DIAGNOSING PIXEL-SPACE DIFFUSION TRANSFORMERS: FROM DEPTH-WISE CONNECTIVITY TO LAYER-WISE ATTENTION PATTERNS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent studies have demonstrated the potential of Diffusion Transformers (DiTs) in pixel space. However, pixel-space models still suffer from slow convergence, high training costs, and the lack of native training at high resolutions. In previous work, fine-tuning a $\sim$1.0B text-to-image model at 1024^2 resolution in pixel space required industrial-scale cluster resources. In this work, we diagnose two potential factors underlying the inefficiency of pixel-space DiTs: depth-wise connectivity and layer-wise attention patterns, and propose the Adaptive Connection Image Transformer (AceiT). AceiT achieves reliable convergence with as little as 7 GFLOPs per sample, reducing the computational cost to one quarter of JiT-B/16 and demonstrating substantial gains in computational efficiency. Meanwhile, AceiT achieves faster convergence, alleviating the reliance on representation alignment (e.g., REPA) while remaining free from additional network components (e.g., pixel decoders), thereby offering better simplicity and scalability. More importantly, with only $8\times$ A100 GPUs, AceiT can be pretrained at 1024^2 resolution with up to 2.1B parameters, making native high-resolution training from scratch in pixel space feasible. We will release our code to facilitate further research and development in the community.

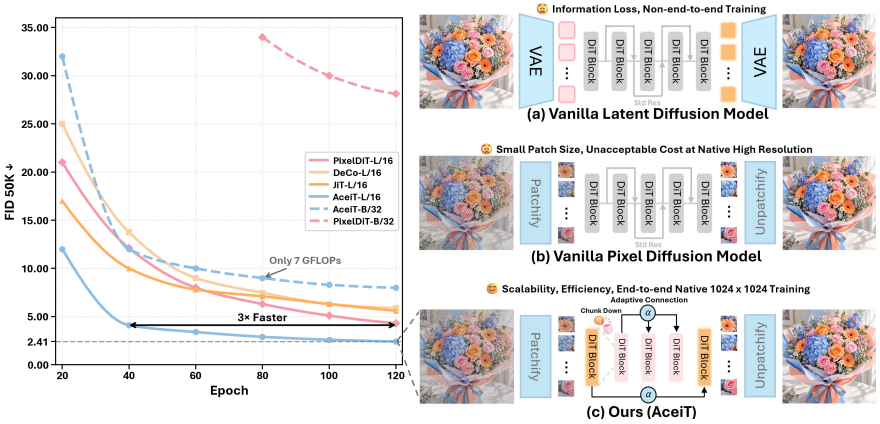


Figure 1: All methods on the left are trained on ImageNet-256 without representation alignment (REPA). AceiT-B/32 and PixelDiT-B/32 operate at approximately 7 and 19 GFLOPs per sample, respectively, while all other models are controlled at around 160 GFLOPs per sample. AceiT achieves significantly faster convergence under the same training settings. The right figure compares vanilla latent diffusion, vanilla pixel-space diffusion, and AceiT, with α denoting the adaptive connection.

1 INTRODUCTION

Although latent diffusion models (Rombach et al., 2022) have demonstrated strong performance in image generation, pixel diffusion models remain an active research focus due to their end-to-end training paradigm and potentially higher ceiling for generation quality, as they avoid the information loss introduced by Variational Autoencoders (VAEs) (Kingma & Welling, 2013).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

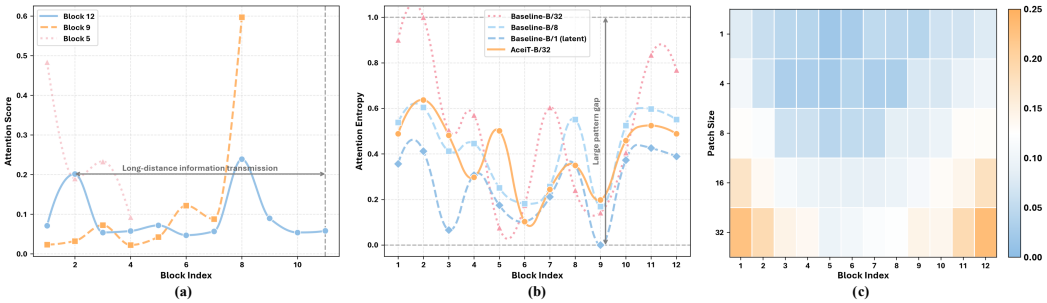


Figure 2: **Diagnosing the inefficiency of pixel-space DiTs:** (a) Adaptive connection mechanism across depth; (b)–(c) Attention patterns analysis within layers. In (b), "latent" denotes a latent diffusion model; all other models operate in pixel space, with entropy values normalized to $[0, 1]$. In (c), we visualize $|1 - \gamma|$ across depth for different patch sizes. See Sec. 3.2 for details.

However, pixel diffusion models still suffer from significant computational bottlenecks: Diffusion Transformers (DiTs) (Peebles & Xie, 2022), while powerful backbones in latent space, become inefficient when directly applied to pixel space (Li & He, 2025). This limitation poses significant challenges for high-resolution generative tasks in pixel space.

To improve efficiency, existing approaches mainly follow two directions. One line of work improves the prediction target (Li & He, 2025; Chen et al., 2026a). As a pioneering approach, JiT replaces velocity prediction (v-prediction) (Liu et al., 2022; Lipman et al., 2022) with clean image prediction (x-prediction), enabling pixel-space image generation with DiTs at manageable computational costs. However, the efficiency still lags significantly behind latent-space models (Yu et al., 2025a; Zheng et al., 2025) and pixel-space methods such as PixelDiT (Yu et al., 2025b) that rely on additional network components.

Another line of work augments DiTs with pixel decoders, offering better generation quality (Wang et al., 2025; Ma et al., 2025; Chen et al., 2025d; Yu et al., 2025b). However, they still suffer from three limitations. First, they rely on the representation alignment (REPA) (Yu et al., 2025a) to improve training efficiency and mitigate instability introduced by the additional components. While REPA has shown effectiveness on small-scale datasets (e.g., ImageNet (Deng et al., 2009)), the recent study (Chefer et al., 2026) reveals that its benefits diminish significantly for large-scale generation. Second, the current approach (Yu et al., 2025b) has enabled pixel-space fine-tuning at 1024^2 resolution, but pretraining these methods at native 1024^2 resolution still incurs excessive computational and memory overhead. Finally, the computational cost of pixel decoders grows rapidly with resolution. Taking PixelDiT as an example, the lightest decoder configuration still requires 394 GFLOPs for one forward pass at 1024^2 resolution. These computational overheads limit the scalability of existing pixel-space models, which typically remain at around 1.0B parameters (Wang et al., 2025; Ma et al., 2025; Yu et al., 2025b).

Therefore, a fundamental question arises: **What makes Diffusion Transformers inefficient in pixel space?**

To diagnose the underlying causes, we analyze pixel-space DiTs from two perspectives: information routing across depth and attention patterns within layers. Our analysis reveals two potential challenges. First, high-resolution generation in pixel space requires additional skip connections to preserve high-dimensional features. However, manually specified connectivity (e.g., U-Net style connections) limits the scalability of DiTs (Ma et al., 2025). Second, pixel-space DiTs can benefit from increasing the patch size to reduce computational cost. However, excessively large patches may lead to ineffective attention patterns and potentially cause generation collapse.

Based on these findings, we propose AceiT (*Adaptive Connection Image Transformer*). It introduces an adaptive connection mechanism for scalable information routing across depth, together with a novel downsampling strategy that facilitates effective attention patterns within layers, enabling fast and efficient convergence. We first evaluate AceiT on ImageNet-256 (Deng et al., 2009), achieving improvements in both generation quality and training efficiency. AceiT-L/16 ($\sim$155 GFLOPs) achieves an FID of 1.75 within 320 epochs, matching the performance of state-of-the-art pixel-space models (Yu et al., 2025b; Chen et al., 2026a) without relying on REPA. AceiT-B/32 ($\sim$7 GFLOPs)

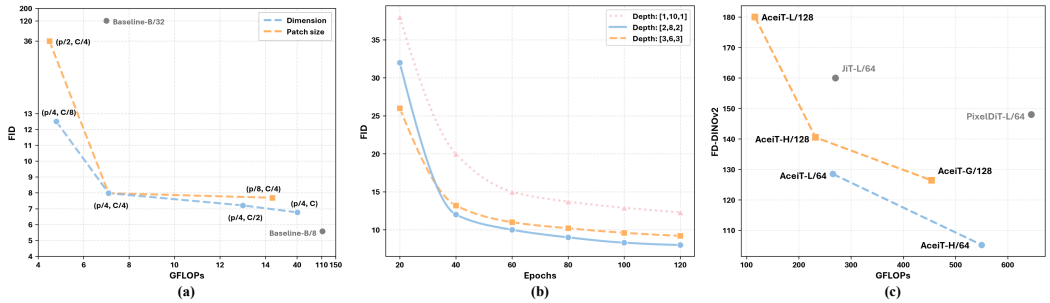


Figure 3: (a) Patch size and hidden dimension ablations for AceiT-B/32. The intermediate blocks use patch size p and hidden dimension C, while we vary the patch size and hidden dimension of the end blocks to examine their impact on FID. (b) Ablation of depth allocation for AceiT-B/32. (c) System-level comparisons on GPIC. Settings: CFG = 4.0. See more details in the appendix.

reaches an FID of 7.98 within 120 epochs, using roughly one-quarter the computation of JiT-B/16 (Li & He, 2025), highlighting the efficiency of our approach. Next, we conduct large-scale text-to-image experiments at 1024^2 resolution on GPIC (Chandrasegaran et al., 2026). We validate AceiT for native high-resolution training from 1.0B to 4.0B parameters, demonstrating the scalability. More importantly, with only $8\times$ A100 GPUs, AceiT can be pretrained directly in pixel space at 1024^2 resolution with up to 2.1B parameters, making native high-resolution training from scratch in pixel space feasible.

Our main contributions are summarized as follows:

- We diagnose two potential challenges in high-resolution pixel-space generation: high-dimensional bottlenecks across depth and ineffective attention patterns within layers.
- We propose AceiT (*Adaptive Connection Image Transformer*), an efficient architecture that enables native high-resolution pixel-space text-to-image generation.
- We demonstrate AceiT's superior convergence efficiency and scalability, enabling native 1024^2 text-to-image training from scratch in pixel space.

2 Inefficiency in Pixel Space

2.1 High-Dimensional Bottlenecks across Depth

We consider a time-dependent process that interpolates between data samples and Gaussian noise. Specifically, given a data sample $x_0 \sim p(x)$ and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$, the intermediate state z_t and velocity v_t at time t can be defined as:

$$z_t = tx_0 + (1-t)\epsilon, \qquad v_t = x_0 - \epsilon. \tag{1}$$

According to *Manifold Assumption* (Chapelle et al., 2006), the data distribution x_0 lies on a low-dimensional manifold embedded in a high-dimensional space. However, the noise component ϵ is distributed over the entire space (Li & He, 2025). In latent-space diffusion models, this issue is alleviated because the spatial dimensions can be substantially compressed by VAEs. In contrast, for pixel-space diffusion models, the high dimensionality of ϵ introduces significant challenges for accurately predicting v_t.

Therefore, JiT proposes to directly predict x_0 and then recover v_t indirectly as $v_t = \frac{x_0 - z_t}{1-t}$. Since the target x_0 lies on a low-dimensional data manifold, x-prediction simplifies the prediction. However, as increasing image details introduces greater intrinsic complexity, the manifold dimensionality still grows with resolution, creating a high-dimensional bottleneck similar to that in v-prediction, which may hinder convergence and degrade generation quality.

2.2 Ineffective Attention Patterns within Layers

Excessively large patch sizes can lead to ineffective attention patterns within layers, and recent studies have explored several possible explanations for this phenomenon. For example, Yu et al.

(2025b); Ma et al. (2025) demonstrate that large patches struggle to capture fine-grained image details. Starodubcev et al. (2026) find that pixel-space DiTs struggle to maintain clean intermediate feature maps at high noise levels, particularly when using large patches.

In fact, the underlying causes may be complex and coupled. Directly adopting existing explanations and architectural designs would make it difficult to disentangle the contributions, while also introducing increasingly complex model designs. Instead, we approach the problem from a different perspective: do large patch sizes make all attention patterns equally ineffective? See Sec. 3.2 for more details.

3 Overcoming the Inefficiency in Pixel Space

3.1 Breaking the High-Dimensional Bottleneck with Cross-Layer Adaptive Connection

Pixel-space DiTs need to preserve high-dimensional features for accurate prediction, which significantly increases the computational overhead. Therefore, a straightforward approach is to directly propagate shallow features to deeper layers through skip connections, thereby reducing the loss of high-dimensional information. However, determining where to place and how many skip connections to use remains non-trivial, posing challenges to scalability. In fact, many early efforts to combine U-Nets with Transformers explored similar designs (Crowson et al., 2024; Hoogeboom et al., 2023; Guo et al., 2026), but they were found to have limited scalability (Ma et al., 2025) and still lag behind modern Transformer architectures (Li & He, 2025; Ma et al., 2025; Chen et al., 2025d; Yu et al., 2025b) in both computation efficiency and generation quality.

In Large Language Models (LLMs), recent work has explored replacing conventional residuals (He et al., 2016) in Transformer blocks with attention residuals (Team et al., 2026). Specifically, the hidden representation at layer l can be written as:

$$h_l = \alpha_{0\to l}\, h_0 + \sum_{i=1}^{l-1} \alpha_{i\to l}\, f_i(h_{i-1}), \tag{2}$$

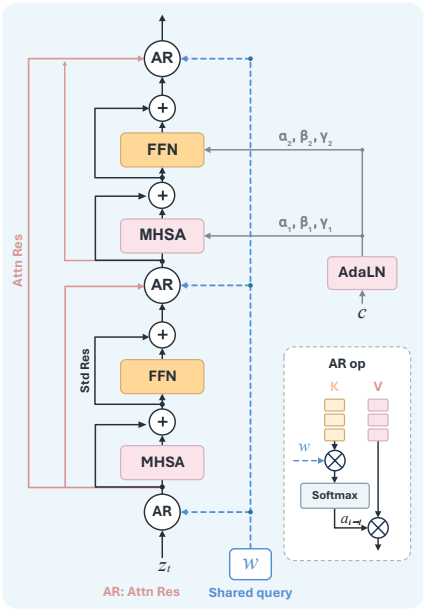


Figure 4: Detailed illustration of query sharing and the hybrid residual mechanism. See Secs. 3.1 and 4.1 for more details.

where $\alpha_{i\to l}$ are layer attention weights satisfying $\sum_{i=0}^{l-1} \alpha_{i\to l} = 1$, and h_0 denotes the network input. Although attention residuals were not designed for pixel-space DiTs, we find that they naturally provide an efficient pathway for preserving high-dimensional information. For example, for the hidden representation h_l at layer l, the network can directly propagate shallow features (e.g., $f_1(h_0)$) to h_l by modulating the contribution of $\alpha_{1\to l}$. More importantly, $\alpha_{1\to l}$ is learned adaptively during training rather than manually specified, effectively addressing the scalability limitations of conventional skip connections. Motivated by these observations, we introduce a new cross-layer connection mechanism, termed *adaptive connection*, specially designed for pixel-space DiTs. We provide a detailed description in Sec. 4.1.

Fig. 2(a) illustrates the mechanism of adaptive connections: deeper blocks (e.g., Block 12) enhance their attention to shallower blocks (e.g., Block 2), directly injecting shallow features into deeper blocks and thereby improving performance (FID improves from 112.56 to 21.80 with v-prediction and from 15.77 to 10.64 with x-prediction), as shown in Tab. 2.

3.2 Establishing Effective Attention Patterns for Fast and Efficient Convergence

To make our diagnosis in Sec. 2.2 concrete, we introduce a learnable scalar γ into the attention computation:

$$Q^* = \gamma Q, \quad A^* = \mathrm{softmax}\left(\frac{Q^* K^\top}{\sqrt{d}}\right). \tag{3}$$

γ provides a direct and controllable way to optimize the attention patterns: When $\gamma > 1$, it proportionally amplifies the attention logits, producing a more concentrated softmax distribution; when $0 < \gamma < 1$, it produces a more uniform pattern. We freeze all parameters of the Baseline models and fine-tune γ. We find that for the Baseline-B/32 (12 blocks across the depth), jointly fine-tuning γ across all blocks yields the largest improvement in generation quality (FID improves from 120.64 to 56.31), while fine-tuning γ only in Blocks 1–2 and 11–12 already achieves most of the improvement (FID 65.82). In contrast, fine-tuning γ in Blocks 3–10 provides limited improvement (FID 102.77). Next, we fine-tune models with different patch sizes (from Baseline-B/1 to Baseline-B/32) and visualize the learned γ values. As shown in Fig. 2(c), as the patch size increases from 1 to 32, the γ values in Blocks 1–2 and 11–12 deviate increasingly from their initial value of 1 compared with those in Blocks 3–10. This observation provides a favorable empirical finding: optimizing the attention patterns of only a small number of blocks at the two ends of the network appears sufficient to achieve substantial improvements. Thus, rather than uniformly reducing the patch size across all blocks, selectively using smaller patches in blocks with larger $|1 - \gamma|$ appears to be an effective alternative. In Fig. 2(b) and Appendix D.1, we visualize the corresponding attention patterns of these blocks through entropy analysis to further characterize the distribution of ineffective patterns.

As shown in Fig. 3(a), reducing the patch size to 8 only at the two ends (Blocks 1–2 and 11–12), with the patch size kept at 32 in the middle (Blocks 3–10), already achieves most of the performance of Baseline-B/8 (FID 6.77 vs. 5.58), requiring only about one-third of the GFLOPs. Next, we reduce the hidden dimension at the two ends of the network to further evaluate the model performance. The results show that the performance degradation caused by reducing the hidden dimension is substantially smaller than that caused by increasing the patch size. When we reduce both the patch size and hidden dimension by a factor of four $(p/4, C/4)$, the model still maintains strong performance (FID 7.98) while keeping the computational cost comparable to Baseline-B/32. We therefore refer to this setting as AceiT-B/32. Finally, we examine the attention entropy distribution of different models. As shown in Fig. 2(b), Baseline-B/32 exhibits larger differences in attention patterns from Baseline-B/1 at the two ends of the network, suggesting the non-uniform depth-wise distribution of these attention patterns. In contrast, AceiT-B/32 narrows this discrepancy. See more details in Appendix D.1.

As a result, AceiT-B/32 improves the FID from 120.64 to 7.98, substantially outperforming the compared methods under a similar computational budget (Li & He, 2025; Ma et al., 2025; Yu et al., 2025b) (shown in Tab. 3). We further evaluate AceiT at larger scales, where it consistently improves both convergence speed and computational efficiency. More results are presented in Sec. 5.3.

4 Detailed Design of AceiT

4.1 Adapting Attention Residuals to Pixel-space Diffusion Transformers

The simplest way to introduce attention residuals into pixel-space DiTs is to retain the AdaLN modulation:

$$\overline{h}_l = h_l + \alpha_1(c) \odot \mathrm{Attn}\left(\gamma_1(c) \odot \mathrm{RMSNorm}(h_l) + \beta_1(c); \mathrm{RoPE}\right), \tag{4}$$

but obtain h_l according to Eq. 2 rather than directly from the preceding layer, where α, γ, and β are modulation projections, while c denotes the conditioning signal.

This design inherits all the components from the original attention residual design that are used to capture complex relationships in language. However, whether these components remain beneficial for pixel-space DiTs is unclear. To figure this out, we conduct a series of ablation experiments. As shown in Tab. 1, we first compare block attention residual with full attention residual. The results show that block attention residual is less effective, with FID increasing to 7.39, potentially due to the substantially smaller model size compared with typical LLMs. We then augment the static



Figure 5: **Qualitative comparison between AceiT and PixelDiT at the native 1024^2 resolution.** See the appendix for prompts, detailed configurations and additional examples.

query with external conditions, making it dynamically conditioned on the input. While this modification provides modest gains in LLMs, we find that it instead destabilizes training and degrades performance in pixel-space DiTs, with FID increasing to 8.66.

We attribute this degradation to the distinct role of attention residuals. Rather than modeling complex semantic relationships in LLMs, attention residuals primarily serve as cross-layer information pathways in DiTs, which benefit more from stable than dynamic queries. This suggests that some of the designs originally introduced for LLMs may be unnecessary in this setting. We therefore further simplify the mechanism. As shown in Fig. 4, we adopt query sharing, where blocks with the same dimension share a single query w, and introduce a hybrid architecture that combines attention residuals with conventional residual connections:

Table 1: **Ablation of attention residual designs.** Components are progressively added following the order from $\mathbb{A}$ to $\mathbb{C}$, while components with the same letter indicate parallel configurations.

	FID↓	Latency (s)↓
$\mathbb{A}$: Block Attn Res	7.39	0.55
$\mathbb{A}$: Full Attn Res	4.80	0.63
$\mathbb{B}$: + *Dynamic Query*	8.66	0.83
$\mathbb{B}$: + *Query Sharing*	**4.73**	<u>0.61</u>
$\mathbb{C}$: + *Hybrid Res*	<u>4.76</u>	**0.39**

$$h_{l+1} = \overline{h}_l + \alpha_2(c) \odot \mathrm{FFN}\big(\gamma_2(c) \odot \mathrm{RMSNorm}(\overline{h}_l) + \beta_2(c)\big)\,, \tag{5}$$

where $\overline{h}_l$ is directly inherited from Eq. 4, without recomputing the depth-attention weights defined in Eq. 2. As shown in Tab. 1, these modifications not only improve generation quality but also substantially reduce computation and memory overhead. Although the hybrid residual incurs a slight performance drop (FID from 4.73 to 4.76), it replaces half of the attention-residual operations with standard residual connections, reducing the reported latency from 0.61 s to 0.39 s. We consider this a favorable trade-off, particularly for high-resolution training. More details on block/full attention residuals and the dynamic query design are provided in Appendix D.3.

4.2 Ensuring Simplicity and Scalability with Chunk Downsampling

In Sec. 3.2, we design two types of DiT blocks. The shallow blocks operate on tensors of shape $(B, L \times 16, C/4)$, while the middle blocks use (B, L, C), where B denotes the batch size, L the sequence length, and C the feature dimension. Connecting these two types of blocks requires a $4\times$ downsampling. A common solution is to adopt a U-shaped architecture (Tian et al., 2024; Crowson et al., 2024) by introducing an additional set of blocks with shape $(B, L \times 4, C/2)$ and connecting them through two successive $2\times$ downsampling. However, this design compromises the simplicity of the original DiT architecture. An alternative is to directly perform $4\times$ downsampling. But as shown in Tab. 2, this design leads to a significant performance degradation compared with the U-shaped architecture (FID from 4.74 to 7.13), potentially due to substantial information loss during aggressive downsampling.

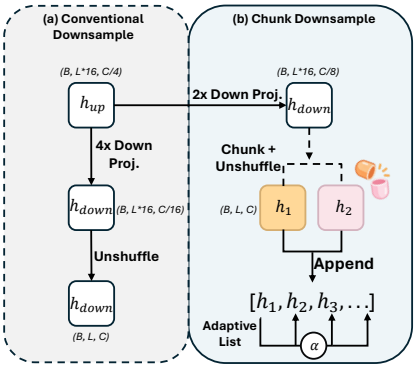


Figure 6: Conventional downsample vs. Chunk downsample.

Table 2: **Component-wise comparison on ImageNet.** Settings: 120 epochs, with CFG, PS for patch size. The final design adopted is highlighted in colors. See more details in Sec. 5.2.

Skip Type	Down Type	Tgt.	PS	FID ↓
ImgNet 256, B Scale				
-	-	v	[16]	112.56
Adaptive	-	v	[16]	21.80
-	-	x	[16]	15.77
Adaptive	-	x	[16]	10.64
U-Net	2×2	x	[4,8,16]	5.57
Adaptive	2×2	x	[4,8,16]	**4.74**
Adaptive	1×4	x	[4,16]	7.13
Adaptive	Chunk(1×1)	x	[4,16]	5.97
Adaptive	Chunk(1×2)	x	[4,16]	<u>4.76</u>
Adaptive	Chunk(1×2)	v	[4,16]	5.08
ImgNet 256, L Scale				
Adaptive	2×2	x	[4,8,16]	2.97
Adaptive	Chunk(1×2)	x	[4,16]	**2.41**
ImgNet 1024, B Scale				
Adaptive	Chunk(1×2)	v	[16,64]	130.87
Adaptive	Chunk(1×2)	x	[16,64]	**7.13**
Adaptive	Chunk(1×2)	v	[4,64]	12.39

Table 3: **System-level comparison on ImageNet-256.** Settings: 120 epochs, with CFG, multi-add = 1 FLOP. Details are provided in the appendix.

Models	REPA	FID ↓	IS ↑	GFLOPs
Baseline-B/32	×	120.64	15.5	7.0
JiT-B/32	✓	150.50	10.6	7.6
DeCo-B/32	✓	50.43	45.7	8.5
PixelDiT-B/32	✓	28.13	70.9	19.0
AceiT-B/32	×	**7.98**	196.5	7.1
Baseline-B/16	×	10.64	113.5	27.6
JiT-B/16	✓	7.76	170.1	25.2
DeCo-B/16	✓	8.83	150.9	24.7
PixelDiT-B/16	✓	7.80	177.1	36.3
AceiT-B/16	×	**4.76**	211.3	27.9
JiT-L/16	✓	3.74	235.1	158
JiT-L/16	×	5.18	180.9	158
DeCo-L/16	✓	2.48	249.0	157
DeCo-L/16	×	6.51	198.2	157
PixelDiT-L/16	✓	**2.31**	283.7	151
PixelDiT-L/16	×	4.29	233.7	151
AceiT-L/16	×	<u>2.41</u>	257.9	155

We therefore propose *chunk downsampling*, a strategy tailored to adaptive connections that supports arbitrary downsampling ratios while flexibly controlling information compression. Instead of aggressively compressing the original channels, chunk downsampling directly partitions the channel dimension into multiple chunks. While conventional residual connections cannot simultaneously process multiple downsampled chunks, adaptive connections are naturally compatible with this design, as they essentially perform attention over multiple elements across depth. The resulting chunks can therefore be directly incorporated into the adaptive connection as additional elements, avoiding the substantial information loss introduced by conventional $4\times$ downsampling.

Tab. 2 further investigates different chunk downsampling designs. For a downsampling ratio of 4, we find that first applying one $2\times$ downsampling and then splitting the features into two chunks (Chunk(1×2)) performs better than directly splitting them into four chunks (Chunk(1×1)). We therefore adopt Chunk(1×2) in our final design, as illustrated in Fig. 6.

5 Experiments

We conduct experiments on ImageNet-1K (Deng et al., 2009) and GPIC (Chandrasegaran et al., 2026). In particular, we investigate the following questions:

(1) How does AceiT perform without REPA? (Sec. 5.3, Tab. 3)

(2) v-prediction and x-prediction, which is better? (Sec. 5.2, Tab. 2)

(3) How can we perform native 1024^2-resolution training from scratch? (Sec. 5.4, Tab. 5, Fig. 3(c))

5.1 Implementation Details

Class-to-Image Generation. We conduct class-conditioned image generation experiments on ImageNet-1K at resolutions of 256^2 and 1024^2. All models are trained with a base learning rate of 1×10^{-4} in Tabs. 2 and 3, using AdamW (Loshchilov & Hutter, 2017) and EMA with decay 0.9999. If REPA is enabled, the weight is set to 0.5. During inference, we enable classifier-free guidance (CFG) (Ho & Salimans, 2022) with interval (Kynkäänniemi et al., 2024). We report Fréchet Inception Distance (FID) (Heusel et al., 2017) and Inception Score (IS) (Salimans et al., 2016), using 50K samples. See more details in the appendix.

Text-to-Image Generation. For text-conditioned experiments, we adopt Qwen3-1.7B (Yang et al., 2025) as the text encoder and train the model from scratch on 30M images from the GPIC dataset for

800K iterations with a base learning rate of 1×10^{-4} and report FD-DINOv2 score (Stein et al., 2023) on the validation set. We first apply $4\times$ super-resolution on GPIC using Real-ESRGAN (Wang et al., 2021), followed by center cropping to 1024^2. We then curate a 120K instruction- and aesthetic-tuning dataset consisting of text-image pairs generated by advanced text-to-image models (Labs, 2024; Wu et al., 2025). We perform an additional 160K iterations of training on this dataset and further evaluate on GenEval (Ghosh et al., 2023) and DPG-Bench (Hu et al., 2024). See more details in the appendix.

5.2 Component-wise Analysis

Patch Size, Hidden Dimension and Depth Allocation. AceiT-B/32 adopts a patch size of 32 and a hidden dimension of 768 in the intermediate blocks. We first investigate the effects of patch size and hidden dimension at the two ends of the network. As shown in Fig. 3(a), using a patch size and hidden dimension of 8 and 192, respectively ($p/4$ and $C/4$), achieves a favorable trade-off between generation quality and computational cost. Further reducing the patch size or increasing the hidden dimension yields marginal gains (FID improves from 7.98 to 7.21), while nearly doubling the computational cost from 7.1 to 13.2 GFLOPs. Fig. 3(b) further evaluates different depth allocations. With 12 blocks in total, allocating 2 blocks to each end and the remaining 8 blocks to the intermediate stages achieves the best performance, consistent with Fig. 2(b) and (c). We therefore adopt these settings or a similar allocation ratio in the subsequent experiments.

Table 4: **Quantitative comparison on ImageNet-256.** The "pre-training" columns list the external models required to obtain the results (note that "Perc." uses a pre-trained VGG classifier (Simonyan & Zisserman, 2014) as perceptual loss (Zhang et al., 2018), and "Self-sup" uses DINOv2 (Oquab et al., 2023) for representation alignment).

	Pre-training							
Models	Token	Perc.	Self-sup.	**Params (M)**	**Epoch**	**GFLOPs**	**FID ↓**	**IS ↑**
Latent Diffusion, Transformer								
DiT-XL/2	SD-VAE	VGG	-	675+49	1400	119	2.27	278.2
REPA, SiT-XL/2	SD-VAE	VGG	DINOv2	675+49	800	119	1.42	305.7
RAE, DiT-XL/1	RAE	VGG	DINOv2	839+415	800	146	**1.13**	262.6
Pixel Diffusion, U-Net								
ADM-G	-	-	-	559	400	1120	4.59	186.7
Pixel Diffusion, Transformer + U-Net Hybrid								
SiD, UViT/2	-	-	-	2000	/	555	2.44	256.3
H-DiT-L/4	-	-	-	557	/	198	3.21	220.6
PixelU-H/16	-	-	DINOv2	1168	320	137	1.77	306.86
Pixel Diffusion, Transformer								
PixelFlow-XL/4	-	-	-	677	320	2909	1.98	282.1
JiT-G/16	-	-	-	2000	600	383	1.82	292.6
AsymFlow-H/16	-	-	-	953	600	182	1.76	312
Pixel Diffusion, Transformer + Pixel Decoder								
DeCo-XL/16	-	-	DINOv2	682	320	118	1.90	303.0
PixelDiT-XL/16	-	-	DINOv2	797	320	156	1.61	292.7
DiP-XL/16	-	-	DINOv2	631	320	/	1.98	282.9
PixNerd-XL/16	-	-	DINOv2	700	320	134	1.95	300.0
Pixel Diffusion, Transformer + Adaptive Connection								
AceiT-L/16	-	-	-	585	320	155	1.75	290.1

Adaptive Connection vs. U-Net Skip Connection. Adaptive connections provide more efficient routing for high-dimensional information than conventional U-Net skip connections. As shown in Tab. 2, simply introducing adaptive connections substantially improves generation quality, reducing the FID from 112.56 to 21.80 with v-prediction and from 15.77 to 10.64 with x-prediction. Moreover, adaptive connections achieve better generation quality through learnable information propagation, attaining an FID of 4.74 compared with 5.57 for U-Net skip connections.

Chunk Downsampling vs. Conventional Downsampling. Chunk downsampling achieves performance comparable to the U-shaped design while preserving the architectural simplicity and scalability. As shown in Tab. 2, 2×2 denotes two successive $2\times$ downsampling operations, with an additional group of DiT blocks inserted between them to form a U-shaped architecture, similar to the design of Tian et al. (2024). Compared with a single $4\times$ downsampling operation (1×4), this design achieves better generation quality (FID 4.74 vs. 7.13), but deviates from the simple, stackable architecture of DiT. In contrast, Chunk(1×2), which first performs one $2\times$ downsampling and then splits the features into two chunks, achieves comparable performance at the B scale (FID 4.76) and surpasses the U-shaped design at the L scale (FID 2.41 vs. 2.97), demonstrating the scalability.

x-prediction vs. v-prediction. x-prediction and v-prediction perform comparably at 256^2 resolution (FID 4.76 vs. 5.08), whereas x-prediction shows a clear advantage at 1024^2 resolution. At 1024^2 resolution, we first increase the patch size from $[4, 16]$ to $[16, 64]$ to maintain a comparable computational cost. Interestingly, only x-prediction converges successfully under this setting (FID 7.13), suggesting that v-prediction may still require smaller patch sizes. To verify, we reduce the patch size at both ends of the network back to 4, and apply chunk downsampling to first perform

one $2\times$ downsampling and then split the features into 8 chunks, yielding an effective $16\times$ downsampling ratio. With this design, the v-prediction model successfully recovers convergence (FID improves from 130.87 to 12.39). This further highlights the advantage of chunk downsampling: compared with a U-shaped architecture, which requires four successive $2\times$ downsampling operations and multiple block configurations to achieve a total $16\times$ downsampling ratio, chunk downsampling achieves the same effective reduction with a substantially simpler design space. Finally, since v-prediction incurs additional computational overhead by splitting the dimension into eight chunks, whereas x-prediction requires only two, we use x as the prediction target.

5.3 System-level Comparison

As shown in Tab. 3, we compare AceiT with several advanced approaches (Li & He, 2025; Yu et al., 2025b; Ma et al., 2025) across three computational budgets. Around 10 GFLOPs, AceiT-B/32 shows a clear advantage, reducing FID below 8 within 120 epochs, highlighting the efficiency of our approach. Around 160 GFLOPs, AceiT-L/16 achieves substantially faster convergence than competing methods without relying on REPA (FID 2.41 vs. 4.29). This demonstrates the potential of AceiT for large-scale high-resolution text-to-image generation.

Table 5: **Comparison on GenEval and DPG.** We compute the TFLOPs (one forward pass) of several methods in the FLOP column. "$\rightarrow$" denotes low-resolution pretraining followed by high-resolution fine-tuning.

Models	Params (B)	GenEval ↑	DPG ↑	FLOP ↓	Throughput (samples/s) ↑
Latent Space, Native 1024					
PixArt-Σ	0.6	0.54	80.5	-	0.40
SDXL	2.6	0.55	74.7	-	0.15
FLUX.1-dev	12.0	0.67	**84.0**	-	0.04
DALL-E 3	/	0.67	83.5	-	-
Pixel Space, 256 → 512					
PixelFlow	0.9	0.60	77.9	-	0.05
PixNerd-XXL/16	1.2	0.73	80.9	0.71	1.04
DeCo-XXL/16	1.1	**0.86**	81.4	0.65	1.07
PixelGen-XXL/16	1.1	0.79	/	0.69	1.05
Pixel Space, 512 → 1024					
PixelDiT-T2I/16	1.3	0.74	83.5	2.18	0.33
Pixel Space, Native 1024					
AceiT-H/64	2.1	<u>0.82</u>	<u>83.9</u>	**0.55**	**0.92**

Further Comparison. We further train AceiT-L/16 for 200 additional epochs and compare it with the recent advanced diffusion models. The results are shown in Tab. 4.

5.4 Towards Native 1024^2 Training from Scratch

We divide the text-to-image experiments into two parts. We first compare AceiT with PixelDiT and JiT on the GPIC dataset at 1024^2 resolution, using the same network depth, hidden dimension, prediction target, patch size, and training settings for all methods. As shown in Fig. 3(c), AceiT-L/64 achieves substantially higher computational efficiency and faster convergence than the competing methods, reducing FD-DINOv2 from 148.0 to 128.5. Fig. 5 further presents qualitative comparisons between AceiT-L/64 and PixelDiT-L/64. Under comparable settings, PixelDiT-L/64 incurs higher computational cost due to its pixel decoder, which introduces an additional 394 GFLOPs. We further scale AceiT to the H and G sizes, and the consistent improvements (FD-DINOv2 improves from 128.5 to 105.2) shown in Fig. 3(c) demonstrate its scalability.

We next compare AceiT-H/64 with existing open-source methods on GenEval and DPG-Bench. The throughput reported in Tab. 5 is measured on a single NVIDIA A100 GPU. At the native 1024^2 resolution, AceiT achieves substantially faster inference, with a throughput of 0.92 samples/s and 0.55 TFLOPs per forward pass, while still maintaining competitive generation quality (GenEval: 0.82; DPG-Bench: 83.9).

6 Conclusion

We propose AceiT, making native high-resolution training from scratch in pixel space feasible. While AceiT follows a minimalist design philosophy, this does not imply that techniques such as representation alignment and pixel decoders are unnecessary. We hope future work can integrate these techniques with AceiT to further improve generation quality. Finally, and more importantly, we diagnose potential challenges of pixel-space DiTs in high-resolution generation. AceiT is not necessarily an optimal solution to these challenges; rather, we hope our findings can inspire future designs of more efficient pixel-space diffusion models.

AI USE STATEMENT

In our work, we used LLMs for the following purposes:

- **Using LLMs to aid paper writing.** We used GPT-5.6 to polish wording, correct grammatical errors, and assist with writing some LaTeX code.
- **Using LLMs for retrieval and discovery.** We used GPT-5.6 to help organize and summarize some recent related work.
- **Using LLMs for code optimization.** We used GPT-5.6 to assist with implementing parts of the code and improving the runtime efficiency of some components.

REPRODUCIBILITY STATEMENT

We provide the necessary implementation details and hyperparameters in the main paper and appendix. All of our code and analysis results will be open-sourced to facilitate further research.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

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*, pp. 22669–22679, 2023.

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

Keshigeyan Chandrasegaran, Kyle Sargent, Suchir Agarwal, Michael Jang, Michael Poli, Juan Carlos Niebles, Justin Johnson, Jiajun Wu, and Li Fei-Fei. Gpic: A giant permissive image corpus for visual generation. *arXiv preprint arXiv:2605.30341*, 2026.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien (eds.). *Semi-Supervised Learning*. The MIT Press, 09 2006. ISBN 9780262255899. doi: 10.7551/mitpress/9780262033589.001.0001. URL https://doi.org/10.7551/mitpress/9780262033589.001.0001.

Hila Chefer, Patrick Esser, Dominik Lorenz, Dustin Podell, Vikash Raja, Vinh Tong, Antonio Torralba, and Robin Rombach. Self-supervised flow matching for scalable multi-modal synthesis. *arXiv preprint arXiv:2603.06507*, 2026.

Hansheng Chen, Jan Ackermann, Minseo Kim, Gordon Wetzstein, and Leonidas Guibas. Asymmetric flow models. *arXiv preprint arXiv:2605.12964*, 2026a.

Hao Chen, Yujin Han, Fangyi Chen, Xiang Li, Yidong Wang, Jindong Wang, Ze Wang, Zicheng Liu, Difan Zou, and Bhiksha Raj. Masked autoencoders are effective tokenizers for diffusion models. In *Forty-second International Conference on Machine Learning*, 2025a.

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

Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, Yao Lu, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. *arXiv preprint arXiv:2410.10733*, 2024b.

Junyu Chen, Dongyun Zou, Wenkun He, Junsong Chen, Enze Xie, Song Han, and Han Cai. Dc-ae 1.5: Accelerating diffusion model convergence with structured latent space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19628–19637, 2025b.

Shoufa Chen, Chongjian Ge, Shilong Zhang, Peize Sun, and Ping Luo. Pixelflow: Pixel-space generative models with flow. *arXiv preprint arXiv:2504.07963*, 2025c.

Zhennan Chen, Junwei Zhu, Xu Chen, Jiangning Zhang, Xiaobin Hu, Hanzhen Zhao, Chengjie Wang, Jian Yang, and Ying Tai. Dip: Taming diffusion models in pixel space. *arXiv preprint arXiv:2511.18822*, 2025d.

Zhennan Chen, Junwei Zhu, Xu Chen, Jiangning Zhang, Jiawei Chen, Zhuoqi Zeng, Wei Zhang, Chengjie Wang, Jian Yang, and Ying Tai. L2p: Unlocking latent potential for pixel generation. *arXiv preprint arXiv:2605.12013*, 2026b.

Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. *arXiv preprint arXiv:2401.11605*, 2024.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

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

Lijie Fan, Tianhong Li, Siyang Qin, Yuanzhen Li, Chen Sun, Michael Rubinstein, Deqing Sun, Kaiming He, and Yonglong Tian. Fluid: Scaling autoregressive text-to-image generative models with continuous tokens. In *International Conference on Learning Representations*, volume 2025, pp. 100218–100231, 2025.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36: 52132–52152, 2023.

Zipeng Guo, Lichen Ma, Yu He, Xiaolong Fu, Jingling Fu, Junshi Huang, and Yan Li. Pixelu: A u-shaped transformer for efficient end-to-end pixel diffusion. *arXiv preprint arXiv:2606.27760*, 2026.

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*, pp. 770–778, 2016.

Yu He, Lichen Ma, Zipeng Guo, Xinyuan Shan, Jingling Fu, Dong Chen, Junshi Huang, and Yan Li. Hyperdit: Hyper-connected transformers for high-fidelity pixel-space diffusion. *arXiv preprint arXiv:2605.15741*, 2026.

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.

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

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

Emiel Hoogeboom, Thomas Mensink, Jonathan Heek, Kay Lamerigts, Ruiqi Gao, and Tim Salimans. Simpler diffusion: 1.5 fid on imagenet512 with pixel-space diffusion. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 18062–18071, 2025.

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

Dengyang Jiang, Ruoyi Du, Zhennan Chen, Dongyang Liu, Zanyi Wang, Mingzhe Zheng, Xiangpeng Yang, Huanqia Cai, Aiming Hao, Yuming Jiang, et al. An empirical study of training pixel-space text-to-image diffusion models. *arXiv preprint arXiv:2608.16887*, 2026.

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

Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. *Advances in Neural Information Processing Systems*, 37:122458–122483, 2024.

Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024.

Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. Repa-e: Unlocking vae for end-to-end tuning of latent diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18262–18272, 2025.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

Tianyu Li, Dongchen Han, Zixuan Cao, Haofeng Huang, Mengyu Zhou, Ming Chen, Erchao Zhao, Xiaoxi Jiang, Guanjun Jiang, and Gao Huang. Siamesenorm: Breaking the barrier to reconciling pre/post-norm. *arXiv preprint arXiv:2602.08064*, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

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

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*, pp. 23–40. Springer, 2024.

Zehong Ma, Longhui Wei, Shuai Wang, Shiliang Zhang, and Qi Tian. Deco: Frequency-decoupled pixel diffusion for end-to-end image generation. *arXiv preprint arXiv:2511.19365*, 2025.

Zehong Ma, Ruihan Xu, and Shiliang Zhang. Pixelgen: Improving pixel diffusion with perceptual supervision. *arXiv preprint arXiv:2602.02493*, 2026.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

Filippo Passerini and Simone Severini. The von neumann entropy of networks. *arXiv preprint arXiv:0812.2597*, 2008.

William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.

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 *International Conference on Learning Representations*, volume 2024, pp. 1862–1874, 2024.

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*, pp. 10684–10695, 2022.

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.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Nikita Starodubcev, Ilia Sudakov, Ilya Drobyshevskiy, Artem Babenko, and Dmitry Baranchuk. Registers matter for pixel-space diffusion transformers. *arXiv preprint arXiv:2605.16147*, 2026.

George Stein, Jesse Cresswell, Rasa Hosseinzadeh, Yi Sui, Brendan Ross, Valentin Villecroze, Zhaoyan Liu, Anthony L Caterini, Eric Taylor, and Gabriel Loaiza-Ganem. Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models. *Advances in Neural Information Processing Systems*, 36:3732–3784, 2023.

Kimi Team, Guangyu Chen, Yu Zhang, Jianlin Su, Weixin Xu, Siyuan Pan, Yaoyu Wang, Yucheng Wang, Guanduo Chen, Bohong Yin, Yutian Chen, Junjie Yan, Ming Wei, Y. Zhang, Fanqing Meng, Chao Hong, Xiaotong Xie, Shaowei Liu, Enzhe Lu, Yunpeng Tai, Yanru Chen, Xin Men, Haiqing Guo, Y. Charles, Haoyu Lu, Lin Sui, Jinguo Zhu, Zaida Zhou, Weiran He, Weixiao Huang, Xinran Xu, Yuzhi Wang, Guokun Lai, Yulun Du, Yuxin Wu, Zhilin Yang, and Xinyu Zhou. Attention residuals, 2026.

Jiayan Teng, Wendi Zheng, Ming Ding, Wenyi Hong, Jianqiao Wangni, Zhuoyi Yang, and Jie Tang. Relay diffusion: Unifying diffusion process across resolutions for image synthesis. *arXiv preprint arXiv:2309.03350*, 2023.

Yuchuan Tian, Zhijun Tu, Hanting Chen, Jie Hu, Chao Xu, and Yunhe Wang. U-dits: Downsample tokens in u-shaped diffusion transformers, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Shuai Wang, Ziteng Gao, Chenhui Zhu, Weilin Huang, and Limin Wang. Pixnerd: Pixel neural field diffusion. *arXiv preprint arXiv:2507.23268*, 2025.

Xianbang Wang, Hanhong Zhao, Yiyang Lu, Kangyang Zhou, Linrui Ma, and Kaiming He. Minit2i: A minimalist baseline for text-to-image generation, 2026. URL `https://peppaking8.github.io/#/post/minit2i`.

Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pp. 1905–1914, October 2021.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International conference on machine learning*, pp. 10524–10533. PMLR, 2020.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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

Yongsheng Yu, Wei Xiong, Weili Nie, Yichen Sheng, Shiqiu Liu, and Jiebo Luo. Pixeldit: Pixel diffusion transformers for image generation. *arXiv preprint arXiv:2511.20645*, 2025b.

Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Joshua M Susskind. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*, pp. 40770–40803. PMLR, 2023.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2510.11690*, 2025.

A Related Work

A.1 Latent Space Diffusion Models

Latent diffusion models (LDMs) (Rombach et al., 2022) perform the denoising process in a compressed latent space obtained from an autoencoder (Kingma & Welling, 2013), rather than directly operating in the high-dimensional pixel space. This design substantially reduces spatial redundancy, enabling more efficient training and scaling to higher resolutions under acceptable computational budgets (Chen et al., 2024b; 2025b;a).

Early LDMs (Rombach et al., 2022; Podell et al., 2024; Betker et al., 2023) adopt U-Net backbones, while more recent approaches replace them with transformer architectures (Peebles & Xie, 2022; Bao et al., 2023; Ma et al., 2024; Chen et al., 2024a), improving scalability and global modeling capacity. Beyond backbone design, several studies (Yu et al., 2025a; Leng et al., 2025) explore improving latent diffusion through representation alignment and joint optimization. REPA-E (Leng et al., 2025) explores an end-to-end training strategy that jointly optimizes the VAE and the DiT backbone. However, this formulation can introduce instability when combined with diffusion objectives, since the latent representation evolves throughout training, resulting in continually shifting denoising targets (Ma et al., 2025). By contrast, pixel-space diffusion operates in a fixed representation space, providing consistent supervision signals and thus more stable optimization.

A.2 Pixel-Space Diffusion Models

Pixel-space diffusion models operate directly on raw images and were among the earliest formulations of diffusion-based generative modeling (Dhariwal & Nichol, 2021). While they can achieve strong fidelity by avoiding compression artifacts, their development has been constrained by the high dimensionality of pixel space, which leads to substantial computational and memory overhead, especially at high resolutions (Dhariwal & Nichol, 2021; Hoogeboom et al., 2023). Early work (Hoogeboom et al., 2023; 2025; Crowson et al., 2024) typically adopted either a standalone U-Net or hybrid architectures that combine U-Net with attention mechanisms; such designs are often computationally intensive and offer limited scalability.

To improve efficiency, a line of work (Chen et al., 2025c; Teng et al., 2023) decomposes the generation process into multi-scale procedures. Relay Diffusion (Teng et al., 2023) trains separate models for each scale, leading to higher cost and two-stage optimization. Pixelflow (Chen et al., 2025c) uses one model across all scales and needs a complex denoising schedule that slows down inference.

More recently, transformer-based architectures (Li & He, 2025; Ma et al., 2025; Yu et al., 2025b; Chen et al., 2025d; Wang et al., 2025; Starodubcev et al., 2026; Ma et al., 2026; He et al., 2026) have been introduced to pixel diffusion, offering improved scalability compared to convolutional backbones. Additional techniques focus on stabilizing training and improving efficiency, such as predicting clean images (Li & He, 2025) to anchor the generation process, introducing lightweight neural field representations (Wang et al., 2025) to parameterize pixel-level outputs, or incorporating a pixel decoder module (Ma et al., 2025; Yu et al., 2025b; Chen et al., 2025d) to enhance fine-grained image details. Although these methods have demonstrated strong performance on ImageNet, they remain difficult to scale to high-resolution text-to-image generation due to the efficiency issue. To the best of our knowledge, PixelDiT is the first to enable fine-tuning at 1024^2 resolution. However,

Table 6: **Configuration details of AceiT.** The table details the configurations of the endpoint and intermediate blocks. Class-to-image experiments are conducted at the B and L scales, while text-to-image experiments are conducted at the L, H, and G scales. Since AceiT-L is used in both settings, its parameter count is reported as 0.585/0.97B, where the latter includes the additional parameters introduced by the text adapter blocks in the text-to-image setting.

	Baseline-B	**AceiT-B**	**AceiT-L**	**AceiT-H**	**AceiT-G**
Architecture					
Hidden dim	768	[192, 768]	[384, 1536]	[512, 2048]	[704, 2816]
Depth	12	[2, 8, 2]	[3, 12, 3]	[3, 16, 3]	[4, 16, 4]
Heads	12	[3, 12]	[6, 24]	[8, 32]	[11, 44]
Patch size	p	$[p/4, p]$	$[p/4, p]$	$[p/4, p]$	$[p/4, p]$
Params (B)	0.131	0.103	0.585 / 0.97	2.1	4.0
Text to Image Configurations					
Image size			Native 1024^2		
Text encoder			Qwen3-1.7B		
Text adapter blocks depth			4		
Text adapter blocks dim			2048		
Optimization					
Base learning rate	1e−4	1e−4	1e−4 → 5e−5	1e−4	1e−4
Optimizer			AdamW, β_1=0.9, β_2=0.95		
Time sampler			$\text{logit}(t) \sim \mathcal{N}(\mu, \sigma^2), \mu = -0.8, \sigma = 0.8$		
Noise scale			*image_size* / 256		
EMA decay			0.9999		
Class drop prob.			0.1		
Clip of $(1-t)$ in division			0.05		
Inference					
Sampler			Heun / Euler		
Sampling steps			50–100		
CFG interval min sweep range			[0.0, 0.125]		
CFG interval max sweep range			[0.8, 1.0]		
CFG scale sweep range			[1.0, 8.0]		

it still requires industrial-scale computational resources and remains challenging to train directly at the native resolution.

A.3 High-Resolution Image Generation in Pixel Space

Although native high-resolution pretraining in pixel space remains challenging, recent works (Chen et al., 2026a;b; Jiang et al., 2026) have explored a series of fine-tuning approaches and demonstrated the feasibility of high-resolution image generation directly in pixel space. These methods leverage priors from the latent space by directly inheriting the parameters of mature latent-space diffusion models (Labs, 2024; Wu et al., 2025) and subsequently fine-tuning them in pixel space, enabling rapid convergence at high resolutions. However, such approaches still face several limitations. First, inheriting parameters from a latent-space model requires the pixel-space model to adopt the same model architecture and scale, which limits architectural flexibility. Second, to ensure stable training, these methods often freeze a large portion of the intermediate network and fine-tune only a few layers at the two ends, potentially limiting the attainable performance. Finally, Jiang et al. (2026) further show that the choice of adaptation data substantially affects convergence: source-model-generated samples facilitate rapid adaptation, while mixing them with high-quality real images improves the overall trade-off, but also limits the applicability of this adaptation strategy. These limitations motivate the need for native high-resolution training in pixel space from scratch.

Table 7: **Configuration details of competing methods.** We follow the recommended configurations of the respective methods and keep the computational cost comparable in terms of GFLOPs whenever possible. PixelDiT has a larger parameter count due to the additional parameters introduced by its pixel decoder.

	DeCo-B	JiT-B	PixelDiT-B	DeCo-L	JiT-L(C2I)	JiT-L(T2I)	PixelDiT-L(C2I)	PixelDiT-L(T2I)
Patch-level Architecture								
Patch-wise dim	768	768	768	1280	1280	1536	1152	1536
Patch-wise depth	12	12	12	28	28	18	26	16
Heads	12	12	12	16	16	24	16	24
Params (B)	0.157	0.133	0.338	0.929	0.835	1.3	0.797	2.9
Pixel-level Architecture								
Pixel-wise dim	32	/	16	32	/	/	16	16
Pixel-wise depth	3	/	2	3	/	/	4	2
Representation Alignment								
Align depth (if enabled)	6	6	6	8	8	/	8	/
Encoder	DINOv2							
Weight	0.5							

B Model Configurations

B.1 Detailed Configurations of AceiT

Tab. 6 summarizes the detailed configurations of AceiT models at different scales. Baseline-B follows the same configuration as AceiT, with the downsampling design removed. For class-to-image experiments, we first train the models for 120 epochs with a learning rate of $1e{-}4$, with the results reported in Tab. 2 and Tab. 3 in the main paper. We then reduce the learning rate to $5e{-}5$ and further train AceiT-L/16 for 200 epochs, with the results reported in Tab. 4.

For text-to-image experiments, following the design of prior work (Ma et al., 2025; Wang et al., 2026; Fan et al., 2025), we append several text adapter blocks after the frozen text encoder to improve text-feature alignment. We first train the models for 800K iterations on the GPIC dataset, with the results reported in Fig. 3(c). We then continue training for another 160K iterations on the instruction-tuning dataset, with the results reported in Tab. 5.

B.2 Detailed Configurations of Compared Methods

Tab. 7 provides the detailed configurations of the compared methods. For the L-scale models, the recommended configurations reported in the original papers differ substantially between class-to-image (C2I) and text-to-image (T2I) generation. We therefore report them separately as C2I and T2I settings in the table.

For class-to-image experiments, we follow the training settings described in Appendix B.1. During inference, we uniformly grid-search the CFG scale with a step size of 0.1 and the guidance interval with a step size of 0.02 for all methods, and report the best FID results for AceiT and the competing methods in Tab. 3.

For text-to-image experiments, we likewise follow the training settings in Appendix B.1 and use the Euler sampler with 50 sampling steps and CFG = 4.0 for all methods in Fig. 3(c).

C Algorithm Design Details

Algorithm 1 presents the pseudocode for the depth-wise attention computation. Specifically, Depth Attention performs softmax attention over block representations using a learned query w, where the resulting attention weights correspond to the coefficients $\alpha_{i\to l}$ in Sec. 3.1, satisfying $\sum_{i=0}^{l-1} \alpha_{i\to l} = 1$.

Algorithm 2 presents the pseudocode for chunk downsampling. Specifically, we first halve the channel dimension with a linear projection, split the resulting features into two channel chunks, and apply spatial pixel-unshuffle to each chunk, thereby achieving an overall $4\times$ downsampling ratio. The downsampled chunks x_1 and x_2 are then directly appended to the block representation list.

Algorithm 1: Depth Attention

```
class DepthAttention(nn.Module):
    def forward(self,blocks,query):
        # blocks: list of block
            representations
        V = torch.stack(blocks, dim
            =0)
        K = self.norm(V)
        logits = (K * query).sum(-1)
        weights = logits.softmax(0)
        out = (weights.unsqueeze(-1)
             * V).sum(0)

        return out
```

Algorithm 2: Chunk Downsampling

```
class ChunkDownsample4x(nn.Module):
    def forward(self,x,blocks):
        # blocks: list of block
            representations
        x = self.proj(x)
        x1, x2 = torch.chunk(x, 2,
            dim=-1)
        x1 = self.pixel_unshuffle(x1
            )
        x2 = self.pixel_unshuffle(x2
            )
        blocks.append(x1)
        blocks.append(x2)
```

D Experiment Design Details

D.1 Attention Entropy

Layer-wise attention entropy in Sec. 3.2 measures the concentration of the attention distribution within each layer, characterizing different attention patterns (Zhai et al., 2023). It is computed as

$$H^{(l)} = -\frac{1}{|N^{(l)}|} \sum_{i \in N^{(l)}} \sum_{j \in N^{(l)}} p_{ij}^{(l)} \log p_{ij}^{(l)}, \tag{6}$$

where $p_{ij}^{(l)}$ denotes the attention weight from token i to token j in the l-th layer.

To ensure the accuracy and stability of the analysis, we first generate 10,000 images on ImageNet-256 using the inference settings in Tab. 6. We then average the entropy across attention heads and timesteps, then normalize it by $\log |N^{(l)}|$ before plotting it in Fig. 2(b) to enable a fair comparison across models with different patch sizes.

Although Fig. 2(b) shows that excessively large patch sizes lead to larger pattern discrepancies and increasingly uniform attention distributions at the endpoint blocks, these observations alone do not establish that the resulting attention patterns are ineffective. We therefore further introduce token-wise spectral entropy (Passerini & Severini, 2008):

$$\bar{H}^{(l)} = -\frac{\sum_k \lambda_k \log(\lambda_k)}{\log |N^{(l)}|}, \quad \rho(A) = \frac{AA^\top}{\mathrm{Tr}(AA^\top)}, \tag{7}$$

where λ_k are the eigenvalues of $\rho(A)$ and $A \in \mathbb{R}^{|N^{(l)}| \times |N^{(l)}|}$ denotes the attention matrix at layer l. Token-wise spectral entropy measures the diversity of attention patterns across tokens within each layer. A lower spectral entropy indicates less diverse attention patterns, e.g., when many tokens concentrate their attention on only a few tokens. Following the same procedure used to compute the layer-wise attention entropy, we further plot the spectral entropy across network depth in Fig. 7. Compared with Baseline-B/8, the endpoint blocks of Baseline-B/32 exhibit substantially lower spectral entropy, indicating that increasing the patch size makes the attention patterns at the endpoints more homogeneous.

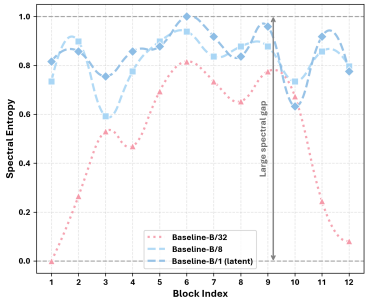


Figure 7: Spectral entropy across depth.

Again, lower spectral entropy alone does not imply that the endpoint attention patterns are ineffective. For example, when many tokens attend to a small subset of tokens, it may instead indicate that the model has identified salient distributional features. However, when higher layer-wise attention entropy is accompanied by lower token-wise spectral

entropy, the two statistics jointly suggest a degenerate attention pattern: each token distributes its attention almost uniformly across other tokens, while different tokens exhibit highly similar attention patterns. In other words, the attention weights within the layer become nearly identical. In the extreme case where the attention weights are identical across all tokens within a layer, the attention pattern becomes completely ineffective, as all tokens receive the same attention distribution. Fig. 7, together with Fig. 2(b) and (c), provides consistent evidence supporting the design of AceiT.

D.2 Fine-tuning γ

As illustrated in Sec. 3.2, γ can be used to optimize the layer-wise attention patterns. To prevent negative γ from reversing the relative ordering of the original attention logits, we parameterize γ as $\gamma = \exp(\eta)$ and initialize η to zero. Since the majority of model parameters are frozen, only a short fine-tuning stage is required. Following the training configuration in Tab. 6, we fine-tune the model for an additional epoch and report the corresponding results in Sec. 3.2 and Fig. 2(c).

D.3 Attention Residual Details

Conventional residual connections (He et al., 2016) are a core component of modern LLMs (Liu et al., 2024; Achiam et al., 2023; Touvron et al., 2023). The canonical update $h_l = h_{l-1} + f_{l-1}(h_{l-1})$ acts as a gradient highway, allowing signals to bypass nonlinear transformations via identity mappings and thus stabilizing deep training. In practice, the PreNorm formulation (Xiong et al., 2020) is widely adopted; however, its unweighted accumulation leads to an $O(L)$ growth in hidden-state magnitude with depth, which can dilute the contribution of individual layers and obscure early representations (Li et al., 2026).

Table 8: **Ablation of dynamic query designs.** We report FID under different conditioning strategies, where t denotes the timestep and c denotes class conditioning.

	c	t	$t+c$
AdaLN	11.62	**8.66**	10.17
Add	12.96	9.12	10.59

Motivated by this perspective, Attention Residual (Team et al., 2026) replaces uniform summation with a weighted aggregation: $h_l = \sum_i \alpha_{i\to l}\, v_i$, where $\alpha_{i\to l}$ are softmax attention weights computed using a learned query for each layer. For this query, there are two possible designs. The first uses a static learnable parameter, which captures the overall data distribution without adapting to external conditions such as the timestep or text input. The second modulates the query with external conditions, allowing it to dynamically adapt to the input conditions. In LLMs, such dynamic query modulation has been shown to provide certain performance gains.

To investigate whether this design benefits pixel-space DiTs, we first evaluate the original dynamic design, which uses a $C \times C$ projection (where C denotes the feature dimension) at each layer to derive the query from the hidden state. This modification degrades the FID from 4.80 to 9.07. We then explore two modulation strategies more widely adopted in DiTs under three combinations of conditioning signals, as shown in Tab. 8. *AdaLN* transforms the external conditioning signal into modulation parameters through a linear layer, while *Add* directly adds the conditioning signal to the query. Tab. 1 uses the best FID reported in Tab. 8 as the reference for evaluating the effectiveness of Dynamic Query.

Finally, for the block attention residuals in Tab. 1, we treat two attention layers and two FFN layers as a single attention residual block.

E Quality Comparison Details

Fig. 5 in the main paper compares AceiT-L/64 and PixelDiT-L/64 after training for 40K iterations on the tuning dataset, providing a clearer illustration of their convergence speed. For both models, we use the Euler sampler with 50 sampling steps and a CFG scale of 4.0.

We provide the text prompts corresponding to Fig. 5 in the main paper:

- *A large gray rock rests in a forest, surrounded by fallen leaves and logs.*
- *A small island lies in the middle of a calm blue lake, with mountains surrounding it and reflected in the water.*

- *Two men wearing suits stand in a formal room, with one in the foreground and the other partially visible behind him.*
- *A fancy car with alternating purple and dark green colors drives along a city street.*
- *A yellowish mushroom with a cracked cap grows on a fallen log, surrounded by dry leaves and twigs in a forest.*
- *A person rides a scooter along a wet street at sunset, with golden light illuminating the sky behind the buildings.*
- *A long-haired man plays a mandolin on stage under bright lights.*

We further show the visualization results of fully trained AceiT-H/64 and the results on ImageNet-256, corresponding to Tab. 5 and Tab. 3 in the main paper, respectively, in Fig. 8 and Fig. 9.

We provide the text prompts corresponding to Fig. 8 in the appendix:

- *A tiny vintage train slowly passing through a quiet autumn forest, surrounded by golden and crimson leaves.*
- *A glass greenhouse hidden in a quiet forest during a gentle summer rain, lush tropical plants and delicate flowers inside.*
- *A small wooden cabin standing beside a perfectly still mountain lake at dawn, surrounded by pale morning mist and tall pine trees.*
- *An abandoned rural railway station completely surrounded by blooming cherry trees, an old wooden platform covered with fallen petals.*
- *A space cowboy sitting inside a futuristic spaceship, holding a cup of coffee, quietly looking out of a large window into deep space.*
- *A transparent glass flower pot containing a miniature city instead of plants, tall modern skyscrapers growing like plants inside the pot.*
- *Van Gogh inspired oil painting, early dawn sky with swirling brush strokes, golden sunrise blending into cool blue sky.*
- *A miniature futuristic city growing inside a vast botanical garden, tiny skyscrapers surrounded by enormous flowers and moss, miniature streets illuminated by warm lights, small bridges crossing narrow streams.*
- *An old rusted revolver partially buried in soil, surrounded and covered by blooming wild flowers.*
- *A majestic snow-capped mountain reflected perfectly in a calm, mirror-like lake, crystal clear water surface with flawless symmetry.*
- *An oil-painting style portrait of a human growing colorful flowers from her hair.*
- *A quiet rooftop garden in a modern city immediately after rain, wet concrete reflecting soft evening lights, small trees and wild grasses growing among minimalist architecture.*
- *A crystal-clear blue river winding through an ancient forest, smooth stones visible beneath the water.*
- *A single wooden bench overlooking a calm ocean from the edge of a grassy cliff, wildflowers moving gently in the sea breeze, pastel sunset spreading across the horizon, a few distant seabirds.*
- *An open book on a table, and a whole miniature world is growing out of the pages, forests, rivers, and cities emerging from paper.*
- *A solitary wooden cabin standing on a perfectly still lake, surrounded by pale morning mist, a narrow wooden walkway disappearing into the fog.*
- *A crystal wine glass filled with pink floating clouds, the clouds flow and swirl inside the glass.*
- *A dimly lit room glows softly in the quiet of night. On the table, a glass bottle holds a swirling miniature Milky Way, while beyond the window, a vast star-filled sky stretches into the distance.*
- *A glowing crystal sphere containing a small enchanted house, surrounded by floating particles and soft mist.*
- *Stars gently falling from the sky into a quiet countryside field, glowing grass and fireflies hand-painted animation style.*

- *A tracked vehicle with license plate "Apex-128" sits in the snow beside a wooden house.*
- *A cozy modern bookstore at sunset, with a large elegant sign above the entrance that clearly reads "SISYPHE".*
- *A white robe stands on a blue mountain, reflected in calm water below. The text "HEAVEN IS NEAR" appears at the top and bottom of the image. Mountains form the background.*
- *A close-up of a brown paper bag with black text listing food items. The words "SOUR CREAM & CHILES" and "SMOKY" are visible, along with phrases "BLACK OR PINTO".*

F Training and Inference Efficiency Details

The models reported in Tab. 6 and Tab. 7 are trained on either $8\times$ NVIDIA A100 GPUs (80GB) or $8\times$ NVIDIA H200 GPUs. In our experiments, $8\times$ NVIDIA A100 GPUs are sufficient to train AceiT-H/64, a 2.1B-parameter model, with a global batch size of 32, and AceiT-L/64 with a global batch size of 256. AceiT-G and PixelDiT-L (T2I) are trained on $8\times$ NVIDIA H200 GPUs, with the corresponding results shown in Fig. 3(c). When the global batch size is smaller than 256, we employ gradient accumulation to maintain an effective batch size of 256. Fully training AceiT-L/64 and AceiT-H/64 takes only approximately 2 and 3 weeks, respectively.

The latency (s) and throughput (samples/s) reported in Tabs. 1 and 5 are measured on a single NVIDIA A100 GPU using the recommended inference settings of each method. For AceiT-H/64, we use the Euler sampler with 50 sampling steps.

G Limitations

Although pixel-space DiTs eliminate the reliance on VAEs and thereby avoid the information loss introduced by VAE compression, we observe that they still exhibit limited text-rendering capabilities in our experiments. The last row of Fig. 8 presents several failure cases, including missing, incorrect, blurry, and extraneous text. This suggests that pixel-space DiTs may still benefit from post-training specifically targeted at improving text-rendering capabilities.

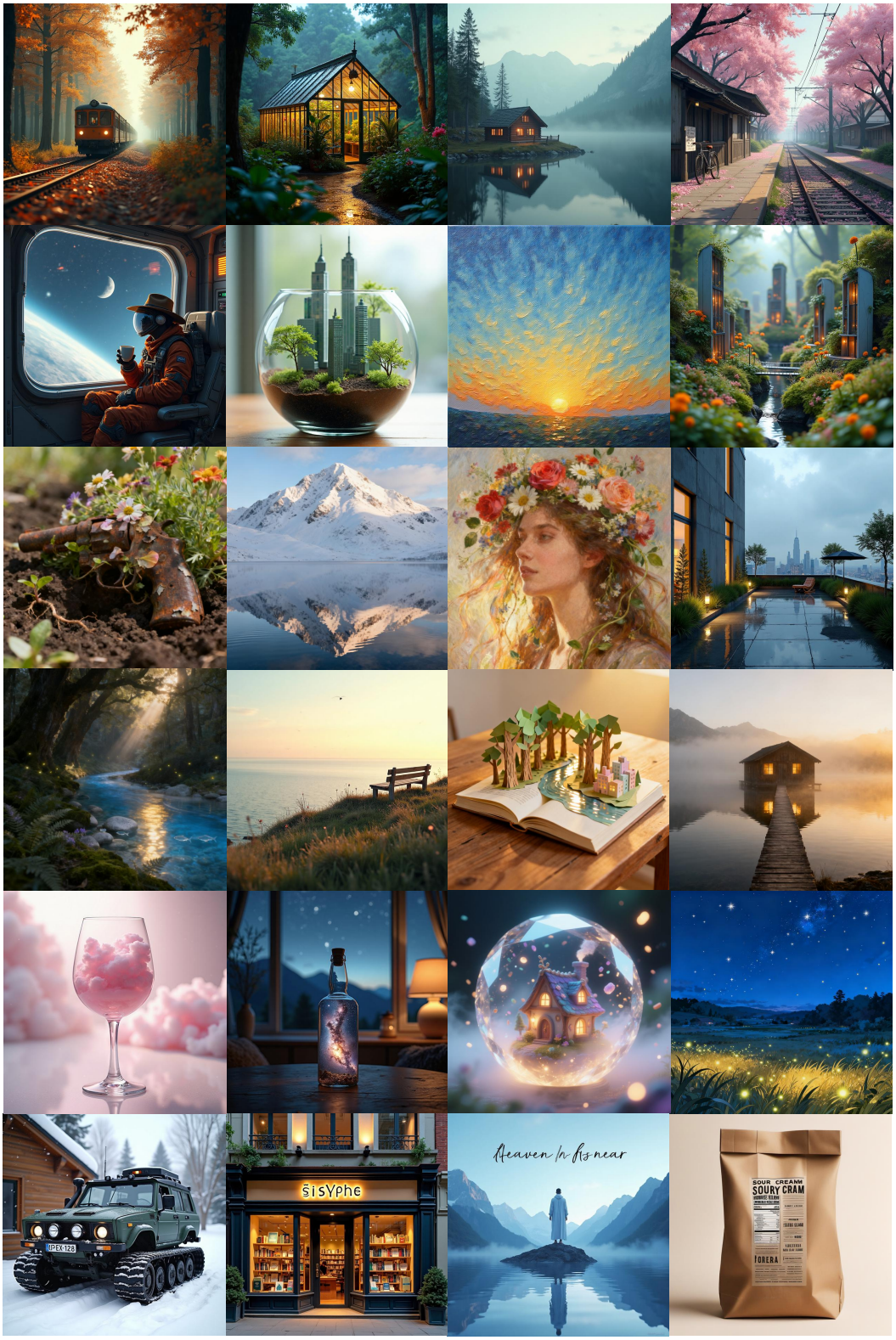


Figure 8: **More qualitative results of AceiT-H/64 at the native** 1024^2 **resolution.** See Appendix E for prompts and detailed configurations.

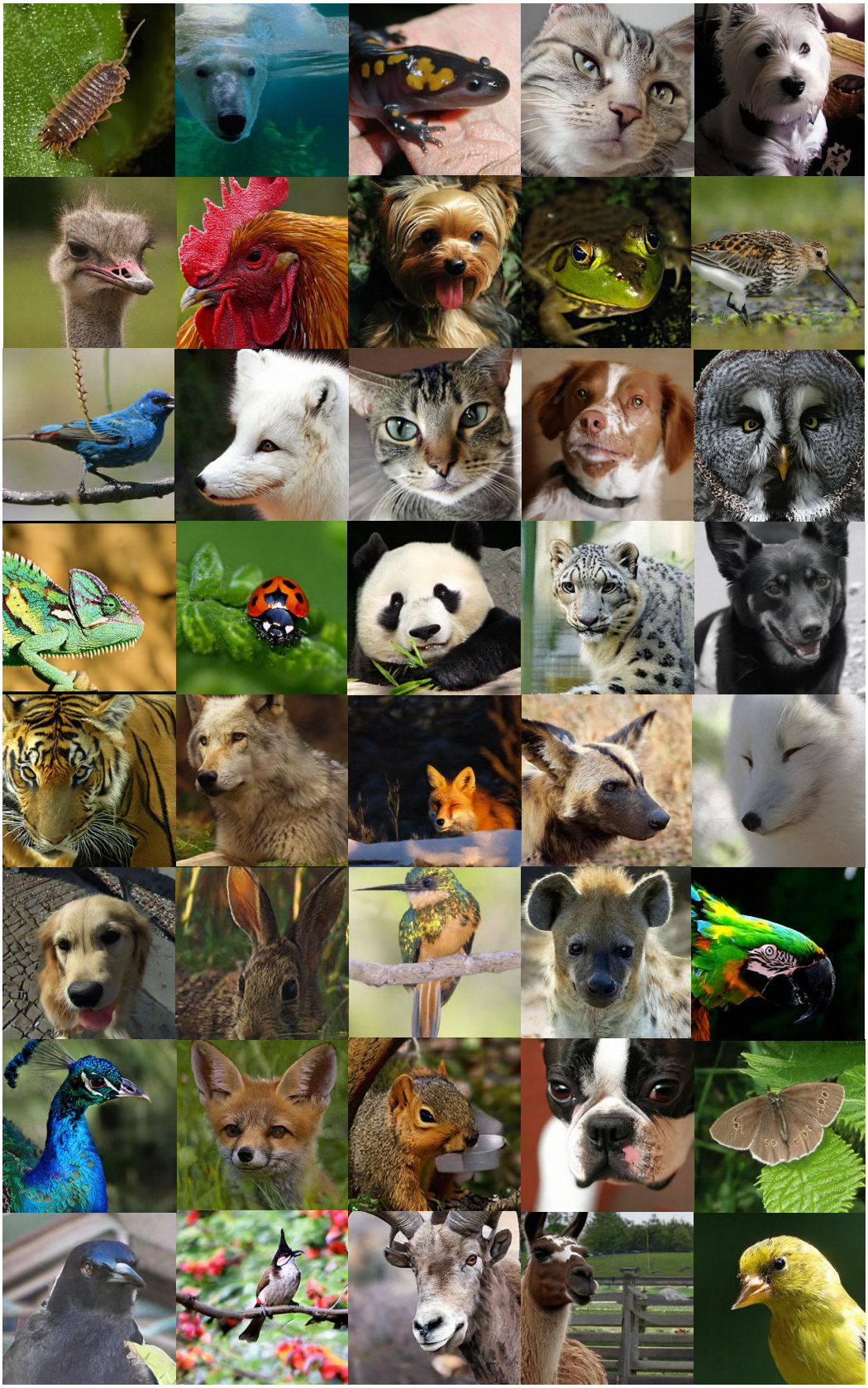


Figure 9: **Qualitative results on ImageNet-256.** Settings: AceiT-L/16, 120 epochs, CFG = 3.25