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

    Guidelines:

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

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

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

    Answer: [NA]

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

    Guidelines:

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

 https://doi.org/10.52202/079017-4254