# Reviving ConvNeXt for Efficient Convolutional Diffusion Models

Taesung Kwon[1,2] Lorenzo Bianchi[2,3,4] Lennart Wittke[2] Felix Watine[2]
Fabio Carrara[3] Jong Chul Ye[1] Romann Weber[†] Vinicius Azevedo[†]

[1]KAIST [2]ETH Zürich [3]ISTI-CNR [4]University of Pisa

Figure 1. **Diffusion models with fully convolutional backbones achieve high-quality image generation with state-of-the-art efficiency.** We show selected samples from two of our class-conditional FCDM-XL models trained on ImageNet at 512×512 and 256×256 resolution.

## Abstract

*Recent diffusion models increasingly favor Transformer backbones, motivated by the remarkable scalability of fully attentional architectures. Yet the locality bias, parameter efficiency, and hardware friendliness—the attributes that established ConvNets as the efficient vision backbone—have seen limited exploration in modern generative modeling. Here we introduce the fully convolutional diffusion model (FCDM), a model having a backbone similar to ConvNeXt, but designed for conditional diffusion modeling. We find that using only 50% of the FLOPs of DiT-XL/2, FCDM-XL achieves competitive performance with 7× and 7.5× fewer training steps at 256×256 and 512×512 resolutions, respectively. Remarkably, FCDM-XL can be trained on a 4-GPU system, highlighting the exceptional training efficiency of our architecture. Our results demonstrate that modern convolutional designs provide a competitive and highly efficient alternative for scaling diffusion models, reviving ConvNeXt as a simple yet powerful building block for efficient generative modeling.*

## 1. Introduction

Over the past decade, convolutional neural networks (ConvNets) [15, 22, 23, 31, 35, 50, 54, 58, 60, 65] have driven most major advances in computer vision. Their success stems in part from the implicit "sliding window" mechanism, which embeds a strong locality inductive bias and

[†]Independent researcher.

Official implementation is available here.

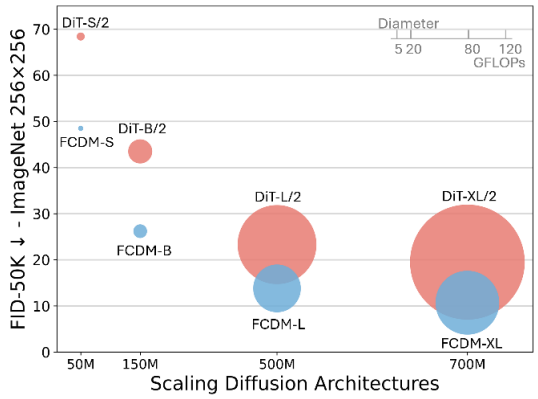



Figure 2. **Is *scalability* exclusive to transformers?** Our Fully Convolutional Diffusion Model (FCDM) exhibits clear scalability: it is more efficient and achieves better convergence than Diffusion Transformers (DiTs). Bubble size indicates the FLOPs of each diffusion model. Across all scales (ordered by parameter count).

| Model | Train steps | FLOPs (G) ↓ | Throughput (it/s) ↑ | FID ↓ |
|---|---|---|---|---|
| DiT-S/2 | 400K | 6 | 1234 | 68.4 |
| FCDM-S | 400K | **3** | **2687** | **48.5** |
| DiT-B/2 | 400K | 23 | 380.1 | 43.5 |
| FCDM-B | 400K | **12** | **1002** | **26.2** |
| DiT-L/2 | 400K | 81 | 114.6 | 23.3 |
| FCDM-L | 400K | **48** | **381.3** | **13.8** |
| DiT-XL/2 | 400K | 119 | 80.5 | 19.5 |
| FCDM-XL | 400K | **65** | **272.7** | **10.7** |
| DiT-XL/2 | 7M | 119 | 80.5 | 9.6 |
| FCDM-XL | 1M | **65** | **272.7** | **7.9** |

Table 1. FCDM consistently yields lower FLOPs, higher throughput, and converges faster to superior performance compared to DiT across all scales.

enables learning effective visual representations with far fewer parameters than fully connected layers. With the incorporation of patch embeddings in the Vision Transformer (ViT) [10, 39], Transformers [63] began to be actively explored in computer vision as well. In particular, the strong scalability of Transformers has allowed them to surpass ConvNets in many areas.

Generative models based on denoising [9, 11, 18, 25, 42, 46, 49, 56] have followed similar architectural trends, ranging from hybrid convolution–attention designs to fully transformer-based backbones. While foundational works [9, 18, 25, 49, 56] employed a hybrid architecture, DiT [46] introduced a fully transformer-based diffusion backbone, replacing convolutions with end-to-end transformer blocks. This shift has driven the success of recent state-of-the-art diffusion models [11, 33], offering improved scalability and generation quality. These developments reflect a prevailing belief that scaling transformer-based networks yields better generative performance. However, such rapid progress has come at the cost of a growing dependence on massive resources, owing to the inherent computational complexity of Transformers. The performance of modern diffusion models is increasingly tied to extensive GPU infrastructure, whose cost and energy demands are becoming major bottlenecks. This trend highlights the need for more efficient diffusion architectures that can achieve competitive performance under practical resource constraints.

In this work, we revisit the role of convolutions in diffusion modeling and introduce FCDM, a backbone similar to ConvNeXt for generative tasks. Building on the architectural strengths of ConvNeXt [40, 64], which has demonstrated strong competitiveness with Vision Transformers [10, 39] in terms of accuracy and scalability on ImageNet classification [51], we design a fully convolutional network for generative diffusion modeling. Specifically, our architecture differs from ConvNeXt by incorporating conditional injection and organizing it in an easily scalable U-shaped design. One of the key contributions of DiT [46] is its ease of scaling through a small set of intuitive hyperparameters (e.g., number of blocks $L$, hidden channel $C$, number of heads, and patch size $p$), which has made it highly practical and widely adopted in follow-up research. Our architecture further simplifies this design space, demonstrating an *Easy Scaling Law* with only two hyperparameters (number of blocks $L$ and hidden channel $C$). As a fully convolutional architecture, the proposed backbone incorporates the efficiency and scalability of modern ConvNets.

To isolate architectural effects and ensure a fair comparison of intrinsic network capabilities, we follow the standard DiT training and evaluation framework [46], in line with prior works [1, 62, 71]. To evaluate scalability, we scale our architecture to match the parameter counts of DiT models and find that our models require approximately 50% fewer FLOPs. Furthermore, we observe faster convergence in terms of training steps while achieving competitive performance relative to fully Transformer-based architectures. As shown in Figure 2 and Table 1, our **F**ully **C**onvolutional **D**iffusion **M**odel (FCDM) is more efficient and converges faster than DiT [46] across all model scales. Through extensive experiments, we find that our architecture remains highly efficient compared to modern diffusion architectures [1, 62, 71]. These findings re-emphasize the importance of convolutional operations as research increasingly favors Transformer-dominant architectures. They also offer a complementary perspective for efficiency-focused work: modern fully convolutional architectures provide an alternative path toward scalable, highly efficient generative modeling. We hope these observations and discussions challenge entrenched assumptions and encourage a reevaluation of the role of convolutions in modern computer vision.

## 2. Related Work

This section reviews the architectural evolution of diffusion models, highlighting the transition from convolution–attention hybrid designs to fully transformer-based backbones. From these trends, it is evident that fully convolutional diffusion architectures remain largely underexplored compared to their hybrid and transformer counterparts.

### 2.1. Hybrid Architectures

Early diffusion models predominantly adopted hybrid designs, combining convolutional layers for local features with self-attention [63] for long-range dependencies. DDPM [18], ScoreSDE [57], and DDIM [56] all used hybrid architectures augmented with attention at select resolutions. ADM [9] further showed that diffusion models could surpass GANs [4, 13, 24] in high-fidelity image generation, solidifying the architecture's viability. LDM [49] improved scalability by operating in a compressed latent space [29], enabling large-scale text-to-image models such as Stable Diffusion and Imagen [52]. Recent works including SDXL [47] and SnapGen [7] refined training strategies and resolution handling while retaining the convolution–attention hybrid backbone.

### 2.2. Transformer-based Diffusion Models

Transformers [10, 63] have emerged as strong alternatives, replacing all convolutions with patch-based attention blocks. DiT [46] demonstrated scalability with a ViT-inspired backbone, followed by U-ViT [3] and U-DiT [61], which introduced U-shaped variants. SiT [42] extended DiT to flow matching [37, 38], surpassing DiT across model scales. PixArt [5, 6], MM-DiT [11], and FLUX [33] scaled the architecture to production-grade text-to-image models by incorporating improved conditioning pipelines and refined transformer architectures. Recent approaches such as EQVAE [30], VAVAE [68], and REPA [69] further accelerated convergence and improved quality.

### 2.3. Convolutional Diffusion Models

Convolutional backbones for diffusion models have only recently reemerged. DiC [62] re-examined fully convolutional U-Nets using $3\times3$ convolutions with sparse skip connections, while DiCo [1] adapted $3\times3$ separable convolutions and proposed compact channel attention to mitigate channel redundancy. Both methods achieve competitive FID with superior throughput, demonstrating their computational efficiency compared to Diffusion Transformers. We demonstrate that our revived ConvNeXt backbone not only achieves superior generative performance but also delivers higher computational efficiency than prior convolutional diffusion models, thereby marking a rediscovery of ConvNeXt in the context of efficient generative modeling.

## 3. Fully Convolutional Diffusion Models

We propose a **F**ully **C**onvolutional **D**iffusion **M**odel (FCDM), reviving the ConvNeXt architecture [40, 64] and adapting it for conditional diffusion generation. Similar to how DiT [46] preserves design practices from Vision Transformers (ViTs) [10], FCDM retains the core principles of ConvNeXt. While ConvNeXt was originally developed for image classification, diffusion modeling imposes distinct requirements. We therefore reassemble ConvNeXt with conditional injection, carefully preserving its core design, and make it a suitable backbone for efficient generative diffusion modeling.

### 3.1. Designing Diffusion with ConvNeXt

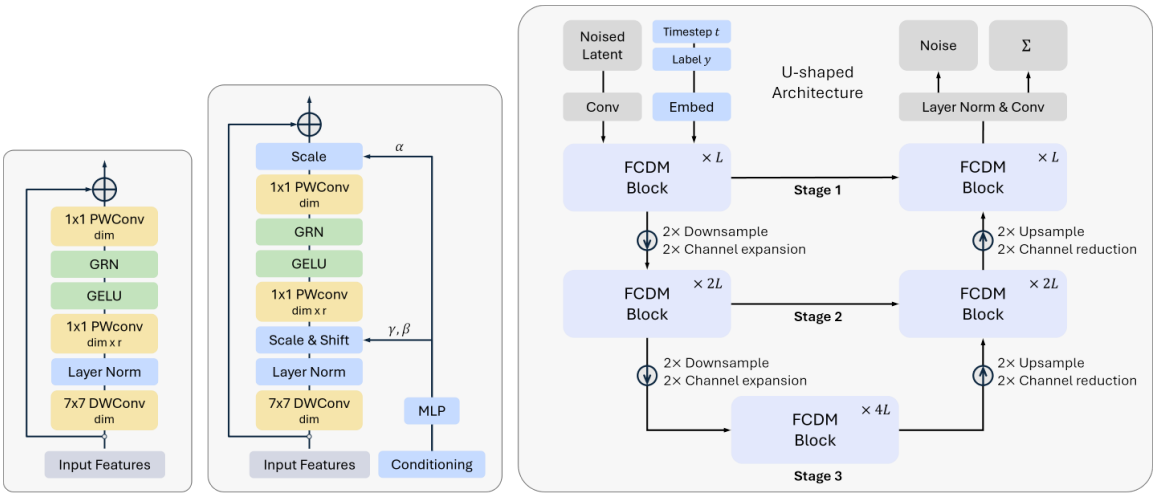
We redesign ConvNeXt into a generative backbone for diffusion models, introducing the Fully Convolutional Diffusion Model (FCDM). In this section, we highlight the advantages of our fully convolutional architecture and detail the key components that define the FCDM design space.

**ConvNeXt block.** As shown in Figure 3 (a), the original ConvNeXt [40, 64] block begins with a $7\times7$ depthwise convolution, followed by layer normalization [2]. Two subsequent $1\times1$ pointwise convolutions handle channel expansion and reduction with a ratio of $r$, while the Global Response Normalization (GRN) [64] in between mitigates channel redundancy.

**Conditional injection.** The original ConvNeXt blocks lack conditioning mechanisms, as they were originally designed for image classification. To enable class and time conditioning, we replace LayerNorm with Adaptive LayerNorm (AdaLN), as shown in Figure 3 (b). A lightweight MLP maps the conditioning vector (derived from class and time embeddings) to $(\gamma, \beta, \alpha)$ parameters that modulate normalized features. Following DiT [46], we zero-initialize the final modulation scale $\alpha$ to stabilize optimization and allow deeper training.

**Easily scalable U-shaped architecture.** Most convolutional networks adopt a U-shaped design with skip connections, which facilitates the integration of global and local features. This structure makes it easier to capture the overall context while preserving the high-resolution details from the early encoder layers. Following this principle, we organize ConvNeXt blocks within a U-Net hierarchy, with skip connections bridging the encoder and decoder stages.

To simplify scalability, we avoid the complex, resolution-specific design choices often used in U-shaped networks. Instead, our architecture is parameterized by only two hyperparameters: the number of blocks $L$ and hidden channels $C$. At each $2\times$ downsampling stage, both




(a) ConvNeXt block (b) FCDM block (ConvNext block with adaLN) (c) Overall architecture of FCDM

Figure 3. **The Fully Convolutional Diffusion Model (FCDM) architecture.** (a) Details of the ConvNeXt block. (b) Our FCDM block, which incorporates conditioning via adaptive layer normalization. (c) We train conditional latent FCDMs. The input latent is processed by multiple FCDM blocks arranged in an easily scalable U-shaped architecture.

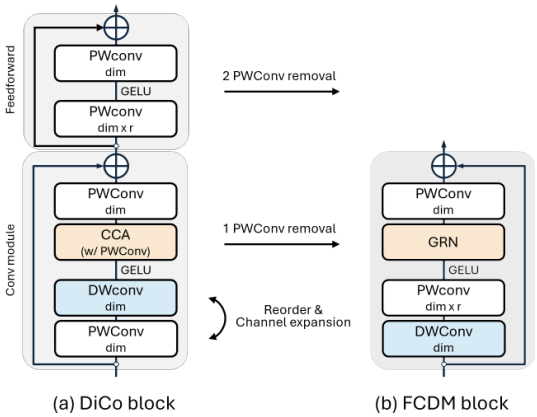
| Model (Params) | Blocks $L$ | Channel $C$ | FLOPs (G) | $\frac{\text{FLOPs (FCDM)}}{\text{FLOPs (DiT)}}$ | $\frac{\text{FLOPs (FCDM)}}{\text{FLOPs (DiCo)}}$ |
|---|---|---|---|---|---|
| FCDM-S (32.7M) | 2 | 128 | 3.1 | 50.8% | 72.9% |
| FCDM-B (127.7M) | 2 | 256 | 12.2 | 53.0% | 72.3% |
| FCDM-L (504.5M) | 2 | 512 | 48.3 | 59.9% | 80.2% |
| FCDM-XL (698.8M) | 3 | 512 | 64.6 | 54.5% | 74.0% |

Table 2. Parameter counts are aligned with the Diffusion Transformer (DiT) configurations for the Small (S), Base (B), Large (L), and XLarge (XL) variants. FLOPs are computed at a 256×256 resolution. Overall, FCDM uses roughly 50% of the FLOPs of DiT and about 75% of the FLOPs of DiCo.

$C$ and $L$ are doubled. This *generalized U-shaped* design (Figure 3 (c)) allows straightforward scaling while retaining the inductive biases of convolutions. By controlling $C$ and $L$, the proposed architecture can be scaled up or down flexibly. Extensive architectural ablations confirm that this simplified design does not compromise performance. For detailed results, please refer to the Supplementary Materials.

### 3.2. Revisiting DiCo.

DiCo [1] represents the current state-of-the-art convolutional diffusion model. It employs 3×3 separable convolutions and introduces a compact channel attention mechanism that encourages more diverse channel activations. Interestingly, we observe a close connection between DiCo and our architecture, where the latter offers a more efficient alternative to DiCo's design choices. As shown in Table 2, FCDM achieves approximately 75% FLOPs efficiency compared to DiCo at a similar parameter scale. In the following, we detail how our model serves as a more efficient solution for convolutional diffusion models.




(a) DiCo block (b) FCDM block

Figure 4. **Simple illustration of DiCo and FCDM block.** Both architectures share a similar high-level structure, but FCDM adopts an inverted bottleneck that expands channels for richer representations while keeping the computational cost of depthwise convolution unchanged. DiCo employs CCA with an additional 1×1 convolution, whereas FCDM uses GRN, requiring no extra pointwise convolutions. FCDM also does not include DiCo's feedforward module, resulting in a simpler and more efficient block.

**Richer channel representations.** As illustrated in Figure 4, the convolution modules of DiCo and FCDM follow a similar high-level structure. However, a key structural difference lies in how each model handles channel dimensionality. While DiCo preserves the channel dimension throughout the convolution module, our design adopts the inverted bottleneck structure of ConvNeXt, introducing an early channel expansion that allows for richer channel computation within the block. Crucially, our design reorders the pointwise and depthwise convolutions so that the channel expansion is applied *after* the depthwise convolu-

tion. This reordering keeps the computational cost of depthwise convolution unchanged regardless of channel expansion, while still leveraging the expressive capacity of the expanded channels.

**GRN promotes diverse channel activations.** DiCo introduces the compact channel attention (CCA) mechanism to promote diverse channel activations. We observe that the global response normalization (GRN) layer in ConvNeXt V2 [64] achieves a similar effect while requiring far fewer learnable parameters. CCA relies on an additional $1\times1$ pointwise convolution to learn channel-wise activations, whereas GRN is composed primarily of parameter-free operations such as L2 normalization and response normalization [31]. From this perspective, both mechanisms aim to enhance channel activation diversity, but GRN provides a substantially more efficient alternative.

Lastly, as shown in Figure 4, DiCo includes an additional feed-forward module composed of two $1\times1$ pointwise convolutions, where the channel expansion is performed. In contrast, our design already applies channel expansion within the earlier convolution module through the inverted bottleneck structure.

Together, these findings indicate that our design provides a more efficient architectural alternative to the current state-of-the-art convolutional diffusion model, DiCo. In the following section, we show that this improved efficiency is also reflected in the empirical performance.

## 4. Experimental Setup

**Model size.** We denote our models by their configurations, parameterized by hidden channels $C$ and number of blocks $L$, which are both doubled at each $2\times$ downsampling stage. To enable fair comparisons, we align the parameter counts of our FCDM scales with those of DiT [46] (e.g., DiT-B: 130M vs. FCDM-B: 127.7M). We evaluate four model scales, as listed in Table 2: FCDM-S, FCDM-B, FCDM-L, and FCDM-XL. These cover a broad range of parameter counts, from 32.7M to 698.8M, allowing us to systematically study scaling behavior and compare with DiT across different scales.

**Training.** We train class-conditional latent FCDMs at $256\times256$ and $512\times512$ resolutions on the ImageNet dataset [51], a standard yet highly competitive benchmark for generative modeling. Training follows common practices of DiT [46]: we use AdamW [28, 41] with a fixed learning rate of $1 \times 10^{-4}$, no weight decay, and batch size 256. The only augmentation applied is horizontal flipping. We use an exponential moving average (EMA) of model weights with a decay factor of 0.9999, and report all results using the EMA weights. We retain diffusion hyperparameters from ADM [9]: $t_{\text{max}} = 1000$ steps with a linear variance schedule ($1 \times 10^{-4}$ to $2 \times 10^{-4}$), ADM's covariance parameterization $\Sigma_\theta$, and their timestep/label embedding method. Please refer to Supplementary Section A for an overview of denoising diffusion probabilistic models and to Supplementary Section B for additional training details and hyperparameters.

**Datasets and Metrics.** We conduct experiments on ImageNet-1K at $256\times256$ and $512\times512$ resolutions for class-conditional image generation. Our primary metric is Fréchet Inception Distance (FID) [16], following the standard evaluation protocol. We sample 50K images with 250 DDPM sampling steps, and compute the metrics using OpenAI's official TensorFlow evaluation toolkit [9]. As secondary metrics, we also report Inception Score (IS) [53] and Precision/Recall [32]. Please refer to Supplementary Section C for more details.

**Compute.** All models are implemented in PyTorch [45]. The largest model, FCDM-XL, trains at approximately 0.9 iterations per second (with gradient checkpointing) at $256\times256$ resolution on four RTX 4090 24GB GPUs with a global batch size of 256, demonstrating the efficiency of our architecture. We also verify that the batch size of 256 fits on a single A100 40GB GPU, further highlighting the memory efficiency of our design.

## 5. Experiments

### 5.1. Scaling model size

We train four FCDM models (S, B, L, XL), all using the same training configuration. Figure 2 summarizes the FLOPs and FID at 400K training iterations. In all cases, scaling up model size improves performance. Figure 5 further shows that FCDM consistently outperforms DiT [46] across all scales. Increasing model scale, both in width and depth, consistently leads to significant FID improvements.

Table 3 provides a broader comparison with DiT and modern diffusion architectures that follow a similar experimental setup. Although our scales are aligned with DiT in terms of parameter counts, FCDM requires about 50% fewer FLOPs than DiT and 25% fewer FLOPs than DiCo [1]. Notably, with 64.6 GFLOPs, FCDM-XL is computationally closer to Large (L) scale models in prior works, yet it outperforms even XL-scale models in terms of FID. In particular, FCDM-XL achieves superior FID while requiring $7\times$ fewer training steps compared to DiT-XL/2. Furthermore, this efficiency yields strong throughput performance. While DiC [62] has the best throughput at S and B scales due to its use of standard convolutions, which are simpler and highly optimized for modern hardware, FCDM outperforms it at L and XL scales, achieving the highest

| Model | Architecture Type | FLOPs (G) ↓ | Throughput (it/s) ↑ | FID ↓ | IS ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|---|---|---|---|
| DiT-S/2 (400K) *(ICCV 2023)* | Transformer | 6.1 | 1234.0 | 68.40 | - | - | - |
| DiC-S (400K) *(CVPR 2025)* | Conv | 5.9 | **3148.8** | 58.68 | 25.82 | - | - |
| DiG-S/2 (400K) *(CVPR 2025)* | Hybrid | 4.3 | 961.2 | 62.06 | 22.81 | 0.39 | 0.56 |
| DiCo-S (400K) *(NeurIPS 2025)* | Conv | 4.3 | 1695.7 | 49.97 | 31.38 | **0.48** | **0.58** |
| **FCDM-S** (400K) | Conv | **3.1** | 2687.2 | **48.52** | **31.64** | **0.48** | **0.58** |
| DiT-B/2 (400K) | Transformer | 23.0 | 380.1 | 43.47 | - | - | - |
| DiC-B (400K) | Conv | 23.5 | **1024.2** | 32.33 | 48.72 | - | - |
| DiG-B/2 (400K) | Hybrid | 17.1 | 345.9 | 39.50 | 37.21 | 0.51 | **0.63** |
| DiCo-B (400K) | Conv | 16.9 | 823.0 | 27.20 | 56.52 | **0.60** | 0.61 |
| **FCDM-B** (400K) | Conv | **12.2** | 1001.6 | **26.21** | **58.04** | 0.59 | 0.61 |
| DiT-L/2 (400K) | Transformer | 80.7 | 114.6 | 23.33 | - | - | - |
| DiG-L/2 (400K) | Hybrid | 61.7 | 109.0 | 22.90 | 59.87 | 0.60 | **0.64** |
| DiCo-L (400K) | Conv | 60.2 | 288.3 | **13.66** | **91.37** | **0.69** | 0.61 |
| **FCDM-L** (400K) | Conv | **48.3** | **381.3** | 13.83 | 93.31 | 0.66 | 0.62 |
| DiT-XL/2 (400K) | Transformer | 118.6 | 80.5 | 19.47 | - | - | - |
| DiC-XL (400K) | Conv | 116.1 | 263.1 | 13.11 | 100.2 | - | - |
| DiG-XL/2 (400K) | Hybrid | 89.4 | 71.7 | 18.53 | 68.53 | 0.63 | **0.64** |
| DiCo-XL (400K) | Conv | 87.3 | 174.2 | 11.67 | 100.4 | **0.71** | 0.61 |
| DiC-H (400K) | Conv | 204.4 | 144.5 | 11.36 | 106.5 | - | - |
| **FCDM-XL** (400K) | Conv | **64.6** | **272.7** | **10.72** | **108.0** | 0.69 | 0.63 |
| DiT-XL/2 (7M) | Transformer | 118.6 | 80.5 | 9.62 | - | - | - |
| DiC-H (800K) | Conv | 204.4 | 144.5 | 8.96 | 124.33 | - | - |
| DiG-XL/2 (1.2M) | Hybrid | 89.4 | 71.7 | 8.60 | 130.03 | 0.68 | **0.68** |
| **FCDM-XL** (1M) | Conv | **64.6** | **272.7** | **7.91** | **135.55** | **0.71** | 0.64 |

Table 3. **Scalability comparisons on ImageNet 256×256.** For each model scale, we report FID, IS, Precision, and Recall (50K samples without guidance), and efficiency metrics (training iterations, FLOPs, throughput). FCDM-XL achieves superior convergence while using 50% fewer FLOPs than DiT-XL/2. The best results are highlighted in **bold**. Evaluated methods operate in the latent space.

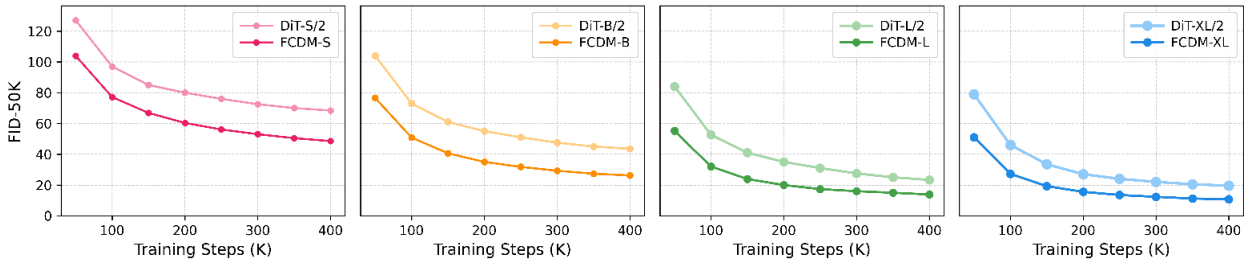



Figure 5. **FCDM improves FID across all model scales.** FID-50K over training iterations for both DiT and FCDM. Across all model scales, FCDM converges much faster.

throughput. For completeness, detailed descriptions of each baseline method are provided in Supplementary Section D, and a detailed scaling analysis is provided in Supplementary Section E.

## 5.2. Benchmarking Performance and Efficiency

**256×256 ImageNet.** Building on the scaling analysis, we train FCDM-XL for 2M iterations (400 epochs) and evaluate it with classifier-free guidance [17]. Figure 1 presents generated samples and Table 4 further compares against prior class-conditional image generation models. Notably, FCDM-XL improves upon baselines despite requiring fewer training epochs. In particular, it achieves an FID of 2.03 and an IS of 285.7, while reaching state-of-the-art efficiency in FLOPs and throughput, demonstrating a strong trade-off between performance and efficiency. As shown in Figure 6, our model achieves state-of-the-art throughput while drastically reducing training cost. Notably, compared with DiCo, our method maintains competitive performance while operating with 2.5× fewer total training FLOPs and 1.5× faster inference throughput. These results highlight, for the first time, the effectiveness of ConvNeXt architectures in generative diffusion modeling, which had previously been shown only in classification, and demonstrate their broader adaptability and potential. It is important to note that, although FCDM-XL demonstrates superior performance over comparable baselines within DiT-style experimental settings, it does not yet surpass the latest state-of-the-art results achieved by models such as EDM-2 [26] or Simpler Diffusion [21]. Nevertheless, as an architecture with a favorable performance–efficiency trade-off, our approach holds the poten-

| Model | Training epochs | FLOPs (G) ↓ | Throughput (it/s) ↑ | FID ↓ | IS ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|---|---|---|---|
| *GAN* | | | | | | | |
| BigGAN-deep | - | - | - | 6.95 | 171.4 | **0.87** | 0.28 |
| StyleGAN-XL | - | - | - | 2.30 | 265.12 | 0.78 | 0.53 |
| *Pixel diffusion* | | | | | | | |
| ADM-U | 400 | 742.0 | - | 3.94 | 215.8 | 0.83 | 0.53 |
| VDM++ | 560 | - | - | 2.12 | 267.7 | - | - |
| Simple Diffusion | 800 | - | - | 2.77 | 211.8 | - | - |
| CDM | 2160 | - | - | 4.88 | 158.7 | - | - |
| *Latent diffusion* | | | | | | | |
| LDM-4 | 200 | 104.0 | - | 3.60 | 247.7 | **0.87** | 0.48 |
| U-ViT-H/2 | 240 | 133.3 | 73.5 | 2.29 | 263.9 | 0.82 | 0.57 |
| MaskDiT | 1600 | - | - | 2.28 | 276.6 | 0.80 | **0.61** |
| SD-DiT | 480 | - | - | 3.23 | - | - | - |
| DiT-XL/2 | 1400 | 118.6 | 80.5 | 2.27 | 278.2 | 0.83 | 0.57 |
| SiT-XL/2 | 1400 | 118.6 | 80.5 | 2.06 | 277.5 | 0.83 | 0.59 |
| DiG-XL/2 | 240 | 89.4 | 71.7 | 2.07 | 279.0 | 0.82 | 0.60 |
| DiCo-XL | 750 | 87.3 | 174.2 | 2.05 | 282.2 | 0.83 | 0.59 |
| DiC-H | 400 | 204.4 | 144.5 | 2.25 | - | - | - |
| **FCDM-XL** | 400 | **64.6** | **272.7** | **2.03** | **285.7** | 0.81 | 0.59 |

Table 4. **Benchmarking class-conditional image generation on ImageNet 256×256.** We compare representative models in terms of FID, IS, Precision, Recall (with guidance), and efficiency metrics (training epochs, FLOPs, throughput). FCDM-XL achieves competitive performance with superior efficiency. The best results are highlighted in **bold**.

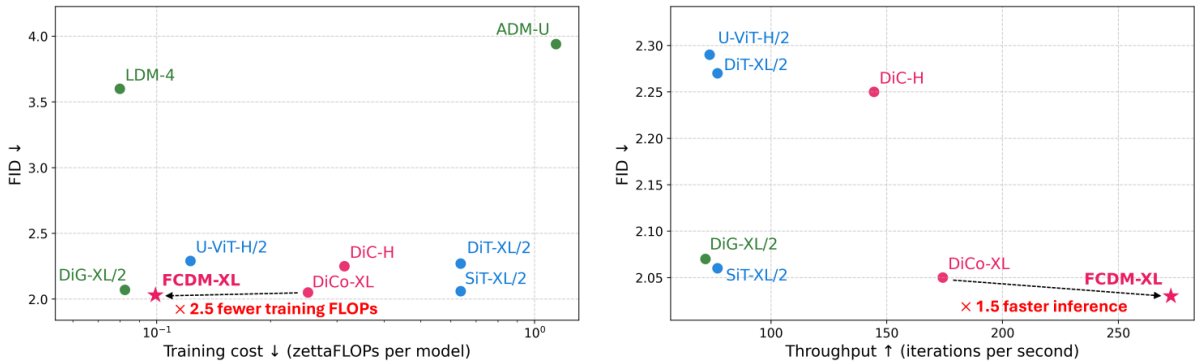



Figure 6. **Benchmarking class-conditional image generation performance and efficiency on ImageNet 256×256.** *Left*: FID versus total training cost. *Right*: FID versus throughput. One zettaFLOP corresponds to $10^{21}$ FLOPs ($10^{12}$ GFLOPs). A training iteration is assumed to cost about 3× one evaluation (forward + backward to inputs + backward to weights). Red denotes fully convolutional, Green denotes hybrid, and Blue denotes fully transformer-based models.

tial to deliver improved results with further scaling and more advanced training frameworks.

**512×512 ImageNet.** We also train FCDM-XL at 512×512 resolution for 1M iterations using the same hyperparameters as the 256×256 model. Table 5 reports FLOPs and FID at both 400K and 1M iterations. At this resolution, FCDM-XL again achieves the best FLOPs and throughput. With this efficiency advantage, FCDM-XL delivers better FID convergence at 400K iterations, and this trend continues at 1M. Remarkably, FCDM-XL outperforms DiT with 7.5× fewer training steps. Moreover, under comparable training iterations, FCDM-XL outperforms DiCo across all evaluation and efficiency metrics. These results show that FCDM maintains strong efficiency while delivering competitive performance, highlighting the effectiveness of the proposed architecture. Interestingly, when the resolution doubles, the throughput of DiT drops by approximately 4×, whereas FCDM degrades by only 2×. This contrast highlights the fundamental computational differences between fully transformer-based and fully convolutional architectures and further illustrates the scalability of FCDM at higher resolutions. Finally, we provide a frequency-domain analysis in Supplementary Section F to better understand the model behavior compared to Diffusion Transformers, text-to-image results in Supplementary Section H, and additional quantitative and qualitative results in Supplementary Sections I and J.

### 5.3. Ablation Study

We conduct ablations with the Large (L) model on ImageNet 256×256 to analyze the effects of key architectural

| Model | Training iterations | FLOPs (G) ↓ | Throughput (it/s) ↑ | FID ↓ | IS ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|---|---|---|---|
| DiT-XL/2 | 400K | 524.7 | 18.6 | 20.94 | 66.3 | 0.74 | 0.58 |
| DiG-XL/2 | 400K | - | - | 17.36 | 69.4 | 0.75 | **0.63** |
| DiC-XL | 400K | 464.3 | 124.2 | 15.32 | 93.6 | - | - |
| DiC-H | 400K | 817.2 | 68.6 | 12.89 | 101.8 | - | - |
| **FCDM-XL** | 400K | **257.7** | **129.6** | **10.23** | **108.7** | **0.79** | 0.60 |
| DiT-XL/2 | 1.3M | 524.7 | 18.6 | 13.78 | - | - | - |
| DiT-XL/2 | 3M | 524.7 | 18.6 | 12.03 | 105.25 | 0.75 | **0.64** |
| DiCo-XL | 1.3M | 349.8 | 82.0 | 8.10 | 132.9 | 0.78 | 0.62 |
| **FCDM-XL** | **1M** | **257.7** | **129.6** | **7.46** | **133.6** | **0.79** | 0.61 |

Table 5. **Benchmarking class-conditional image generation on ImageNet 512×512.** We report FID, IS, Precision, Recall (without guidance), and efficiency metrics for representative models. Even at this resolution, FCDM surpasses models trained for 3M iterations with only 1M iterations and achieves the best efficiency in FLOPs and throughput. The best results are highlighted in **bold**.

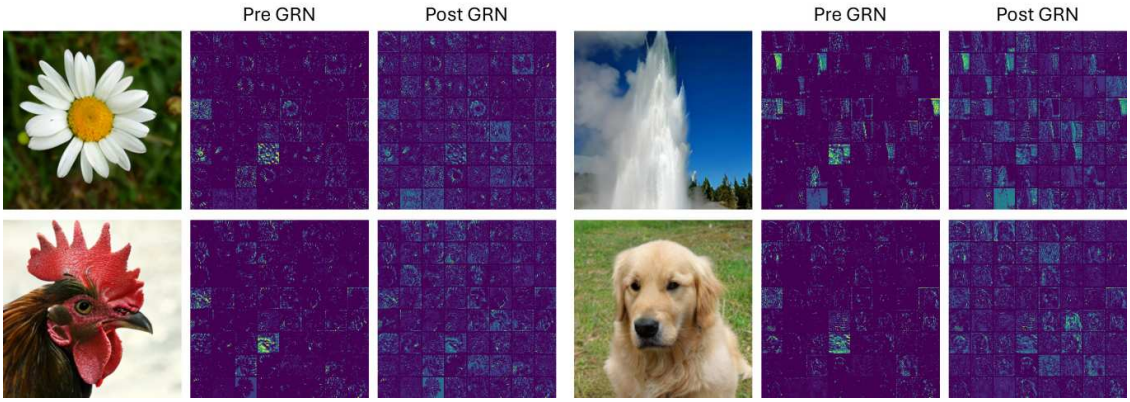



Figure 7. **Feature activation visualization.** We visualize features before and after the GRN layer during sampling for each image on the left. The first 64 channels of the last block in the first stage are shown as 8×8 grids. GRN clearly reduces channel redundancy.

| Model | FLOPs (G) ↓ | FID ↓ | IS ↑ |
|---|---|---|---|
| **FCDM-L** (Default: 7×7 DWConv) | 48.3 | **19.97** | **69.19** |
| → 5×5 DWConv | 48.2 | 20.48 | 66.69 |
| → 3×3 DWConv | 48.1 | 21.28 | 64.11 |
| **FCDM-L** (Default: GRN) | 48.3 | **19.97** | **69.19** |
| → CCA* [1] | 48.3 | 23.85 | 61.60 |
| **FCDM-L** (Default: w/o Feedforward) | 48.3 | **19.97** | **69.19** |
| → w/ Feedforward* | 48.2 | 28.52 | 47.16 |
| **FCDM-L** (Default: w/ Inv. Bottleneck) | 48.3 | **19.97** | **69.19** |
| → w/o Inv. Bottleneck* | 48.3 | 28.76 | 52.20 |
| **FCDM-L** (Default: FCDM block) | 48.3 | **19.97** | **69.19** |
| → ResNet block* | 48.4 | 31.14 | 49.10 |

Table 6. **Ablation study on FCDM design choices.** We analyze the effects of kernel size, GRN, DiCo [1] design choices, and the FCDM block. Training iterations are fixed to 200K. $*$ indicates that $C$ is adjusted to match FLOPs to ensure a fair comparison.

components, as summarized in Table 6. First, reducing the kernel size consistently degrades performance, indicating that large kernels successfully expand the effective receptive field to capture broader context. Second, incorporating DiCo [1] design choices, such as the application of compact channel attention (CCA) or a feedforward module, consistently degrades performance, demonstrating the effectiveness of our design in diffusion modeling. Finally, removing the inverted bottleneck or replacing FCDM blocks with standard ResNet blocks [15] results in a severe degradation, reaffirming the advantage of our design. Furthermore, as shown in Figure 7, GRN clearly reduces channel redundancy, confirming its contribution to enhancing channel diversity. These results demonstrate that GRN provides an effect similar to CCA, but with far fewer parameters. More ablation results are provided in Supplementary Section G.

## 6. Conclusion

As interest in efficient diffusion architectures continues to grow, we revive the role of ConvNeXt in generative modeling and introduce FCDM, our redesigned ConvNeXt backbone tailored for conditional diffusion models. By redesigning the ConvNeXt architecture to incorporate conditional injection and organizing it into an easily scalable U-shaped design, we enable FCDM to achieve competitive generative performance with state-of-the-art computational efficiency compared to modern diffusion architectures. These results demonstrate that modern convolutional architectures provide an alternative path toward scalable, highly efficient generative modeling and challenge the prevailing belief that larger Transformers are the sole path to progress in diffusion models.

## Acknowledgements

We thank Jakob Buhmann, Farnood Salehi, and Jingwei Tang for helpful discussions. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2026-25468886). Taesung Kwon and Jong Chul Ye are further supported by the National Research Foundation of Korea (NRF) (RS-2024-00336454 and RS-2023-00262527). We also acknowledge the support of ETH Zürich in providing access to the Euler cluster for this research.

## References

[1] Yuang Ai, Qihang Fan, Xuefeng Hu, Zhenheng Yang, Ran He, and Huaibo Huang. Dico: Revitalizing convnets for scalable and efficient diffusion modeling. *Advances in neural information processing systems*, 38, 2025. 2, 3, 4, 5, 8, 7

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[3] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22669–22679, 2023. 3

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. 3

[5] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-$\sigma$: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. In *European Conference on Computer Vision*, pages 74–91. Springer, 2024. 3

[6] Junsong Chen, Jincheng YU, Chongjian GE, Lewei Yao, Enze Xie, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-$\alpha$: Fast training of diffusion transformer for photorealistic text-to-image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[7] Jierun Chen, Dongting Hu, Xijie Huang, Huseyin Coskun, Arpit Sahni, Aarush Gupta, Anujraaj Goyal, Dishani Lahiri, Rajesh Singh, Yerlan Idelbayev, et al. Snapgen: Taming high-resolution text-to-image models for mobile devices with efficient architectures and training. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 7997–8008, 2025. 3

[8] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 2, 3, 5

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 2, 3

[11] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024. 2, 3, 10

[12] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Masked diffusion transformer is a strong image synthesizer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 23164–23173, 2023. 11

[13] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 3

[14] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6185–6194, 2023. 8

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 8, 7

[16] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 5, 3

[17] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 6

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2, 3, 1

[19] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022. 3

[20] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning*, pages 13213–13232. PMLR, 2023. 3

[21] Emiel Hoogeboom, Thomas Mensink, Jonathan Heek, Kay Lamerigts, Ruiqi Gao, and Tim Salimans. Simpler diffusion (sid2): 1.5 fid on imagenet512 with pixel-space diffusion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2025. 6, 8

[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. 1

[23] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 1

[24] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 3

[25] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 2

[26] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24174–24184, 2024. 6

[27] Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. *Advances in Neural Information Processing Systems*, 36: 65484–65516, 2023. 3

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 5

[29] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014. 3, 2

[30] Theodoros Kouzelis, Ioannis Kakogeorgiou, Spyros Gidaris, and Nikos Komodakis. EQ-VAE: Equivariance regularized latent space for improved generative image modeling. In *Forty-second International Conference on Machine Learning*, 2025. 3, 2, 7

[31] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 1, 5

[32] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems*, 32, 2019. 5, 3

[33] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024. 2, 3

[34] Andrew Lavin and Scott Gray. Fast algorithms for convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4013–4021, 2016. 7

[35] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 1

[36] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 10

[37] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. 3, 4

[38] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. 3

[39] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 2

[40] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022. 2, 3, 7

[41] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 5

[42] Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pages 23–40. Springer, 2024. 2, 3, 4

[43] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter Battaglia. Generating images with sparse representations. In *International Conference on Machine Learning*, pages 7958–7968. PMLR, 2021. 3

[44] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 1

[45] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 5

[46] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 2, 3, 5, 1, 8

[47] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024. 3, 10

[48] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 10

[49] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image

synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2, 3, 7

[50] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1

[51] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 2, 5

[52] 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. *Advances in neural information processing systems*, 35:36479–36494, 2022. 3

[53] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 5, 3

[54] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 1, 7

[55] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015. 1

[56] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 2, 3

[57] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. 3

[58] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 1

[59] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 3

[60] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 1

[61] Yuchuan Tian, Zhijun Tu, Hanting Chen, Jie Hu, Chao Xu, and Yunhe Wang. U-dits: Downsample tokens in u-shaped diffusion transformers. *Advances in Neural Information Processing Systems*, 37:51994–52013, 2024. 3

[62] Yuchuan Tian, Jing Han, Chengcheng Wang, Yuchen Liang, Chao Xu, and Hanting Chen. Dic: Rethinking conv3x3 designs in diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2469–2478, 2025. 2, 3, 5

[63] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2, 3

[64] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16133–16142, 2023. 2, 3, 5, 7

[65] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 1

[66] Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *Forty-first International Conference on Machine Learning*, 2024. 3

[67] Jingfeng Yao, Cheng Wang, Wenyu Liu, and Xinggang Wang. Fasterdit: Towards faster diffusion transformers training without architecture modification. *Advances in Neural Information Processing Systems*, 37:56166–56189, 2024. 11

[68] Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15703–15712, 2025. 3

[69] Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *The Thirteenth International Conference on Learning Representations*, 2025. 3

[70] Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar. Fast training of diffusion models with masked transformers. *Transactions on Machine Learning Research*, 2024. 3

[71] Lianghui Zhu, Zilong Huang, Bencheng Liao, Jun Hao Liew, Hanshu Yan, Jiashi Feng, and Xinggang Wang. Dig: Scalable and efficient diffusion models with gated linear attention. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 7664–7674, 2025. 2, 3

[72] Rui Zhu, Yingwei Pan, Yehao Li, Ting Yao, Zhenglong Sun, Tao Mei, and Chang Wen Chen. Sd-dit: Unleashing the power of self-supervised discrimination in diffusion transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8435–8445, 2024. 3

# Reviving ConvNeXt for Efficient Convolutional Diffusion Models

## Supplementary Material

We provide the following supplementary sections:

- Section A: Overview of denoising diffusion probabilistic models (DDPMs).
- Section B: Hyperparameter and implementation details.
- Section C: Descriptions of evaluation metrics.
- Section D: Descriptions of baseline models.
- Section E: Additional results and analyses on model scalability.
- Section F: Frequency-based analysis of model behavior.
- Section G: Additional analyses of architectural variants.
- Section H: Experimental results on text-to-image generation.
- Section I: Additional quantitative results and analyses.
- Section J: Additional qualitative results and visual samples.

## A. Overview of denoising diffusion probabilistic models

Diffusion models [18, 55] aim to model a target distribution $p(x)$ by learning a gradual denoising process from Gaussian noise $\mathcal{N}(0, I)$ to $p(x)$. Specifically, the model learns a *reverse* process $p_\theta(x_{t-1}|x_t)$ of a predefined *forward* diffusion process $q(x_t|x_{t-1})$, which progressively adds Gaussian noise over $T$ timesteps.

For an initial sample $x_0 \sim p(x)$, the *forward* process is defined as:

$$q(x_t|x_{t-1}) = \mathcal{N}\Big(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I\Big)\,, \tag{1}$$

where $\beta_t \in (0, 1)$ is a variance schedule. A closed-form expression of $q(x_t|x_0)$ can also be derived as:

$$q(x_t|x_0) = \mathcal{N}\big(x_t; \sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t) I\big)\,, \quad \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s). \tag{2}$$

The denoising diffusion probabilistic model (DDPM) [18] parameterizes the *reverse* transition as:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\Big(x_{t-1}; \frac{1}{\sqrt{\alpha_t}}\Big(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\,\epsilon_\theta(x_t, t)\Big), \sigma_t^2 I\Big), \tag{3}$$

where the noise predictor $\epsilon_\theta(x_t, t)$ is trained using the simple denoising autoencoder objective:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{x_t, x_0, \epsilon, t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\right]. \tag{4}$$

Following DDPM, one can set $\sigma_t^2 = \beta_t$ for simplicity. Meanwhile, improved DDPM (iDDPM) [44] shows that performance can be improved by jointly learning the variance $\Sigma_\theta(x_t, t)$, which is parameterized as an interpolation between $\beta_t$ and $\tilde{\beta}_t$ in the log domain:

$$\log \Sigma_\theta(x_t, t) = v \log \beta_t + (1-v)\, \log \tilde{\beta}_t, \tag{5}$$

where $\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$, and $v$ denotes the interpolation weight predicted by the model in a dimension-wise manner. In this work, we adopt the iDDPM framework for both training and sampling, following the same design choice as DiT [46].

## B. Hyperparameters and implementation details

We design Fully Convolutional Diffusion Models (FCDMs) at multiple scales, aligned by parameter counts with DiT. Thanks to their easy scalability, we adjust only two hyperparameters, $L$ and $C$, to obtain these variants. Notably, when compared to DiT in terms of FLOPs, our FCDMs require only 50.8% to 59.9% of those consumed by DiT, demonstrating the state-of-the-art computational efficiency of our design. Our implementation is based on the original DiT codebase [46]. Detailed configurations of these hyperparameters, along with additional implementation details, are provided in Table 7. For the latent space, we adopt an off-the-shelf pre-trained variational autoencoder (VAE) [29, 30, 49] with a downsampling factor of 8. Accordingly, an input RGB image of shape $256\times256\times3$ is encoded to a latent tensor of $32\times32\times4$. All diffusion training operates in this latent space, and latents are decoded back to pixels by the VAE decoder.

| Resolution | FCDM-S 256×256 | FCDM-B 256×256 | FCDM-L 256×256 | FCDM-XL 256×256 | FCDM-XL 512×512 |
|---|---|---|---|---|---|
| **Architecture** | | | | | |
| Input dim. | 32×32×4 | 32×32×4 | 32×32×4 | 32×32×4 | 64×64×4 |
| Num. blocks ($L$) | 2 | 2 | 2 | 3 | 3 |
| Hidden channels ($C$) | 128 | 256 | 512 | 512 | 512 |
| 1×1 conv. expansion ratio ($r$) | 3 | 3 | 3 | 3 | 3 |
| **Training** | | | | | |
| Training iteration | 400K | 400K | 400K | 2M | 1M |
| Global batch size | 256 | 256 | 256 | 256 | 256 |
| Optimizer | AdamW | AdamW | AdamW | AdamW | AdamW |
| Learning rate | $1\times10^{-4}$ | $1\times10^{-4}$ | $1\times10^{-4}$ | $1\times10^{-4}$ | $1\times10^{-4}$ |
| Learning rate schedule | constant | constant | constant | constant | constant |
| $(\beta_1, \beta_2)$ | (0.9, 0.999) | (0.9, 0.999) | (0.9, 0.999) | (0.9, 0.999) | (0.9, 0.999) |
| Weight decay | 0 | 0 | 0 | 0 | 0 |
| Numerical precision | fp32 | fp32 | fp32 | fp32 | fp32 |
| Data augmentation | random flip | random flip | random flip | random flip | random flip |
| **Sampling** | | | | | |
| Sampler | iDDPM | iDDPM | iDDPM | iDDPM | iDDPM |
| Sampling steps | 250 | 250 | 250 | 250 | 250 |

Table 7. **Hyperparameter setup of FCDM model scales.** For all scales of FCDM, we adopt the same experimental settings as DiT.

**Computing resources.** Thanks to the superior efficiency of our fully convolutional architecture, we are able to train ImageNet models at 256×256 resolution using *consumer*-grade GPUs such as the NVIDIA RTX 4090 (24GB). For 256×256, we train FCDM-XL with a batch size of 256 on 4× RTX 4090 GPUs, achieving a training throughput of approximately 0.9 steps/s with gradient checkpointing enabled. We also confirm that the same batch size 256 fits on a single NVIDIA A100 40GB GPU, further demonstrating the memory efficiency of our design. For 512×512, we use 4× NVIDIA H100 80GB GPUs and obtain a throughput of about 0.7 steps/s (also with gradient checkpointing) with the same batch size.

## C. Evaluation metrics

For evaluation, we follow the setup of ADM [9] and use the same reference batches provided in their official implementation. Specifically, we generate 50K samples and compute the metrics using OpenAI's official TensorFlow evaluation toolkit. All evaluations are conducted on NVIDIA RTX 4090 or NVIDIA H100 GPUs, except for certain reported numbers that are taken from prior work.

The following gives a concise description of the evaluation metrics used in our experiments.

- **FID** [16] measures the distance between the feature distributions of real and generated images. It is computed using the Inception-v3 network [59], under the assumption that both feature distributions follow multivariate Gaussian distributions.
- **sFID** [43] computes FID using spatial feature maps from intermediate layers of Inception-v3, thereby better capturing the spatial structure of generated images.
- **IS** [53] evaluates only generated images using the Inception-v3 network. It assigns higher scores when the images are classifiable with high confidence (sharp and meaningful) and when the set of generated images is diverse across different categories.
- **Precision and Recall** [32] measure realism and diversity in feature space. Precision reflects the fraction of generated images that look realistic, while recall reflects how much of the real data distribution is covered by the generated samples.

We additionally report computational efficiency. FLOPs are computed using `torchprofile`, and throughput is evaluated under the sampling configurations of DiT [46] with a batch size of 64. FlashAttention-2 [8] and Flash Linear Attention [66] are activated in DiT and DiG, respectively.

## D. Baseline models

The following summarizes the key ideas of the diffusion baselines used for the evaluation.

- **ADM** [9] improves hybrid U-Net architecture for diffusion models and introduces classifier guidance, which enables a trade-off between sample quality and diversity.
- **VDM++** [27] enhances training efficiency by proposing a simple adaptive noise schedule for diffusion models.
- **Simple diffusion** [20] proposes a diffusion model for high-resolution image generation by carefully redesigning the noise schedule and model architecture.
- **CDM** [19] adopts a cascaded framework in which a base model first generates a low-resolution image, and subsequent super-resolution diffusion models progressively refine it to higher fidelity.
- **LDM** [49] proposes latent diffusion models that operate in a compressed latent space, greatly improving training efficiency while retaining high generation quality.
- **U-ViT** [3] adapts Vision Transformers for latent diffusion by introducing long skip connections similar to those in U-Net.
- **MaskDiT** [70] proposes an asymmetric encoder–decoder architecture for diffusion transformers, trained with an auxiliary mask reconstruction task to improve efficiency.
- **SD-DiT** [72] reframes the mask modeling of MaskDiT as a self-supervised discrimination objective.
- **DiT** [46] replaces the hybrid U-Net architecture with a fully transformer-based backbone, introduces AdaIN-zero conditioning to stabilize training, and shows that diffusion transformers scale effectively.
- **SiT** [42] reformulates DiT training by transitioning from discrete diffusion to continuous flow matching.
- **DiG** [71] integrates Gated Linear Attention [66], enabling sub-quadratic complexity efficiency of diffusion transformers.
- **DiC** [62] re-examines purely convolutional denoisers by scaling standard 3×3 convolutional blocks in a U-Net design, introducing sparse skip connections.
- **DiCo** [1] proposes a 3×3 separable convolutional block in a U-Net design, introducing compact channel attention to activate more informative channels.

## E. Additional Scaling Results

As shown in Figure 5, we clearly demonstrate the scalability of FCDM in terms of FID. We also observe consistent scalability across other metrics, including sFID, Inception Score, Precision, and Recall, as reported in Table 8.

In addition, we trained FCDMs using the original SiT implementation [42] to examine whether our proposed design also exhibits scalability under this framework. Following the original implementation details, we trained the model with the flow-matching objective [37, 42]. We used AdamW with a constant learning rate of $1 \times 10^{-4}$, $(\beta_1, \beta_2) = (0.9, 0.999)$, and no weight decay. For sampling, we employed the Euler–Maruyama SDE sampler with 250 steps, setting the final step size to 0.04.

As shown in Table 9, we again observe clear scalability when training the proposed network with the flow-matching objective. Interestingly, under our framework, the flow-matching objective yields better performance at the Small (S) scale, while showing slightly worse results at the Base (B) through XLarge (XL) scales. Nevertheless, these results confirm that the proposed architecture possesses generalized scalability beyond a specific training objective.

| Model | FLOPs (G) | Training Steps | FID ↓ | sFID ↓ | IS ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|---|---|---|---|
| FCDM-S | 3.10 | 50K | 103.93 | 15.03 | 12.01 | 0.3013 | 0.3513 |
| | | 100K | 77.17 | 12.15 | 17.11 | 0.3809 | 0.4473 |
| | | 150K | 66.85 | 11.02 | 20.68 | 0.4136 | 0.5106 |
| | | 200K | 60.33 | 10.70 | 23.71 | 0.4396 | 0.5537 |
| | | 250K | 56.08 | 10.54 | 26.30 | 0.4568 | 0.5609 |
| | | 300K | 53.01 | 10.59 | 28.10 | 0.4687 | 0.5806 |
| | | 350K | 50.44 | 10.29 | 30.08 | 0.4757 | 0.5729 |
| | | 400K | **48.53** | **10.12** | **31.64** | **0.4836** | **0.5840** |
| FCDM-B | 12.20 | 50K | 76.61 | 9.75 | 16.45 | 0.3797 | 0.4569 |
| | | 100K | 50.83 | 8.24 | 26.90 | 0.4854 | 0.5543 |
| | | 150K | 40.60 | 7.53 | 35.35 | 0.5268 | 0.5894 |
| | | 200K | 35.00 | 7.22 | 42.42 | 0.5525 | 0.5981 |
| | | 250K | 31.77 | 7.11 | 47.22 | 0.5665 | 0.6117 |
| | | 300K | 29.26 | 7.03 | 51.62 | 0.5705 | 0.6141 |
| | | 350K | 27.34 | 6.94 | 55.10 | 0.5855 | 0.6127 |
| | | 400K | **26.21** | **6.86** | **58.04** | **0.5908** | **0.6112** |
| FCDM-L | 48.30 | 50K | 55.15 | 8.33 | 22.74 | 0.4655 | 0.5403 |
| | | 100K | 32.03 | 7.10 | 43.20 | 0.5791 | 0.5857 |
| | | 150K | 23.88 | 6.47 | 58.51 | 0.6106 | 0.6052 |
| | | 200K | 19.97 | 6.17 | 69.19 | 0.6312 | 0.6128 |
| | | 250K | 17.33 | 5.99 | 77.79 | 0.6427 | 0.6233 |
| | | 300K | 15.98 | 5.82 | 84.28 | 0.6477 | 0.6245 |
| | | 350K | 14.95 | 5.82 | 87.55 | 0.6527 | 0.6257 |
| | | 400K | **13.83** | **5.65** | **93.31** | **0.6612** | **0.6218** |
| FCDM-XL | 64.60 | 50K | 51.00 | 8.31 | 24.37 | 0.4940 | 0.5475 |
| | | 100K | 27.23 | 6.86 | 49.52 | 0.6108 | 0.5824 |
| | | 150K | 19.25 | 6.15 | 68.62 | 0.6492 | 0.5995 |
| | | 200K | 15.54 | 5.95 | 81.40 | 0.6690 | 0.6051 |
| | | 250K | 13.50 | 5.74 | 91.74 | 0.6785 | 0.6117 |
| | | 300K | 12.31 | 5.64 | 98.58 | 0.6829 | 0.6192 |
| | | 350K | 11.19 | 5.54 | 104.86 | 0.6914 | 0.6227 |
| | | 400K | **10.72** | **5.47** | **108.04** | **0.6864** | **0.6273** |

Table 8. **Performance of FCDMs across scales and training steps on ImageNet 256×256 (Diffusion).** Scaling FCDMs consistently leads to improved generative performance when trained with the diffusion objective.

| Model | FLOPs (G) | Training Steps | FID ↓ | sFID ↓ | IS ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|---|---|---|---|
| FCDM-S | 3.10 | 50K | 103.10 | 13.10 | 12.23 | 0.2863 | 0.3111 |
| | | 100K | 76.95 | 11.12 | 16.96 | 0.3826 | 0.4547 |
| | | 150K | 66.97 | 10.18 | 20.56 | 0.4228 | 0.4953 |
| | | 200K | 60.53 | 9.62 | 23.90 | 0.4428 | 0.5372 |
| | | 250K | 55.94 | 9.53 | 26.16 | 0.4670 | 0.5515 |
| | | 300K | 52.43 | 9.22 | 28.78 | 0.4787 | 0.5632 |
| | | 350K | 49.76 | 9.05 | 31.06 | 0.4866 | 0.5730 |
| | | 400K | **47.84** | **8.91** | **32.89** | **0.4944** | **0.5749** |
| FCDM-B | 12.20 | 50K | 80.10 | 18.48 | 15.46 | 0.3286 | 0.4072 |
| | | 100K | 52.23 | 8.34 | 25.95 | 0.4891 | 0.5450 |
| | | 150K | 42.58 | 7.85 | 33.83 | 0.5346 | 0.5780 |
| | | 200K | 37.01 | 7.51 | 40.72 | 0.5554 | 0.5819 |
| | | 250K | 33.14 | 7.24 | 46.39 | 0.5728 | 0.5855 |
| | | 300K | 30.16 | 7.07 | 51.60 | 0.5883 | 0.5953 |
| | | 350K | 28.40 | 6.99 | 55.07 | 0.5941 | 0.6066 |
| | | 400K | **26.61** | **6.85** | **58.51** | **0.6050** | **0.6017** |
| FCDM-L | 48.30 | 50K | 57.32 | 15.82 | 21.61 | 0.4467 | 0.4872 |
| | | 100K | 33.71 | 7.74 | 40.77 | 0.5852 | 0.5615 |
| | | 150K | 26.10 | 6.94 | 54.79 | 0.6232 | 0.5865 |
| | | 200K | 21.91 | 6.63 | 65.12 | 0.6429 | 0.5909 |
| | | 250K | 19.39 | 6.47 | 73.55 | 0.6557 | 0.5940 |
| | | 300K | 17.59 | 6.31 | 80.47 | 0.6650 | 0.6075 |
| | | 350K | 16.27 | 6.18 | 85.62 | 0.6681 | 0.6064 |
| | | 400K | **15.30** | **6.17** | **90.09** | **0.6751** | **0.6126** |
| FCDM-XL | 64.60 | 50K | 51.21 | 11.89 | 24.21 | 0.5010 | 0.5006 |
| | | 100K | 29.37 | 7.34 | 46.27 | 0.6172 | 0.5680 |
| | | 150K | 22.07 | 6.77 | 63.02 | 0.6572 | 0.5870 |
| | | 200K | 17.98 | 6.34 | 75.71 | 0.6747 | 0.5909 |
| | | 250K | 15.72 | 6.24 | 85.30 | 0.6853 | 0.5991 |
| | | 300K | 14.16 | 6.07 | 92.79 | 0.6952 | 0.6040 |
| | | 350K | 13.06 | 5.97 | 98.22 | 0.6976 | 0.6072 |
| | | 400K | **12.11** | **5.96** | **103.07** | **0.7030** | **0.6070** |

Table 9. **Performance of FCDMs across scales and training steps on ImageNet 256×256 (Flow-Matching).** Scaling FCDMs also demonstrates consistent improvements in generative performance when trained with the flow-matching objective.

## F. Frequency-based analysis

To better highlight the differences between our fully convolutional diffusion model (FCDM) and the transformer-based DiT, we examine the evolution of the spectral energy, defined as the sum of the log-magnitude spectrum of the predicted noise, over the course of the diffusion process (using models trained for 400K iterations at 512×512 resolution). For each predicted noise sample, we compute the 2D Fourier transform, take the magnitude spectrum, and apply a logarithmic scaling $\log(1 + F)$ to compress the dynamic range. We then define the total spectral energy as the sum of all values in this log-magnitude spectrum, which reflects the overall distribution of frequency components. Figure 8 presents the total spectral energy, averaged over 128 validation samples, calculated at each of the 1,000 diffusion timesteps.

Across all diffusion steps, FCDM consistently exhibits higher spectral energy than DiT. This difference is most pronounced in the early-to-middle stages of the diffusion trajectory, where the model must simultaneously capture global structure and fine-grained detail. The elevated energy of FCDM indicates that its predicted noise retains stronger high-frequency components, which can be associated with sharper textures, edges, and local structures. By contrast, DiT produces lower spectral energy, suggesting smoother predictions with fewer high-frequency details. While this observation may provide a partial explanation for the performance gap between FCDM and DiT, further theoretical analysis is required.

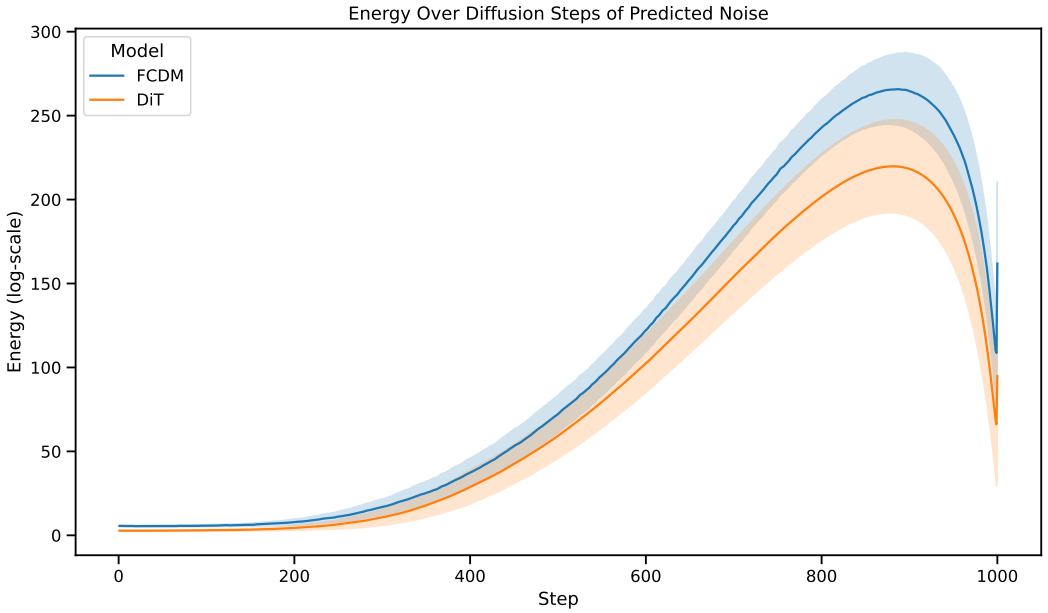



Figure 8. **Spectral energy of predicted noise across diffusion steps.** FCDM consistently exhibits higher spectral energy than DiT across the entire diffusion process, suggesting potential for better preservation of high-frequency components.

## G. Analysis of additional architectural variants

This section provides further details on the ablations in the main manuscript and introduces additional architectural ablation results. Table 6 presents architectural ablations using the Large (L) model on ImageNet at 256×256. We analyze the effects of specific architectural elements, including kernel size, Global Response Normalization (GRN), the inverted bottleneck (channel expansion), the feedforward module, and FCDM blocks. In addition, we evaluate various other design choices to further investigate their impact on performance and efficiency. Interestingly, although ConvNeXt [40, 64] was developed for a different task, we observe similar ablation trends in our experiments.

**Effect of Kernel Size.** Vision Transformers employ non-local self-attention, enabling each layer to access a global receptive field. In contrast, ConvNets traditionally rely on stacking small 3×3 convolutions (popularized by VGGNet [54]), which are efficient on modern GPUs [34]. Our experiments show that reducing the kernel size from the default 7×7 consistently degrades performance, with FID increasing from 19.97 to 20.48 and 21.28. This suggests that larger kernels provide a larger effective receptive field, allowing the model to capture broader context.

Additionally, we explore kernel sizes larger than the default 7×7. Interestingly, as shown in Table 10, performance degrades for kernel sizes beyond 7×7, which may slow down convergence speed and leave the models under-trained within our 200K iteration ablation studies.

Taken together, these results suggest that the 7×7 kernel provides the best balance between expressiveness and convergence speed for our architecture.

| Model Configuration | FLOPs (G) ↓ | FID ↓ | IS ↑ |
|---|---|---|---|
| **FCDM-L** (Default: 7×7 DWConv) | 48.3 | **19.97** | **69.19** |
| 7×7 → 9×9 DWConv | 48.4 | 23.85 | 59.68 |
| 7×7 → 11×11 DWConv | 48.6 | 23.93 | 60.18 |

Table 10. **Ablation study on larger kernels.** We analyze effects of larger kernel sizes. Training iterations are fixed to 200K.

**Effect of Applying DiCo Design Choices.** Following DiCo [1], we investigate whether the architectural design choices from DiCo improve performance in our framework. For a fair comparison, we adjust the channel dimensions ($C$) to match the total FLOPs across variants.

First, we replace the GRN layer with CCA [1], as both modules are designed to enhance channel diversity. As shown in Table 6, using CCA degrades performance compared to GRN, indicating that GRN is a more effective channel enhancement module in our architecture. Second, we remove channel expansion to align with DiCo, which does not apply channel expansion in its convolutional blocks. As shown in Table 6, disabling channel expansion leads to a clear drop in performance, demonstrating that channel expansion is a crucial component of our architecture for maintaining generative capacity. Lastly, we add the feedforward module after the convolutional block to follow the DiCo design, whereas our baseline does not include this component. As shown in Table 6, adding the feedforward module further degrades performance, suggesting that our design is better suited for efficient diffusion modeling.

Overall, these results indicate that our design choices consistently lead to superior performance at matched FLOPs, validating the effectiveness of our architecture.

**Effect of FCDM Blocks.** We replaced FCDM blocks with ResNet blocks [15] using standard 3×3 convolutions. To match FLOPs, the hidden channels were reduced from 512 to 336, given the higher computational cost of standard convolutions compared to separable convolutions. This substitution results in a substantial degradation, with FID increasing from 19.97 to 31.14, indicating that the FCDM block is better suited for this task than the ResNet block.

**Effect of Autoencoders.** Since FCDM operates in latent space, we tested whether performance persists under different VAEs. As shown in Table 11, FCDM consistently outperforms DiT under both SD-VAE [49] and EQ-VAE [30]. Similar to DiT, our model performs best with EQ-VAE, improving further over SD-VAE. These findings suggest that techniques originally proposed to enhance DiT (e.g., stronger VAEs) transfer equally well to FCDM, indicating the potential for further performance improvements.

| Model Configuration | Training iterations | FLOPs (G) ↓ | FID ↓ |
|---|---|---|---|
| *SD-VAE* | | | |
| DiT-XL/2 | 400K | 118.6 | 19.47 |
| FCDM-XL | 400K | **64.6** | **11.57** |
| *EQ-VAE* | | | |
| DiT-XL/2 | 400K | 118.6 | 14.50 |
| FCDM-XL | 400K | **64.6** | **10.72** |

Table 11. **Ablation study on autoencoders.** Across different latent spaces (SD-VAE and EQ-VAE), FCDM consistently outperforms DiT.

**Effect of Isotropic Architecture.** To further analyze our architecture, we evaluated an isotropic variant of our model. As shown in Table 12, despite using only ∼48% of the parameters of DiT-B/2 [46], our isotropic variant achieves a lower FID given the same number of training steps. This again validates the effectiveness and efficiency of our fully convolutional block design.

| Model Configuration | Params (M) | FID ↓ |
|---|---|---|
| DiT-B/2 | 130 | 55.00 |
| **FCDM (Iso.)** | **62 (∼48%)** | **41.15** |

Table 12. **Ablation study on isotropic architecture.** Comparison of our isotropic variant against DiT-B/2 [46]. All models are trained on ImageNet $256 \times 256$ for 200k iterations under identical settings.

**Effect of Replacing Convolution with Local Attention.** We conduct an ablation study to examine the impact of replacing depthwise convolution with local self-attention in our architecture. Specifically, we employ Neighborhood Attention (NA) [14], which implements sliding-window local attention with efficient C++ and CUDA kernels. For a fair comparison, we use a 7×7 attention window to match the receptive field of our 7×7 depthwise convolution. We also adjust the channel dimensions to ensure matched FLOPs across variants.

As shown in Table 13, replacing depthwise convolution with NA results in a significant performance drop in terms of both FID and IS. Moreover, the throughput is substantially reduced, indicating that local self-attention is considerably less efficient in practice.

These results suggest that depthwise convolution is more effective than local self-attention in terms of both performance and efficiency in our architecture, making it a better choice for efficient diffusion modeling.

| Model Configuration | FLOPs (G) ↓ | TP (it/s) ↑ | FID ↓ | IS ↑ |
|---|---|---|---|---|
| **FCDM-L** (Default: DWConv) | 48.3 | **381.3** | **19.97** | **69.19** |
| DWConv → NA* [14] | 45.9 | 122.8 | 29.81 | 50.92 |

Table 13. **Ablation study on replacing depthwise convolution with local self-attention.** Neighborhood Attention (NA) [14] with a 7×7 window is used as a replacement for depthwise convolution. All models are trained for 200K iterations at matched FLOPs. $*$ indicates that $C$ is adjusted to match FLOPs to ensure a fair comparison.

**Effect of Asymmetric Encoder–Decoder Allocation.** Following [21], we investigated whether an asymmetric allocation of compute between the encoder and decoder could outperform the symmetric setup. Intuitively, assigning more compute to the decoder appears advantageous, since upsampling from low to high resolution is more demanding and additionally requires processing skip connections. As shown in Table 14, an asymmetric design slightly improves FID for the Large (L) model (19.97 → 19.55). However, for the XLarge (XL) model, the asymmetric setup performs on par with the symmetric variant. While asymmetric encoder–decoder architectures remain an interesting direction, we adopt the symmetric setup for its simplicity and more straightforward scalability.

**Effect of Block Scaling across U-Net Levels.** We investigated how block scaling across U-Net levels affects model behavior. We compared (1) a default scaling setup, where deeper levels have more blocks, and (2) a uniform setup, where all levels

| Model Configuration | Hidden channel $C$ | Depths | Params (M) | FLOPs (G) ↓ | TP (it/s) ↑ | FID ↓ |
|---|---|---|---|---|---|---|
| *Asymmetric U-Net Ablations* | | | | | | |
| **FCDM-L** (Default: Sym. U-Net) | 512 | [2, 4, 8, 4, 2] | 504.5 | 48.3 | 381.3 | 19.97 |
| Asym. U-Net | 512 | [2, 3, 8, 5, 2] | 504.5 | 48.3 | 381.3 | **19.55** |
| **FCDM-XL** (Default: Sym. U-Net) | 512 | [3, 6, 12, 6, 3] | 698.8 | 64.6 | 272.7 | **15.54** |
| Asym. U-Net | 512 | [3, 5, 12, 7, 3] | 698.8 | 64.6 | 272.7 | **15.54** |
| Asym. U-Net | 512 | [3, 3, 12, 9, 3] | 698.8 | 64.6 | 272.7 | 15.55 |

Table 14. **Ablation study on asymmetric block allocation.** All models trained for 200K iterations under identical training settings.

use the same number of blocks. To ensure fairness, we kept the total number of blocks fixed and matched the total parameter count by adjusting channel sizes. As shown in Table 15, the uniform setup yields better FID for the Large (L) model (19.97 $\rightarrow$ 17.63). However, it significantly degrades efficiency: FLOPs increase from 48.3G to 62.8G, reaching a level similar to the XLarge (XL) model with the scaling setup (62.8G vs. 64.6G), while still yielding worse FID (17.63 vs. 15.54). Given this trade-off, we adopt the default scaling setup in our architecture, since efficiency is a core design goal of this work.

| Model Configuration | Hidden channel $C$ | Depths | Params (M) | FLOPs (G) ↓ | TP (it/s) ↑ | FID ↓ |
|---|---|---|---|---|---|---|
| **FCDM-L** (Default: Scaling) | 512 | [2, 4, 8, 4, 2] | 504.5 | **48.3** | **381.3** | 19.97 |
| Uniform | 600 | [4, 4, 4, 4, 4] | 496.6 | 62.8 | 261.6 | 17.63 |
| **FCDM-XL** (Default: Scaling) | 512 | [3, 6, 12, 6, 3] | 698.8 | 64.6 | 272.7 | **15.54** |

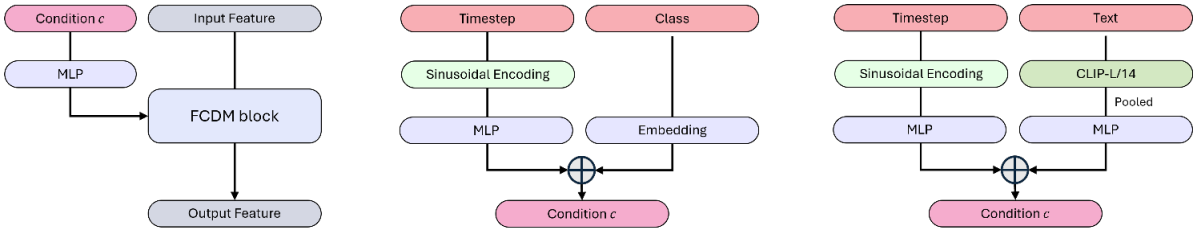
Table 15. **Ablation study on block scaling strategies.** All models are trained on ImageNet $256 \times 256$ for 200k iterations under identical settings. The channel size is adjusted to match the total number of parameters.

## H. Text-to-image generation experiment

Originally, the FCDM block (Figure 9 (a)) is designed for class-conditioned generation, where the conditioning vector $c$ is derived from the diffusion timestep and the class label using the conditioning module illustrated in Figure 9 (b). To adapt FCDM for text-to-image generation, we modify this conditioning module by replacing the class embedding layer with a CLIP text encoder [48] followed by MLP layers, as shown in Figure 9 (c). This design enables the network to be conditioned on both the diffusion timestep and the pooled text representation [11, 47]. By modifying only the conditioning module while preserving the rest of the architecture, we can reuse the same FCDM backbone for text-to-image generation.

We train FCDM-XL with the text-conditioning module from scratch on the MS-COCO [36] training split and evaluate it on the validation split. The model is trained for 100K iterations with a batch size of 256, using the CLIP text encoder to compute pooled text embeddings. Apart from the modified conditioning module, all other training hyperparameters and settings are kept identical to those of the original FCDM configuration.

As shown in Figure 10, FCDM successfully generates images corresponding to the given text descriptions, even when trained for only 100K iterations using pooled text embeddings alone. These findings indicate that text conditioning can be effectively incorporated into FCDM. Nevertheless, extending FCDM to support joint conditioning on full text embeddings, as in MMDiT [11], represents an important direction for future work toward learning richer representations and improving generation performance.




(a) FCDM block (b) Class conditioning (c) Text conditioning

Figure 9. **Conditioning modules for class and text in the FCDM architecture.** (a) FCDM block with conditioning vector $c$, (b) Conditioning module for class conditioning, (c) Conditioning module for text conditioning incorporating the CLIP text encoder.

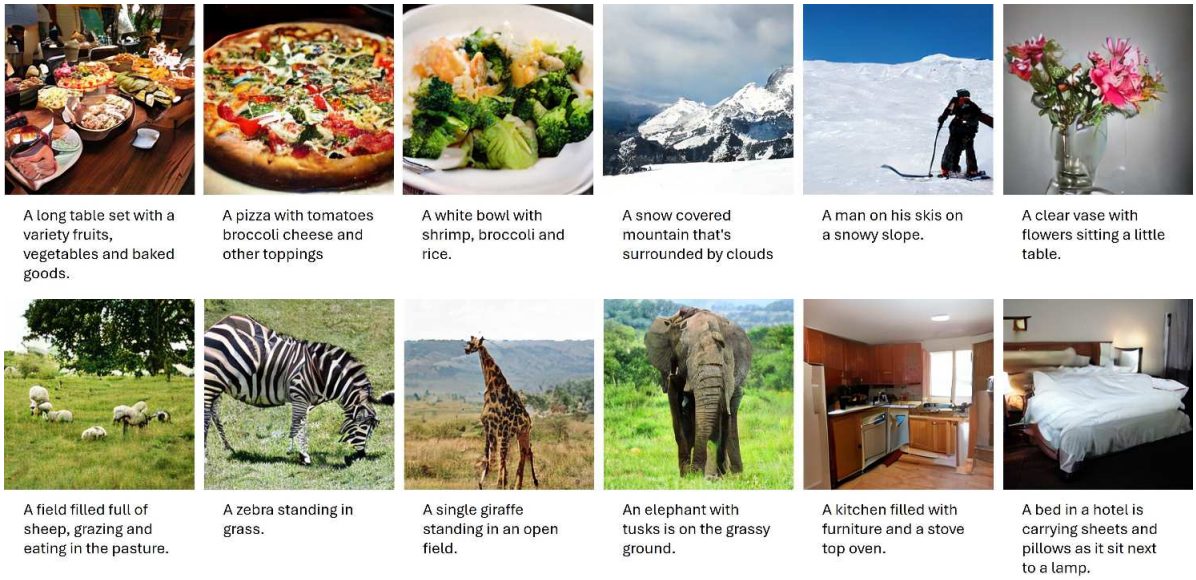



Figure 10. **Qualitative results on text-to-image generation (MS-COCO).** We use classifier-free guidance with scale = 4.0.

# I. More quantitative results

While we position our architecture as a state-of-the-art model in terms of *efficiency*, it is also important to assess how its *performance* relates to other diffusion models. To this end, building upon the results in Table 3 and Table 4, we extend our evaluation to additional baselines [12, 67], focusing on both efficiency and performance.

As presented in Table 16 and Table 17, our findings indicate that FCDM-XL achieves generation quality closely aligned with FasterDiT [67]. Despite this similarity in performance, our architecture provides substantial efficiency benefits: it achieves 3.3× faster inference throughput and requires 1.8× fewer total training FLOPs.

These results reaffirm that our architecture remains highly competitive, offering competitive performance while maintaining state-of-the-art efficiency. We hope that our architecture will serve as a practical and efficient backbone for future research.

| Model | Architecture Type | FLOPs (G) ↓ | Throughput (it/s) ↑ | FID ↓ | IS ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|---|---|---|---|
| MDT-XL/2 (400K) [12] | Transformer | 118.7 | 83.9 | 16.42 | - | - | - |
| FasterDiT-XL/2 (400K) [67] | Transformer | 118.6 | 80.5 | 11.90 | - | - | - |
| **FCDM-XL** (400K) | Conv | **64.6** | **272.7** | **10.72** | 108.0 | 0.69 | 0.63 |
| MDT-XL/2 (1.3M) | Transformer | 118.7 | 83.9 | 9.60 | - | - | - |
| FasterDiT-XL/2 (1M) | Transformer | 118.6 | 80.5 | 8.72 | 121.17 | 0.68 | **0.67** |
| **FCDM-XL** (1M) | Conv | **64.6** | **272.7** | **7.91** | **135.55** | **0.71** | 0.64 |

Table 16. **Additional comparisons on ImageNet 256×256 without guidance.** We report efficiency metrics (FLOPs, throughput) and performance metrics (FID, IS, Precision, Recall) using 50K samples. The best results are shown in **bold**.

| Model | Training epochs | FLOPs (G) ↓ | Training FLOPs (Z) ↓ | Throughput (it/s) ↑ | FID ↓ | IS ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|---|---|---|---|---|
| MDT-XL/2 [12] | 500 | 118.7 | 0.228 | 83.9 | 2.15 | 249.3 | **0.82** | 0.58 |
| FasterDiT-XL/2 [67] | 400 | 118.6 | 0.182 | 80.5 | **2.03** | 270.0 | 0.81 | **0.60** |
| **FCDM-XL** | 400 | **64.6** | **0.099** | **272.7** | **2.03** | **285.7** | 0.81 | 0.59 |

Table 17. **Additional comparisons on ImageNet 256×256 with guidance.** We report efficiency metrics (FLOPs, training FLOPs, throughput) and performance metrics (FID, IS, Precision, Recall) using 50K samples. The best results are shown in **bold**.

## J. More qualitative results

Class label = “husky” (250)

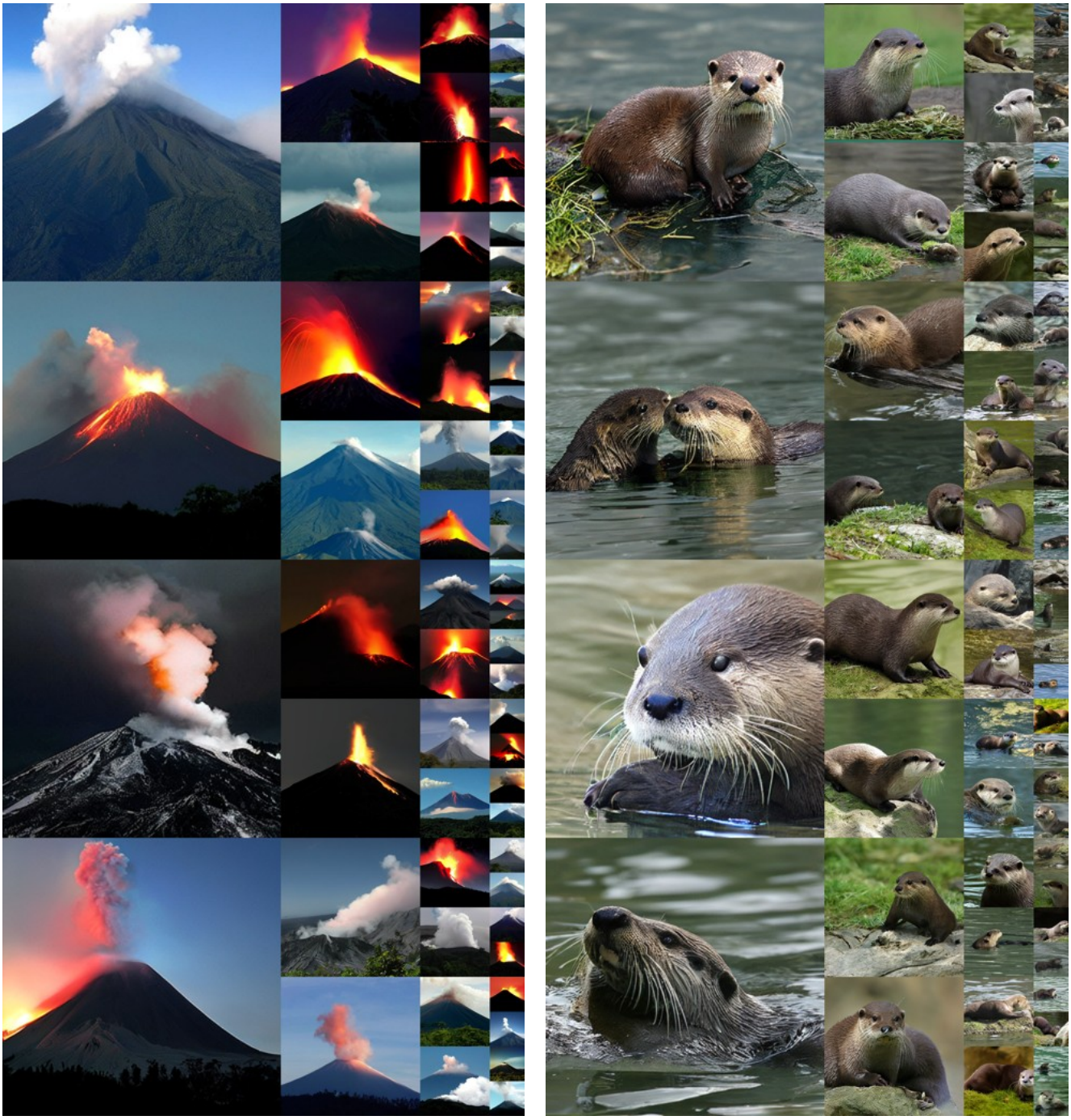
Class label = “sulphur-crested cockatoo” (89)

Figure 11. **Uncurated 512×512 FCDM-XL samples.** Classifier-free guidance scale = 4.0

Class label = “lion” (291)

Class label = “arctic wolf” (270)

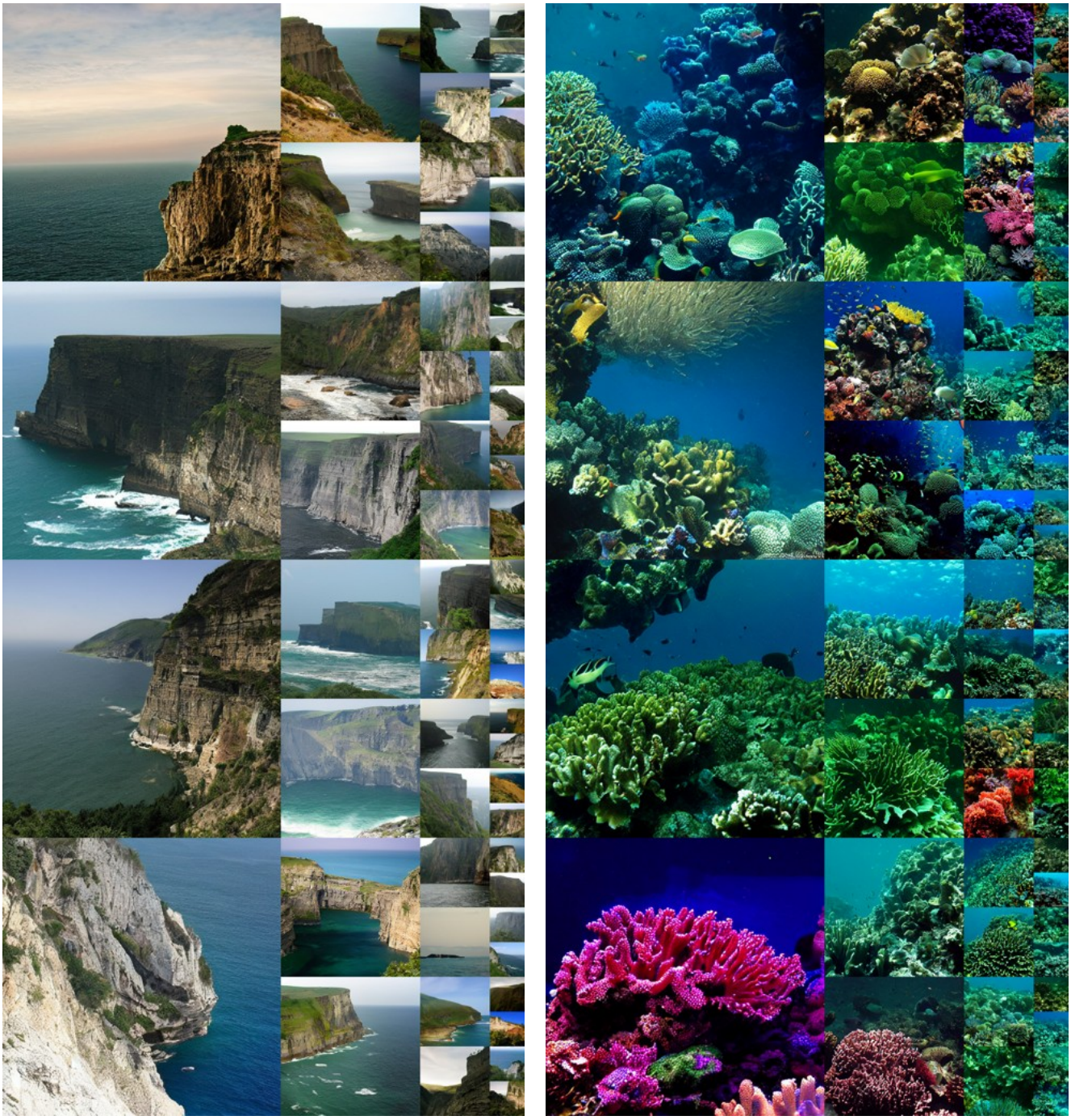
Figure 12. **Uncurated 512×512 FCDM-XL samples.** Classifier-free guidance scale = 4.0



Class label = “volcano” (980)

Class label = “otter” (360)

Figure 13. **Uncurated 512×512 FCDM-XL samples.** Classifier-free guidance scale = 4.0



Class label = “cliff drop-off” (972)

Class label = “coral reef” (973)

Figure 14. **Uncurated 512×512 FCDM-XL samples.** Classifier-free guidance scale = 4.0

Class label = “panda” (388)

Class label = “red panda” (387)

Figure 15. **Uncurated 512×512 FCDM-XL samples.** Classifier-free guidance scale = 4.0

Class label = “macaw” (88)

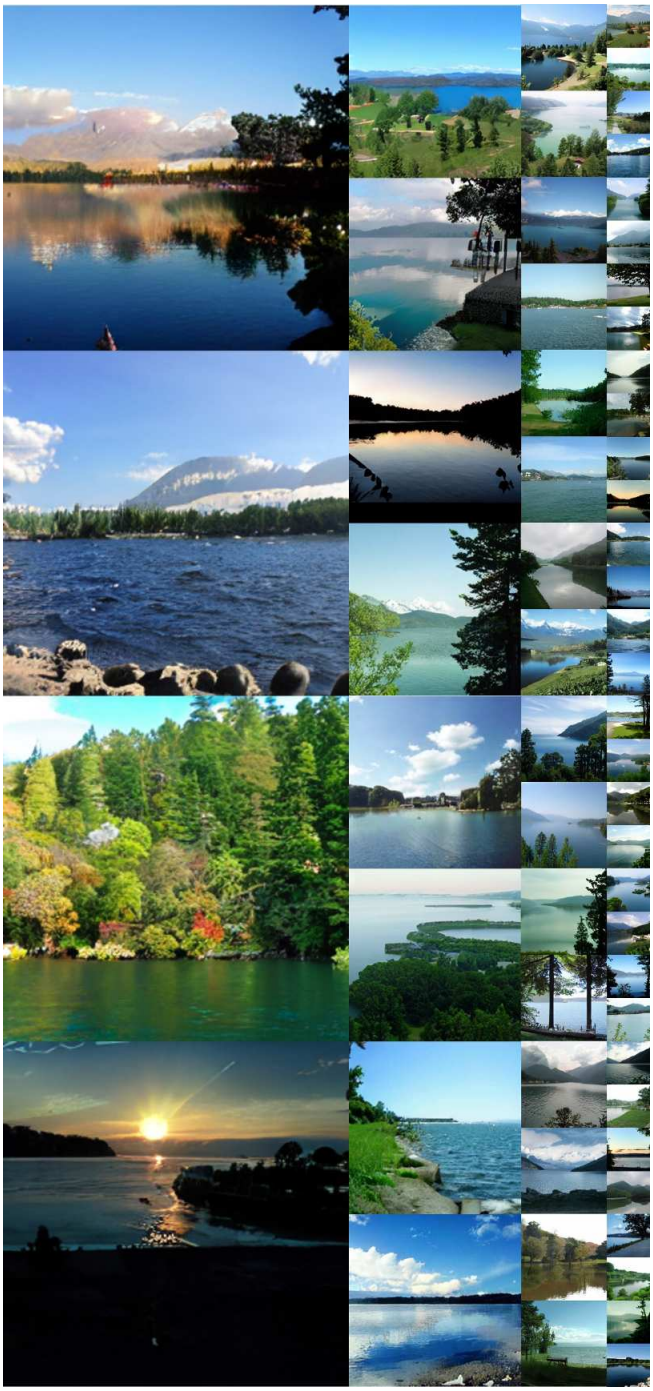
Class label = “arctic fox” (279)

Figure 16. **Uncurated 256×256 FCDM-XL samples.** Classifier-free guidance scale = 4.0

Class label = “golden retriever” (207)

Class label = “loggerhead sea turtle” (33)

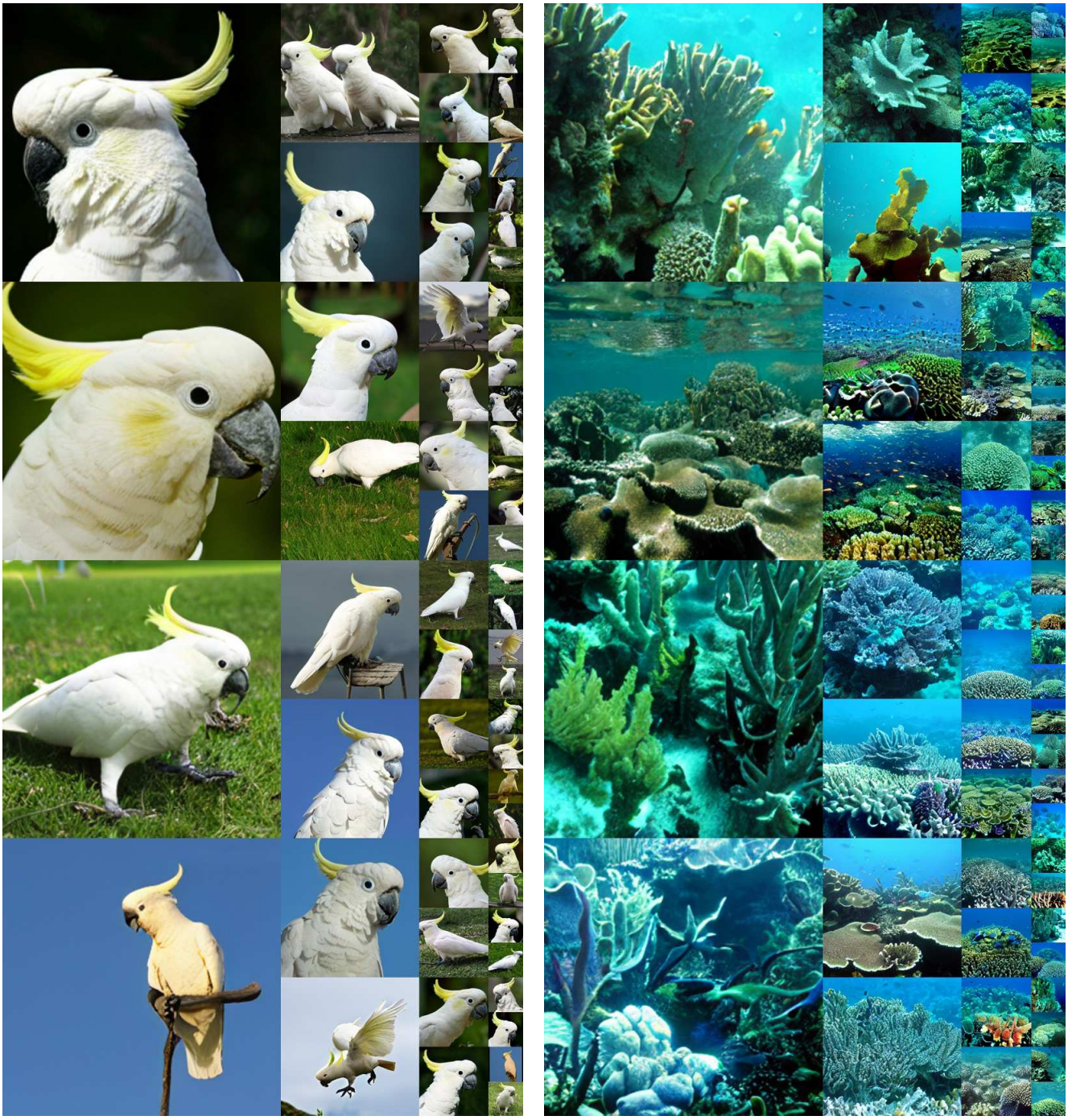
Figure 17. **Uncurated 256×256 FCDM-XL samples.** Classifier-free guidance scale = 4.0



Class label = “lake shore” (975)

Class label = “space shuttle” (812)

Figure 18. **Uncurated 256×256 FCDM-XL samples.** Classifier-free guidance scale = 4.0



Class label = “sulphur-crested cockatoo” (89)

Class label = “coral reef” (973)

Figure 19. **Uncurated 256×256 FCDM-XL samples.** Classifier-free guidance scale = 4.0

Class label = “otter” (360)

Class label = “geyser” (974)

Figure 20. **Uncurated 256×256 FCDM-XL samples.** Classifier-free guidance scale = 4.0

Class label = “cliff drop-off” (972)

Class label = “ice cream” (928)

Figure 21. **Uncurated 256×256 FCDM-XL samples.** Classifier-free guidance scale = 4.0