HierarchicalPrune: Position-Aware Compression for Large-Scale Diffusion Models

Young D. Kwon[1*] **Rui Li**[1*]
Sijia Li[2] **Da Li**[1] **Sourav Bhattacharya**[1] **Stylianos I. Venieris**[1]
[1] Samsung AI Center-Cambridge [2]Independent Researcher
{yd.kwon, rui.li, s.venieris}@samsung.com

arXiv:2508.04663v1 [cs.CV] 6 Aug 2025

Abstract

State-of-the-art text-to-image diffusion models (DMs) achieve remarkable quality, yet their massive parameter scale (8-11B) poses significant challenges for inferences on resource-constrained devices. In this paper, we present *HierarchicalPrune*, a novel compression framework grounded in a key observation: DM blocks exhibit distinct functional hierarchies, where early blocks establish semantic structures while later blocks handle texture refinements. *HierarchicalPrune* synergistically combines three techniques: (1) Hierarchical Position Pruning, which identifies and removes less essential later blocks based on position hierarchy; (2) Positional Weight Preservation, which systematically protects early model portions that are essential for semantic structural integrity; and (3) Sensitivity-Guided Distillation, which adjusts knowledge-transfer intensity based on our discovery of block-wise sensitivity variations. As a result, our framework brings billion-scale diffusion models into a range more suitable for on-device inference, while preserving the quality of the output images. Specifically, when combined with INT4 weight quantisation, *HierarchicalPrune* achieves 77.5-80.4% memory footprint reduction (*e.g.*, from 15.8 GB to 3.2 GB) and 27.9-38.0% latency reduction, measured on server and consumer grade GPUs, with the minimum drop of 2.6% in GenEval score and 7% in HPSv2 score compared to the original model. Last but not least, our comprehensive user study with 85 participants demonstrates that *HierarchicalPrune* maintains perceptual quality comparable to the original model while significantly outperforming prior works.

1 Introduction

Diffusion-based text-to-image (T2I) synthesis (Song et al. 2021; Stability AI 2023; Lin, Wang, and Yang 2024; Lipman et al. 2023; Karras et al. 2022; Peebles and Xie 2022; Lu et al. 2024) has emerged as a powerful tool for a wide variety of applications, such as the generation of educational content, creative artwork, and UX/UI design prototyping, intensifying demand for deployable billion-parameter diffusion models (DMs). While recent advances, such as Stable Diffusion 3.5 (SD3.5) (Esser et al. 2024) and FLUX (Black Forest Labs 2024), significantly outperform previous generations (SDXL (Podell et al. 2023), SD1.5 (Rombach et al. 2022), and DALLE-2) in image quality and text alignment,

*Co-first authors

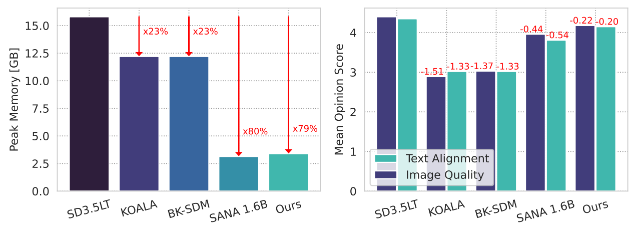


Figure 1: *HierarchicalPrune* achieves 79.5% memory reduction (left) while maintaining image quality. User study (right) demonstrates minimal quality drop (4.8-5.3%), contrary to the excessive degradation (11.1-52.2%) of prior methods.

they also come with excessive model sizes (8-11B parameters) and compute demands, limiting the accessibility of such advanced models.

At the same time, despite quantitative metrics (*e.g.*, GenEval (Ghosh, Hajishirzi, and Schmidt 2023) and DPG-Bench (Hu et al. 2024b)) suggesting smaller models (approx. 2B parameters) like SANA-Sprint (Chen et al. 2025b) perform better than their large-scale counterparts, user-based evaluations on the Artificial Analysis Leaderboard[1] reveal a significant gap in the *perceived quality* between compact models and their larger counterparts, which quantitative metrics fail to capture. As such, there is a longstanding need for deploying large models even in resource-constrained settings in order to provide users with high-quality T2I capabilities.

Nonetheless, this endeavour constitutes a very challenging task due to the high memory and compute intensity (Li et al. 2025) of large DMs, thereby often requiring cloud solutions equipped with high-end GPUs and a minimum of 80GB VRAM. Concurrently, most of the renowned recent releases of foundational DMs, including SD3.5 (Esser et al. 2024), FLUX (Black Forest Labs 2024), and Seedream 3.0 (ByteDance 2025), are built upon multi-modal diffusion Transformer (MMDiT) backbones (Esser et al. 2024) instead of U-Net, revealing new compression opportunities (Section 2.1). At the moment, existing efforts to improve efficiency face important limitations. **Firstly,** *sampling step*

[1]https://artificialanalysis.ai/text-to-image/arena?tab=leaderboard-text

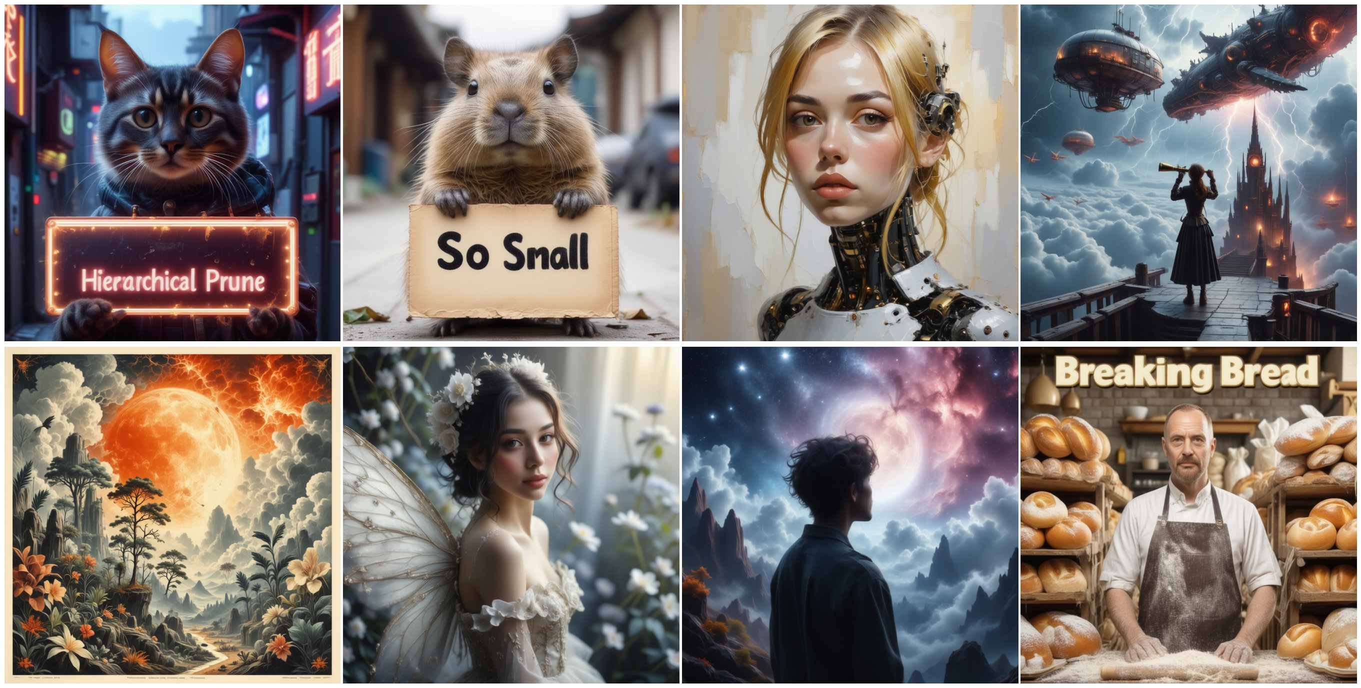


Figure 2: High-resolution image samples generated by our pruned/distilled model using *HierarchicalPrune*, showcasing its superior visual quality across various visual styles, precisely following text prompts, and preserving the ability to draw typography.

reduction (Li et al. 2023; Stability AI 2023; Sauer et al. 2023) and *efficient operator design* (Xie et al. 2025; Dao et al. 2022) are tailored to improving the speed of DMs rather than reducing memory requirements, leaving the important task of DM deployment in memory-constrained devices unresolved. **Secondly,** existing depth-pruning methods (Lee et al. 2024; Kim et al. 2024a; Fang et al. 2024) show promising results in both memory and computation reduction (Kim et al. 2024a; Fang et al. 2024), outperforming width pruning (Fang, Ma, and Wang 2023; Castells et al. 2024a), but face critical scalability challenges. While they achieve reasonable compression on U-Net-based small DMs (2.6B or less), they fail to compress large-scale, state-of-the-art (SOTA) DMs such as SD3.5 Large (8B) and FLUX (11B) (experiencing significant degradation at 20-30% memory reduction as demonstrated in Table 1). The full-block-removal methods employed therein cannot capture the fine-grained impact of the subcomponents within each block, and more importantly, the different roles of blocks in different positions across the network's hierarchy, leading to excessive performance drop at high pruning ratios. **Lastly,** orthogonal to pruning, *reduced-precision computation* (He et al. 2023; Li et al. 2025; Wang et al. 2024) has been proven effective, but it has not been combined with prior block-removal DM compression methods.

By observing the limitations of prior work, this paper identifies a novel insight for DMs that consist of MMDiT blocks: Such DMs form a two-fold hierarchy, spanning the inter- and intra-block levels. Inter-block hierarchy reflects the contribution of different blocks to disparate aspects of the output image (*e.g.* semantic structure, finer visual details), which is determined by their position in the overall architecture. Intra-block hierarchy highlights the subcomponents that compose each MMDiT block and their diverse patterns of importance to the overall quality.

Building on this new viewpoint, we propose ***HierarchicalPrune***, a principled compression methodology that overcomes the limitations of existing methods using three hierarchy-informed techniques (Fig. 3). First, we introduce **Hierarchical Position Pruning (HPP)**, a method that leverages our empirical insight that later MMDiT blocks contribute less to fundamental image structure, by strategically maintaining early blocks that form core image structures while pruning later blocks that primarily handle refinements. Second, we incorporate **Positional Weight Preservation (PWP)**, which freezes the non-pruned and earlier portions of the model during the distillation process. This approach maintains the integrity of early blocks, which are essential for image formation, while allowing later, less critical blocks to be updated. Finally, we propose **Sensitivity-Guided Distillation (SGDistill)**, which operates with a counterintuitive yet effective principle: blocks with higher importance are also more sensitive to change. Our analysis reveals that *in aggressive pruning settings, attempting to update these highly important blocks often proves detrimental to model performance.* As such, we enforce inverse distillation weights—assigning minimal or zero update weights to the most important blocks, while concentrating updates on less sensitive components.

We extensively benchmark *HierarchicalPrune* on both server and desktop-grade GPUs, demonstrating superior results over SOTA methods. Our contributions include:

- We identify a dual hierarchical structure in MMDiT DMs: an inter-block hierarchy (earlier blocks establish seman-

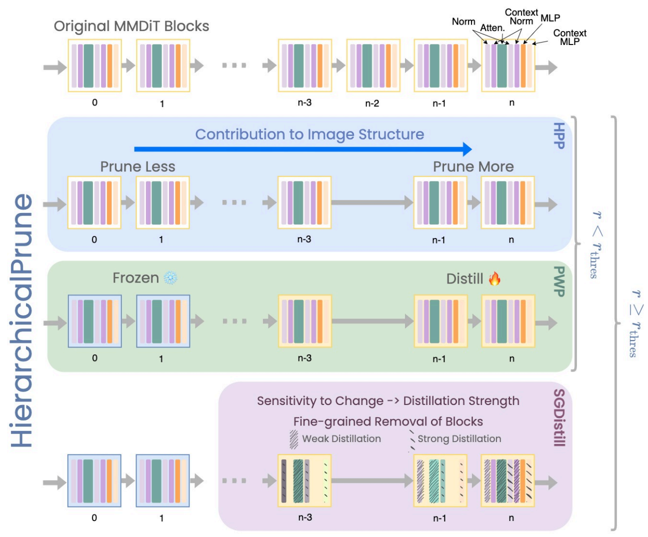


Figure 3: *HierarchicalPrune*'s compression framework leverages MMDiT's two-fold hierarchy (*inter-block*: early blocks establish semantics, later blocks refine; *intra-block*: varying subcomponent importance). It comprises (1) Hierarchical Position Pruning (HPP), maintaining early blocks while pruning later ones, (2) Positional Weight Preservation (PWP), freezing critical early blocks during distillation, and (3) Sensitivity-Guided Distillation (SGDistill), applying inverse weights—minimal updates to sensitive blocks and subcomponents. The resulting framework enables effective compression while preserving model capabilities.

tics, later blocks handle refinements) and an intra-block hierarchy (varying importance patterns of subcomponents within each MMDiT block).

- *HierarchicalPrune* establishes a comprehensive, position-aware pruning and distillation framework for large-scale DMs, for the first time, by combining HPP, PWP, and SGDistill with INT4 quantisation that achieves 77.5-80.4% memory reduction with minimal quality loss (3.2-4.8% ours vs. 15.3-41.2% degradation for prior works).
- Through an extensive user study with 85 participants, we demonstrate that *HierarchicalPrune* significantly outperforms all the baselines: KOALA and BK-SDM shows a substantial 44.0-52.2% user-perceived quality degradation, a SOTA small-scale DM (SANA-Sprint-1.6B) show a 11.1-14.2% drop. In contrast *HierarchicalPrune* shows a mere 4.8-5.3% degradation (Fig. 1).

2 Methodology

2.1 Motivation

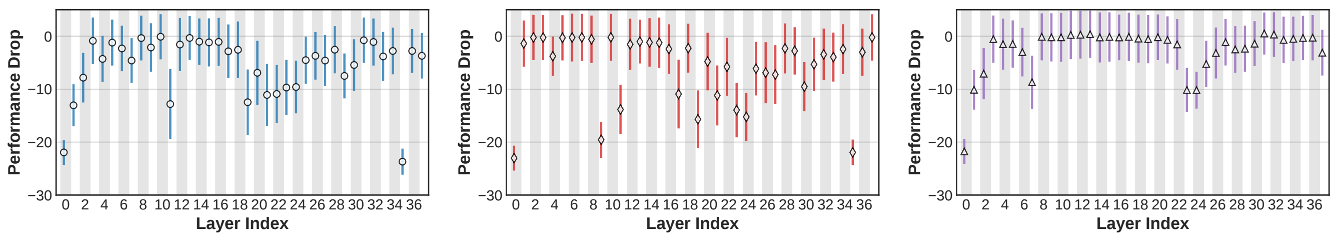
Examining the architecture and performance of MMDiT-based DMs, we identify a two-fold hierarchy: *i) inter-block* and *ii) intra-block hierarchy*. Inter-block hierarchy refers to the organisation of blocks along the overall architecture and implies a functional hierarchy where blocks have different responsibilities based on their position. Intra-block hierarchy focuses on the fact that individual blocks are internally composed of subcomponents, which in the case of MMDiT consist of Norm, Context Norm, Attention, MLP, and Context MLP modules (Esser et al. 2024), with each subcomponent having a varying impact on performance based on its type and position.

Given this dual hierarchy, we conjectured that each block is responsible for different aspects of the generated images and that its position in the DM architecture largely determines these aspects. This, in turn, holds for subcomponents but is also affected by their type. Concretely, we hypothesised that the contribution of different blocks to the output image is not uniform, both in terms of importance to overall performance and influence on specific image traits (*e.g.*, structure, texture or finer details).

To investigate this, we first conducted a contribution analysis on SD3.5 Large Turbo over the HPSv2 dataset (Wu et al. 2023b), by removing both individual MMDiT blocks (Fig. 4a) and subcomponents (Fig. 4b and 4c in this section, and Fig. 8b–8e in Appendix B), and comparing the performance before and after. We observe that each of the subcomponents, as well as whole-block removal, demonstrates *different patterns of impact at different locations*. This was further highlighted when analysing the joint removal of multiple subcomponent types, where different subcomponent combinations exhibited significantly different effects on the final performance (see Appendix B).

We further observe that the performance drop is typically more severe in earlier stages of the network. Given the inter-block hierarchy hypothesis, we attribute this to earlier blocks contributing more to core elements of the output image that affect its quality, such as semantic structure, whereas later blocks are more important for finer visual details. To verify this, we examined the generated images when removing a number of MMDiT blocks at different locations throughout the network. Specifically, we randomly selected a few examples from the HPSv2 dataset and performed T2I generation with SD3.5 Large Turbo, removing three non-consecutive blocks each time. Fig. 5 shows that the overall image structure changes dramatically when layers are removed before and up to layer 10, while after layer 30, removal of the same number of blocks has minimal impact on the structure, while finer details, such as style, are still affected, providing evidence for the inter-block hierarchy of MMDiT DMs.

Despite this variability, existing full-block removal methods (Lee et al. 2024; Kim et al. 2024a) treat MMDiT blocks homogeneously, compromising performance when targeting high compression ratios. Our observations reveal a critical shortcoming: by ignoring the inter-block differences, these methods inadvertently prune blocks that are disproportionately important to visual quality while retaining less impactful ones. Moreover, by coarsely removing whole blocks, existing methods not only discard redundant layers but also eliminate subcomponents that might be essential for capturing fine-grain features. As shown in Section 3.2, this lack of differentiation leads to a steep drop in model performance under aggressive pruning rates. These insights motivate us to design hierarchy-informed techniques for the effective compression of large MMDiT-based DMs.



(a) Block Removal (b) Multi-modal Attention (c) MLP

Figure 4: Fine-grained contribution analysis of SD3.5 Large Turbo on the HPSv2 dataset by removing either an entire MMDiT block (a), following prior depth pruning approaches (Lee et al. 2024; Kim et al. 2024a; Fang et al. 2024), or an intra-block subcomponent (b, c and see Fig. 8 in Appendix B for full set of analysis). We report the performance drop compared to the original model. The discrepancy in performance drop patterns reveals the different patterns of importance of each subcomponent.

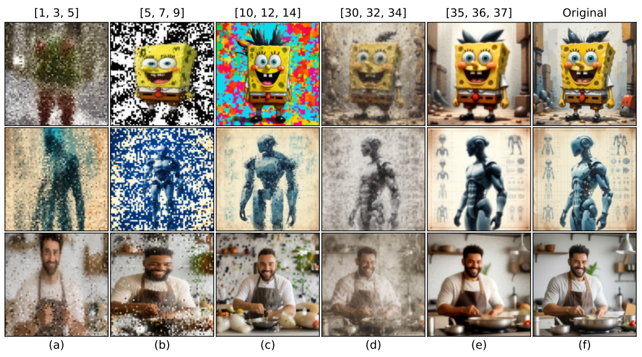


Figure 5: Impact of removing MMDiT blocks at different positions. Compared to original output (f), removing earlier layers leads to high impact on image structure (a-c), whereas removing later blocks affects mainly fine details (d, e).

2.2 HierarchicalPrune

Motivated by our findings in Section 2.1, we propose *HierarchicalPrune* (Fig. 3), a cohesive, multi-stage pruning and distillation approach that respects the hierarchical nature of DMs. *HierarchicalPrune* introduces three key techniques: *i)* Hierarchical Position Pruning (HPP), *ii)* Positional Weight Preservation (PWP), and *iii)* Sensitivity-Guided Distillation (SGDistill). These techniques operate complementarily with the objective to maximise the model compression rate while maintaining output quality throughout the T2I process. Algorithm 1 presents the complete methodology. Given a pre-trained model m, the first stage (lines 1-3) is responsible for producing a pruned model m_{pruned} with a compression ratio r using HPP (lines 1-3).

In the second stage (lines 4-7), *HierarchicalPrune* first prepares the pruned model for distillation, by preserving its most sensitive blocks by means of PWP (line 4), yielding the selectively frozen model $m_{\text{pruned,frz}}$. Then, distillation is applied in order to improve the attainable image quality. Specifically, if the target compression ratio r indicates aggressive compression by surpassing a threshold r_{thres} (line 5), *HierarchicalPrune* applies SGDistill (line 6), our proposed sensitivity-guided distillation method. In the case of moderate compression, a simpler distillation process is followed (line 8). As a final step, the resulting model is optionally quantised with 4-bit weights and 16-bit activations (line 9), to achieve further compression gains.

Algorithm 1: HierarchicalPrune

```
Input: DM $m$ with $|\mathcal{B}|$ MMDiT blocks and parameters $\boldsymbol{\theta}$
       Subcomponent set $\mathcal{C}$ of each MMDiT block
       Quality threshold $q$, Compression ratio $r$
       Dataset $\mathcal{D}$ and calibration set $\mathcal{D}_{\text{calib}} \subset \mathcal{D}$
Output: Pruned and distilled DM $m'$ with parameters $\boldsymbol{\theta'}$
   /* - - - Stage 1 - HPP / PWP - - - */
1  $\Delta P(i,c) \leftarrow$ ContributionAnalysis$(m, \mathcal{D}_{\text{calib}}, i, c)$,
     $\forall i \in [0, |\mathcal{B}|-1], \forall c \in \mathcal{C}$
           ▷ Estimate importance based on performance drop
2  $Score(i,c) \leftarrow$ PrunabilityScore$(\Delta P, i, c)$
     $\forall i \in [0, |\mathcal{B}|-1], \forall c \in \mathcal{C}$
           ▷ Calculate prunability as per Eq. (1) and (2)
3  $m_{\text{pruned}}, \boldsymbol{\theta}_{\text{pruned}} \leftarrow$ ApplyHPP$(m, \boldsymbol{\theta}, Score, r)$
     ▷ Given the prunability per block, remove blocks until compression ratio
   /* - - - Stage 2 - Distillation - - - */
4  $m_{\text{pruned,frz}}, \boldsymbol{\theta}_{\text{pruned,frz}} \leftarrow$ ApplyPWP$(m_{\text{pruned}}, \boldsymbol{\theta}_{\text{pruned}})$
     ▷ Freeze the non-pruned and earlier parts of the model
5  if $r \geq r_{\text{thres}}$ then
6      $m', \boldsymbol{\theta'} \leftarrow$ SGDistill$(m'_{\text{pruned,frz}}, \boldsymbol{\theta'}_{\text{pruned,frz}}, \mathcal{D}, q)$
           ▷ If aggressive pruning, use sensitivity-guided distillation
7  else
8      $m', \boldsymbol{\theta'} \leftarrow$ Distill$(m'_{\text{pruned,frz}}, \boldsymbol{\theta'}_{\text{pruned,frz}}, \mathcal{D}, q)$
           ▷ If moderate pruning, use normal distillation
   /* - - - Stage 3 - Post-Training Quantisation - - - */
9  $m', \boldsymbol{\theta'} \leftarrow$ PTQ$(m', \boldsymbol{\theta'}$, precision=W4A16)
           ▷ Apply PTQ for further compression
10 return $m'$, $\boldsymbol{\theta'}$
```

Hierarchical Position Pruning: At its foundation, *Hierarchical Position Pruning (HPP)* leverages our insight that deeper MMDiT blocks contribute less to core visual structure. As such, HPP strategically targets blocks for removal based on their hierarchical position within the model, maintaining early blocks that form image structures while pruning deeper blocks that mainly handle visual details.

Formally, for a DM with $|\mathcal{B}|$ MMDiT blocks, we first introduce a position weight function $W_{\text{pos}}(i)$ (Eq. (2)) that captures the inter-block hierarchy by mapping the more critical earlier blocks to lower values and guiding the pruning process towards the later layers. Next, we calculate a prunability score $Score$ (Eq. (1)) for each block in $\mathcal{B}$ and subcomponent(s), consisting of the performance drop from the original model scaled by the position weight function. Concretely:

$$Score(i, c) = -|\Delta P(i, c)| \times W_{\text{pos}}(i) \quad \forall i \in [0, |\mathcal{B}|-1] \tag{1}$$

$$W_{\text{pos}}(i) = e^{(i-|\mathcal{B}|)/|\mathcal{B}|} \tag{2}$$

where i is the block index and $\mathcal{B}$ is the number of blocks, $c \in \mathcal{C}$ is the subcomponent type, $\Delta P(i, c)$ is the importance score quantified as performance drop from the original model when block i and/or subcomponents c are removed, $W_{\text{pos}}(i)$ is the position weight function that favours later layers.

Positional Weight Preservation: Building upon this position-aware foundation, we incorporate *Positional Weight Preservation (PWP)*, which freezes the non-pruned and earlier portions of the model during the distillation process. This straightforward yet effective approach maintains the integrity of early blocks which are essential for image formation, while allowing later, less critical blocks to be updated. By systematically keeping weights static based on their position in the network, PWP significantly outperforms basic position-based pruning, HPP, for moderate pruning scenarios (*e.g.*, 25% parameter reduction), as it ensures that the most structurally important parts of the model remain intact. Note that even at moderate compression levels, prior methods suffer substantial quality degradation (15.3-41.2% in Table 1), while our approach maintains near-original performance.

Sensitivity-Guided Distillation: For aggressive parameter reduction (*e.g.*, $\geq$30%), however, we discovered that even with careful pruning and preservation, excessive block-level pruning leads to unacceptable quality degradation. To address this challenge, we introduce *Sensitivity-Guided Distillation (SGDistill)*, which operates with a counterintuitive yet effective principle: blocks with higher importance are also more sensitive to change.

Our analysis reveals that *in aggressive pruning settings, attempting to update these highly important blocks often proves detrimental to model performance* (showing 31.9% average quality reduction even with PWP, see Table 3). Consequently, SGDistill applies inverse distillation weights—assigning minimal or zero update weights to the most important blocks, while concentrating updates on less sensitive components. This approach, combined with subcomponent (*e.g.*, Normalisation and MLP within an MMDiT block) pruning for the remaining portion of the DM after pruning up to r_{thres} (line 5) with HPP and PWP, extends our hierarchical approach from the block-level to the intra-block subcomponents, preserving the carefully tuned parameters of critical blocks while allowing adaptation in less sensitive regions. Our experimental results confirm this hypothesis, showing that protecting sensitive blocks from significant updates during distillation is key to maintaining quality at high compression ratios, reducing average quality degradation from 31.9% to 10.1% (Table 3).

Concretely, the objective of the distillation process of *HierarchicalPrune* leverages a combination of the feature loss, $\mathcal{L}_{\text{feat}}$, *i.e.*, a feature distillation loss, and the standard knowledge distillation (KD) loss to minimise the difference between the final output of the compressed (*i.e.*, student) model $\boldsymbol{\theta'}$ and the original (*i.e.*, teacher) model $\boldsymbol{\theta}$, leading to the overall loss $\mathcal{L} = \mathcal{L}_{\text{feat}} + \mathcal{L}_{\text{KD}}$, with $\mathcal{L}_{\text{KD}}$ expressed as:

$$\mathcal{L}_{\text{KD}} = \mathbb{E}\left[||v_{\boldsymbol{\theta'}}(x_t, t) - v_{\boldsymbol{\theta}}(x_t, t)||^2\right] \tag{3}$$

where t is the time step, and x_t is the noisy diffusion sample, started from the clean latent x_0 from the autoencoder (VAE). For the selected set of blocks to update $\mathcal{B}^* \in \mathcal{B}$ each associated with an importance score $\Delta P(i, c)$ (Eq. (1)), by denoting feature output of block i in teacher and student models as $f^{l}_{\boldsymbol{\theta}}$ and $f^{l'}_{\boldsymbol{\theta'}}$, respectively, we have:

$$\mathcal{L}_{\text{feat}} = \mathbb{E}\left[\sum_{i \in [0, |\mathcal{B}^*|-1]} ||f^{i'}_{\boldsymbol{\theta'}}(x_t, t) - f^{i}_{\boldsymbol{\theta}}(x_t, t)||^2\right] \tag{4}$$

SGDistill specifies that, for each block, we scale the final parameter update by its sensitivity, *i.e.*, $\frac{1}{\Delta P(i,c)}$, regulating in this way the rate of change of each block during distillation.

3 Evaluation

To evaluate *HierarchicalPrune* against existing methods, we conducted both quantitative and qualitative comparisons (Section 3.2). We also perform an ablation study to assess each proposed component, the impact of quantisation, and the robustness of text-drawing capability (Section 3.3).

3.1 Experimental Setup

Architectures, Datasets, and Implementation: We target SD3.5 Large Turbo (8B) and FLUX.1-Schnell (12B), two SOTA models designed to perform diffusion tasks within a small number of steps (*e.g.*, 4). Following prior work (Lee et al. 2024), we use the YE-POP dataset (HuggingFace 2024) consisting of 500K images, derived from LAION-5B (Schuhmann et al. 2022). We implemented the *HierarchicalPrune* pipeline in PyTorch, building upon SD3.5 Large Turbo and FLUX.1-Schnell model checkpoints and pipelines from `Diffusers` (von Platen et al. 2022). For 4-bit weight quantisation, we adopt `bitsandbytes` (Dettmers et al. 2022, 2023). Note that other quantisation methods, such as SVDQuant (Li et al. 2025), can also be combined with *HierarchicalPrune*, and we leave such exploration as future work. We aim to release our codebase upon acceptance of the paper.

Baselines: We compare *HierarchicalPrune* with two prior works related to depth pruning and distillation: *i)* KOALA (Lee et al. 2024), and *ii)* BK-SDM (Kim et al. 2024a). BK-SDM proposed block pruning of U-Net-based models using the CLIP score (Hessel et al. 2021), followed by distillation of the pruned model using knowledge from the original model. KOALA follows a similar method, replacing CLIP-score-based importance ranking with scores derived from each block's input-output cosine similarity. Moreover,

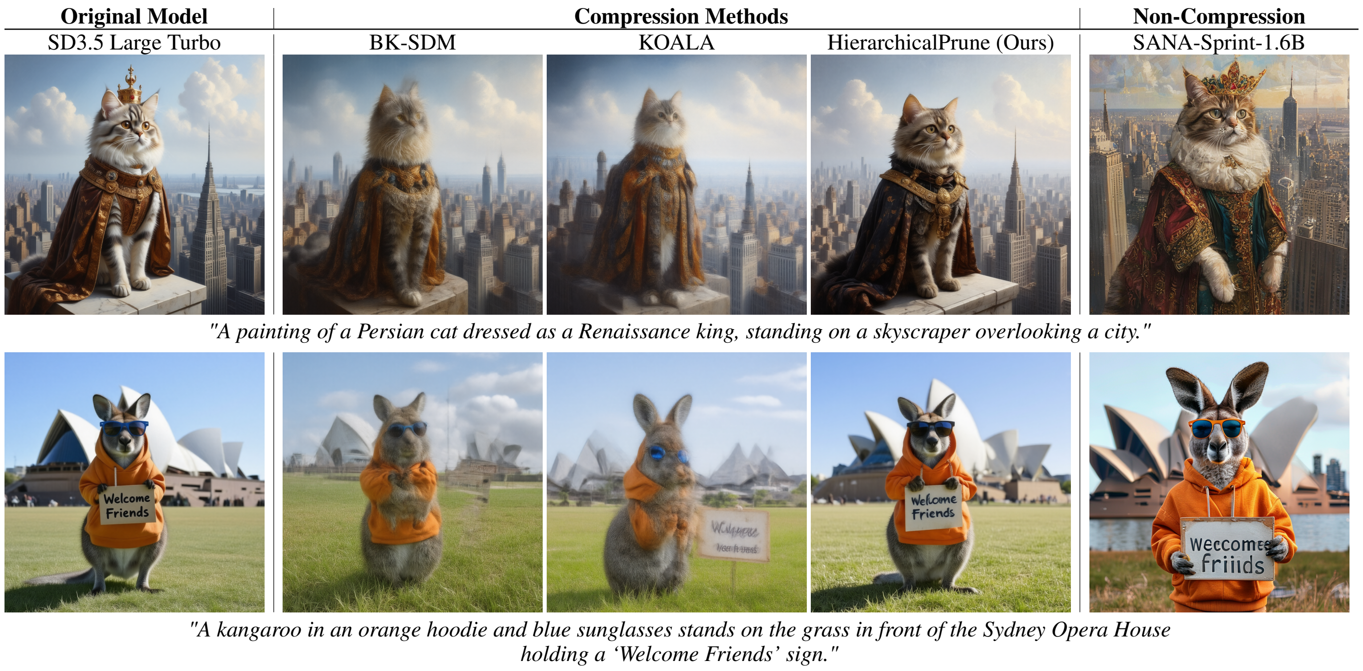

Original Model
Compression Methods
Non-Compression
SD3.5 Large Turbo
BK-SDM
KOALA
HierarchicalPrune (Ours)
SANA-Sprint-1.6B
"A painting of a Persian cat dressed as a Renaissance king, standing on a skyscraper overlooking a city."
Welcome Friends
Wecome friids
"A kangaroo in an orange hoodie and blue sunglasses stands on the grass in front of the Sydney Opera House holding a 'Welcome Friends' sign."

Figure 6: Visual comparison demonstrating the quality difference between the original model (column 1), depth pruning based on BK-SDM (column 2), KOALA (column 3), and our proposed *HierarchicalPrune* (column 4), as well as the SOTA small-scale diffusion model, SANA-Sprint-1.6B (column 5). Our approach successfully maintains visual quality while delivering 79.5% memory reduction over the original model. Notably, our method preserves the text drawing capability of the original SD3.5 Large Turbo model, where SANA-Sprint-1.6B is limited.

we employ the SOTA small-scale DM, SANA (Chen et al. 2025b) as a baseline in our evaluation.[2]

Metrics: To quantitatively evaluate all methods, we employ the two most recent and representative image quality metrics, GenEval (Ghosh, Hajishirzi, and Schmidt 2023) (used in (Xie et al. 2025; Esser et al. 2024)) and HPSv2 (Wu et al. 2023a) (used in KOALA (Lee et al. 2024)). Furthermore, we conduct a user study to assess human preferences for the generated images that are difficult to capture with the quantitative quality metrics. Similar to (Sheynin et al. 2023) and (Dai et al. 2023), we evaluate two aspects: (i) Text Alignment: How well a generated image follows the description of the text prompt, (ii) Image Quality: The overall visual quality considering clarity, colour, composition, and other factors. On the system resource side, we report the measured peak memory usage and latency of running the target models on diverse GPUs, including A6000 (Table 2), GTX 3090 and A100 GPUs (see Appendix Table 5).[3]

3.2 Main Results

Image Quality: Fig. 6 presents a qualitative comparison of example outputs. *HierarchicalPrune* shows better visual outputs, outperforming all the baselines, and maintaining high fidelity to the original DMs, both in structure and fine details (see more examples in Fig. 2 and Appendix C).

Table 1: Quantitative evaluation of image quality. KOALA and BK-SDM experience substantial degradation of image quality when reducing memory usage by 20-30%. *HierarchicalPrune* achieves significantly reduced memory usage while maintaining the image quality close to the original models.

Model	Method	Memory (%)	GenEval ⇑	HPSv2 ⇑	Reduction ⇓
Linear DiT	SANA-Sprint	(100%)	0.77	29.61	-
SD3.5 Large Turbo	Original	(100%)	0.71	30.29	-
	KOALA	(79.4%)	0.37	19.99	38.5%
	BK-SDM	(79.4%)	0.38	21.21	41.2%
	Ours (HPP, PWP, Quant)	(22.5%)	**0.69**	**28.15**	**4.8%**
	Ours (All)	(**20.5%**)	0.62	26.29	13.3%
FLUX.1 Schnell	Original	(100%)	0.66	29.71	-
	KOALA	(70.5%)	0.56	24.04	17.1%
	BK-SDM	(70.5%)	0.57	24.67	15.3%
	Ours (All)	(**19.6%**)	**0.64**	**28.69**	**3.2%**

Table 1 shows that *HierarchicalPrune* achieves comparable quality to the original model based on both HPSv2 and GenEval. While baselines induce substantial quality degradation of 38.5-41.2%, our approach yields only a minimal drop (∼3.2-4.8%), showcasing the effectiveness of our hierarchical compression strategy.

User Study: Our user study with 85 participants reaffirms the effectiveness of *HierarchicalPrune*. Fig. 1 presents the mean opinion score (MOS) across all methods. *HierarchicalPrune* achieves remarkably close MoS to the original SD3.5 Large Turbo model with only minimal reduction (4.8% for text alignment, 5.3% for image quality). In contrast, SANA-Sprint-1.6B shows a noticeable quality drop (14.2% for text alignment, 11.1% for image quality) compared to the original

[2]We would like to emphasise that SANA (Chen et al. 2025b) is a specialised on-device model requiring training *from scratch*, instead of a compression method, such as KOALA, BK-SDM, and *HierarchicalPrune*.

[3]Computational overhead analysis of the distillation process as well as contribution analysis is provided in Appendix C.

Table 2: Comparison of the peak memory during inference and per-step latency of the DM Transformer measured on an A6000 GPU with 48 GB of VRAM.

Model	**Method**	**Memory** ⇓	**Reduction** ⇑	**Latency** ⇓	**Reduction** ⇑
Linear DiT	SANA-Sprint	3.14 GB	-	54ms	-
SD3.5 Large Turbo	Original	15.8 GB	-	823 ms	-
	KOALA	12.6 GB	20.6%	642 ms	22.0%
	BK-SDM	12.6 GB	20.6%	642 ms	22.0%
	Ours (All)	**3.24 GB**	**79.5%**	**593 ms**	**27.9%**
FLUX.1 Schnell	Original	22.6 GB	-	756 ms	-
	KOALA	15.9 GB	29.5%	**432 ms**	**42.9%**
	BK-SDM	15.9 GB	29.5%	**432 ms**	**42.9%**
	Ours (All)	**4.44 GB**	**80.4%**	469 ms	38.0%

model, while other baselines (*i.e.*, BK-SDM, KOALA) show a substantial 44.0-52.2% degradation.

Peak Memory & Latency: On the resource front, as in Table 2, our compression pipeline achieves 79.5% lower peak memory (from 15.8 GB to just 3.24 GB) and 27.9% faster latency compared to the original SD3.5 Large Turbo model. Measurements of memory and latency were conducted on both server- and desktop-grade GPUs with different specifications (A100, A6000, and GTX 3090 with 80, 48, and 25 GB VRAM, respectively, reported in Appendix Table 5). While we report the measurement results from A6000, the reduction rates are similar across GPUs. Notably, the peak memory of our compressed SD3.5 model (3.24 GB) is comparable to SANA-Sprint (3.14 GB), but with substantially better image quality. Similarly, for FLUX.1-Schnell, *HierarchicalPrune* achieves 80.4% peak memory reduction and 38.0% speed gain, indicating its generalisability to larger DMs.

Overall, our findings reveal that respecting the hierarchical sensitivity of diffusion model components enables more effective model compression than approaches that treat all blocks as equally important, establishing a new paradigm for the efficient deployment of large-scale generative models.

3.3 Ablation Study and Analysis

Impact of Each Component of *HierarchicalPrune*: We conducted an ablation study of *HierarchicalPrune* to investigate the contribution of each component: (1) HPP, (2) PWP, (3) HPP+PWP+SGDistill, and (4) our final form, HPP+PWP+SGDistill+Quant. Table 3 shows that using HPP+PWP drastically improves image quality compared to HPP only and prior works such as KOALA and BK-SDM. Moreover, SGDistill substantially prevents the image generation quality degradation compared to HPP+PWP with an aggressive paramter reduction rate of 30%, leading to higher memory reduction without compromising performance drastically. *Our method - leveraging all three components - outperforms all the baselines with superior image quality, lower memory, and faster execution.*

Impact of Quantisation: We investigate how much W4A16 quantisation in our pipeline affects the final image quality. Table 3 shows the quality metrics with and without quantisation applied. The impact of quantisation is as small as 2.4-3.5% in both GenEval and HPSv2 scores.

Robustness of Text Generation: As shown in Fig. 6, prior works (BK-SDM and KOALA in pruning) and small-scale DMs (SANA-Sprint-1.6B) are limited in synthesising legible texts in the generated images. However, *HierarchicalPrune* demonstrates the superior quality in text generation in Fig. 6 and 2. *These results represent the effectiveness of our proposed method in preserving the innate capability of the original model, not only in the aesthetic quality but also in other aspects like text generation.*

Table 3: Ablation study of each component and quantisation in *HierarchicalPrune* on SD3.5 Large Turbo.

Method	**Memory (%)**	**GenEval** ⇑	**HPSv2** ⇑	**Avg. Reduction** ⇓
Original	(100%)	0.71	30.29	-
Ours (HPP)	(79.4%)	0.03	11.08	79.4%
Ours (+PWP)	(79.4%)	**0.71**	**29.97**	**2.5**%
Ours (+Quant)	(**22.5**%)	0.69	28.15	4.8%
Ours (HPP)	(71.5%)	0.0	7.00	88.4%
Ours (+PWP)	(71.5%)	0.46	21.74	31.9%
Ours (+SGDistill)	(71.5%)	**0.64**	**27.29**	**10.1**%
Ours (+Quant)	(**20.5**%)	0.62	26.29	13.3%

4 Related Work

T2I Diffusion Models. Since Stable Diffusion (SD) (Rombach et al. 2022), there has been rapid community adoption and iterative updates in the field. SDv1.4 and v1.5 were released in 2022, enhancing efficiency and enabling specialised fine-tuning (*e.g.*, DreamBooth). By mid-2023, Stable Diffusion XL (SDXL) significantly advanced resolution (1024×1024), text comprehension, and image quality. By the end of 2024, the adoption of MMDiT (Esser et al. 2024) as a backbone in SD3, SD3.5, and FLUX brought a significant boost in image generation quality and alignment with long text input. At the same time, the increasing parameter count, particularly in the backbone Transformer blocks, significantly improved image quality but with excessive resource demands. As SOTA models like SD3.5 scale up to as many as 8B parameters, efficient inference on resource-constrained devices becomes impractical. This underlines the significance of approaches such as our *HierarchicalPrune* framework, to enable DM deployment outside of high-end compute setups.

Compression Techniques. Traditional model compression methods, including distillation (Hinton, Vinyals, and Dean 2015), pruning (Han, Mao, and Dally 2016) (such as depth pruning via block removal (Ghiasi, Lin, and Le 2018; Kim et al. 2024b), width pruning (Kwon et al. 2024)), and quantisation (Jacob et al. 2018) have been applied to DMs (Lee et al. 2024; Kim et al. 2024a; Castells et al. 2024b; Hu et al. 2024a; Fang et al. 2024; Li et al. 2025), resulting in variable parameter efficiency gains. Specifically, (Kim et al. 2024a) and (Lee et al. 2024) explore block removal in the U-Net backbone of SD1.5 and SDXL models, respectively. More recently, (Li et al. 2025) explored the effectiveness of quantisation on DMs for memory reduction, hence our *HierarchicalPrune* employs an adjunct post-training quantisation feature as part of its design. In this paper, we investigated the hierarchical nature of the SOTA MMDiT backbone in SD3.5 and FLUX, and further leveraged these insights for the compression of SOTA multi-billion-scale DMs, under a unified framework.

5 Conclusion

In this work, we proposed *HierarchicalPrune*, pushing the limit of compressing MMDiT-based large-scale DMs through hierarchical insights. By combining HPP, PWP, and SGDistil with quantisation, *HierarchicalPrune* achieves 77.5-80.4% memory reduction and 27.9-38.0% latency improvements with minimal quality loss, bringing SOTA large-scale DMs (SD3.5 Large Turbo, FLUX.1-Schnell) within reach of resource-constrained environments and democratising access to high-quality T2I generation.

References

Black Forest Labs. 2024. Flux.1 Model Family. https://blackforestlabs.ai/announcing-black-forest-labs/.

ByteDance. 2025. Seedream 3.0 Technical Report. In *https://arxiv.org/pdf/2504.11346*.

Castells, T.; Song, H.-K.; Kim, B.-K.; and Choi, S. 2024a. LD-Pruner: Efficient Pruning of Latent Diffusion Models using Task-Agnostic Insights. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

Castells, T.; Song, H.-K.; Piao, T.; Choi, S.; Kim, B.-K.; Yim, H.; Lee, C.; Kim, J. G.; and Kim, T.-H. 2024b. EdgeFusion: On-Device Text-to-Image Generation. *CoRR*, abs/2404.11925.

Chen, J.; Cai, H.; Chen, J.; Xie, E.; Yang, S.; Tang, H.; Li, M.; and Han, S. 2025a. Deep Compression Autoencoder for Efficient High-Resolution Diffusion Models. In *International Conference on Learning Representations (ICLR)*.

Chen, J.; Xue, S.; Zhao, Y.; Yu, J.; Paul, S.; Chen, J.; Cai, H.; Xie, E.; and Han, S. 2025b. SANA-Sprint: One-Step Diffusion with Continuous-Time Consistency Distillation. arXiv:2503.09641.

Dai, X.; Hou, J.; Ma, C.-Y.; Tsai, S.; Wang, J.; Wang, R.; Zhang, P.; Vandenhende, S.; Wang, X.; Dubey, A.; Yu, M.; Kadian, A.; Radenovic, F.; Mahajan, D.; Li, K.; Zhao, Y.; Petrovic, V.; Singh, M. K.; Motwani, S.; Wen, Y.; Song, Y.; Sumbaly, R.; Ramanathan, V.; He, Z.; Vajda, P.; and Parikh, D. 2023. Emu: Enhancing Image Generation Models Using Photogenic Needles in a Haystack. arXiv:2309.15807.

Dao, T.; Fu, D.; Ermon, S.; Rudra, A.; and Ré, C. 2022. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Dettmers, T.; Lewis, M.; Belkada, Y.; and Zettlemoyer, L. 2022. GPT3.int8(): 8-bit Matrix Multiplication for Transformers at Scale. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Esser, P.; Kulal, S.; Blattmann, A.; Entezari, R.; Müller, J.; Saini, H.; Levi, Y.; Lorenz, D.; Sauer, A.; Boesel, F.; et al. 2024. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In *International Conference on Machine Learning (ICML)*.

Fang, G.; Li, K.; Ma, X.; and Wang, X. 2024. TinyFusion: Diffusion Transformers Learned Shallow. *arXiv preprint arXiv:2412.01199*.

Fang, G.; Ma, X.; and Wang, X. 2023. Structural Pruning for Diffusion Models. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*.

Ghiasi, G.; Lin, T.-Y.; and Le, Q. V. 2018. DropBlock: A Regularization Method for Convolutional Networks. *Advances in Neural Information Processing Systems (NeurIPS)*.

Ghosh, D.; Hajishirzi, H.; and Schmidt, L. 2023. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems (NeurIPS)*.

Han, S.; Mao, H.; and Dally, W. J. 2016. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *International Conference on Learning Representations (ICLR)*.

He, Y.; Liu, L.; Liu, J.; Wu, W.; Zhou, H.; and Zhuang, B. 2023. PTQD: Accurate Post-Training Quantization for Diffusion Models. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*.

Hessel, J.; Holtzman, A.; Forbes, M.; Le Bras, R.; and Choi, Y. 2021. CLIPScore: A Reference-free Evaluation Metric for Image Captioning. In Moens, M.-F.; Huang, X.; Specia, L.; and Yih, S. W.-t., eds., *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 7514–7528. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.

Hu, D.; Chen, J.; Huang, X.; Coskun, H.; Sahni, A.; Gupta, A.; Goyal, A.; Lahiri, D.; Singh, R.; Idelbayev, Y.; et al. 2024a. SnapGen: Taming High-Resolution Text-to-Image Models for Mobile Devices with Efficient Architectures and Training. *arXiv preprint arXiv:2412.09619*.

Hu, X.; Wang, R.; Fang, Y.; Fu, B.; Cheng, P.; and Yu, G. 2024b. ELLA: Equip Diffusion Models with LLM for Enhanced Semantic Alignment. arXiv:2403.05135.

HuggingFace. 2024. YE-POP Dataset. https://huggingface.co/datasets/Ejafa/ye-pop. Accessed: 2025-March.

Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; and Kalenichenko, D. 2018. Quantization and Training of Neural Networks for Efficient Integer-arithmetic-only Inference. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Karras, T.; Aittala, M.; Aila, T.; and Laine, S. 2022. Elucidating the Design Space of Diffusion-Based Generative Models. In *Advances in Neural Processing Systems (NeurIPS)*.

Katharopoulos, A.; Vyas, A.; Pappas, N.; and Fleuret, F. 2020. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *International Conference on Machine Learning (ICML)*.

Kim, B.-K.; Song, H.-K.; Castells, T.; and Choi, S. 2024a. BK-SDM: A Lightweight, Fast, and Cheap Version of Stable Diffusion. In *European Conference on Computer Vision (ECCV)*.

Kim, J.; Halabi, M. E.; Ji, M.; and Song, H. O. 2024b. LayerMerge: Neural Network Depth Compression through Layer

Pruning and Merging. In *International Conference on Machine Learning (ICML)*.

Kwon, Y. D.; Li, R.; Venieris, S. I.; Chauhan, J.; Lane, N. D.; and Mascolo, C. 2024. TinyTrain: Resource-Aware Task-Adaptive Sparse Training of DNNs at the Data-Scarce Edge. In *International Conference on Machine Learning (ICML)*.

Lee, Y.; Park, K.; Cho, Y.; Lee, Y.-J.; and Hwang, S. J. 2024. Koala: Empirical Lessons toward Memory-Efficient and Fast Diffusion Models for Text-to-Image Synthesis. *Advances in Neural Information Processing Systems (NeurIPS)*.

Li, M.; Lin, Y.; Zhang, Z.; Cai, T.; Guo, J.; Li, X.; Xie, E.; Meng, C.; Zhu, J.-Y.; and Han, S. 2025. SVDQuant: Absorbing Outliers by Low-Rank Component for 4-Bit Diffusion Models. In *International Conference on Learning Representations (ICLR)*.

Li, Y.; Wang, H.; Jin, Q.; Hu, J.; Chemerys, P.; Fu, Y.; Wang, Y.; Tulyakov, S.; and Ren, J. 2023. SnapFusion: Text-to-Image Diffusion Model on Mobile Devices within Two Seconds. In *https://arxiv.org/abs/2306.00980*.

Lin, S.; Wang, A.; and Yang, X. 2024. SDXL-Lightning: Progressive Adversarial Diffusion Distillation. In *https://arxiv.org/abs/2402.13929*.

Lipman, Y.; Chen, R. T. Q.; Ben-Hamu, H.; Nickel, M.; and Le, M. 2023. Flow Matching for Generative Modeling. In *International Conference on Learning Representations (ICLR)*.

Lu, Z.; Wang, Z.; Huang, D.; Wu, C.; Liu, X.; Ouyang, W.; and Bai, L. 2024. FiT: Flexible Vision Transformer for Diffusion Model. In *International Conference on Machine Learning (ICML)*.

Peebles, W.; and Xie, S. 2022. Scalable Diffusion Models with Transformers. In *International Conference on Computer Vision (ICCV)*.

Pernias, P.; Rampas, D.; Richter, M. L.; Pal, C. J.; and Aubreville, M. 2024. Wüerstchen: An Efficient Architecture for Large-Scale Text-to-Image Diffusion Models. In *https://openreview.net/forum?id=gU58d5QeGv*.

Podell, D.; English, Z.; Lacey, K.; Blattmann, A.; Dockhorn, T.; Müller, J.; Penna, J.; and Rombach, R. 2023. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. In *https://arxiv.org/abs/2307.01952*.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sauer, A.; Lorenz, D.; Blattmann, A.; and Rombach, R. 2023. Adversarial Diffusion Distillation. In *https://arxiv.org/abs/2311.17042*.

Schuhmann, C.; Beaumont, R.; Vencu, R.; Gordon, C.; Wightman, R.; Cherti, M.; Coombes, T.; Katta, A.; Mullis, C.; Wortsman, M.; Schramowski, P.; Kundurthy, S.; Crowson, K.; Schmidt, L.; Kaczmarczyk, R.; and Jitsev, J. 2022. LAION-5B: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems (NeurIPS)*.

Sheynin, S.; Polyak, A.; Singer, U.; Kirstain, Y.; Zohar, A.; Ashual, O.; Parikh, D.; and Taigman, Y. 2023. Emu Edit: Precise Image Editing via Recognition and Generation Tasks. arXiv:2311.10089.

Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations (ICLR)*.

Stability AI. 2023. SDXL-Turbo. https://huggingface.co/stabilityai/sdxl-turbo.

von Platen, P.; Patil, S.; Lozhkov, A.; Cuenca, P.; Lambert, N.; Rasul, K.; Davaadorj, M.; Nair, D.; Paul, S.; Berman, W.; Xu, Y.; Liu, S.; and Wolf, T. 2022. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers.

Wang, H.; Shang, Y.; Yuan, Z.; Wu, J.; and Yan, Y. 2024. QuEST: Low-bit Diffusion Model Quantization via Efficient Selective Finetuning. In *https://arxiv.org/abs/2402.03666*.

Wu, X.; Hao, Y.; Sun, K.; Chen, Y.; Zhu, F.; Zhao, R.; and Li, H. 2023a. Human Preference Score v2: A Solid Benchmark for Evaluating Human Preferences of Text-to-Image Synthesis. *arXiv preprint arXiv:2306.09341*.

Wu, X.; Sun, K.; Zhu, F.; Zhao, R.; and Li, H. 2023b. Human Preference Score: Better Aligning Text-to-Image Models with Human Preference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Xie, E.; Chen, J.; Chen, J.; Cai, H.; Tang, H.; Lin, Y.; Zhang, Z.; Li, M.; Zhu, L.; Lu, Y.; and Han, S. 2025. SANA: Efficient High-Resolution Text-to-Image Synthesis with Linear Diffusion Transformers. In *International Conference on Learning Representations (ICLR)*.

Supplementary Material
HierarchicalPrune: Position-Aware Compression for Large-Scale Diffusion Models

A Detailed Experimental Setup

A.1 Datasets

We use the YE-POP dataset from HuggingFace for distillation training of the target models, similar to prior work by (Lee et al. 2024). The YE-POP dataset is comprised of 500,000 high-quality images sampled from the LAION-5B dataset (Schuhmann et al. 2022), incorporating a variety of visual modalities across multiple disciplines, including images of natural scenes, objects, people, and artistic images, providing extensive breadth for the training and evaluation of diffusion models (DMs). Our dataset preprocessing pipeline includes standard normalisation processes and low-resolution image and damage filtering, and ensures consistent image resolutions and aspect ratios to provide a fair data comparison for our model configurations and baseline methods.

A.2 Architectures and Implementation

Target Models: We examine the two most recent DMs: SD3.5 Large Turbo (8B parameters) and FLUX.1-Schnell (12B). Both models are state-of-the-art (SOTA) for few-step, diffusion-based T2I generation, producing high-quality high-resolution images in an average of 4 steps.
***HierarchicalPrune* Hyperparameters:** The pruning process is configured through hyperparameters tuned for the target architecture. For SD3.5 Large Turbo and FLUX.1-Schnell, we set α to 0.55 and 0.01, respectively. Hyperparameter α in Eq. (2) controls the strength of positional importance weighting within the network. Higher values of α emphasise positional weighting more strongly, prioritising the removal of components based on their network position, while lower α values reduce this positional weighting, allowing importance scores to be determined based more on component-specific contributions rather than architectural location.
Implementation: The *HierarchicalPrune* pipeline is implemented in PyTorch. In particular, we built on top of checkpoints and inference pipelines for SD3.5 Large Turbo and FLUX.1-Schnell from the `Diffusers` library (von Platen et al. 2022). This allows us both consistency with pre- and post-processing steps, and also reproducible experiments. The distillation training process is conducted on a single node of 8 A100 GPUs with 80 GB of VRAM.
Quantization Strategy: To further reduce the memory footprint of the models, we employed 4-bit weight quantization from using `bitsandbytes` (Dettmers et al. 2022, 2023).
Open-Sourcing: We plan to release our trained checkpoints, distillation and inference pipelines upon acceptance of the paper to assist the community to reproduce our work as well as to further advance this line of research.

A.3 Baseline Comparisons

We compare *HierarchicalPrune* with (1) BK-SDM and (2) KOALA, two prior works, employing depth pruning and knowledge distillation, and (3) SANA-Sprint-1.6B, the SOTA small-scale DM.
BK-SDM (Kim et al. 2024a): BK-SDM performs block pruning for U-Net-based diffusion using CLIP (Hessel et al. 2021) scores as importance metric, where it ranks blocks based on the reduction of CLIP scores of the output images from the model before and after pruning each block. After pruning, BK-SDM distils the compressed model using knowledge from the original unpruned model, to recover performance from the previous steps.
KOALA (Lee et al. 2024): KOALA uses an importance ranking approach by calculating input-output cosine similarity to inform the block pruning strategy for U-Net based DMs. The blocks of the transformer have intrinsic importance relative to input-output transformations by evaluating the cosine similarity between input and output representations/features rather than utilising an external quality metric of transformation.
SANA-Sprint-1.6B (Chen et al. 2025b): We use SANA-Sprint-1.6B as the state-of-the-art (SOTA) baseline for an efficient, small-scale diffusion purpose-built model, incorporating techniques such as deep-compression auto-encoder (Chen et al. 2025a) and linear attention (Katharopoulos et al. 2020). Intrinsically, SANA models require training of the transformer from scratch, which departs from compressing existing models.

All comparisons to the baselines and SANA-Sprint-1.6B are made with identical experimental protocols. We ensure all methods use the same prompt sets, generation parameters, and hardware to isolate confounded variables and allow performance attribution of each approach to the specific methodologies.

A.4 Evaluation Framework

Quantitative Metrics: We adopt two representative and complementary image quality assessment metrics, capturing two different aspects of the quality of generation. First, **GenEval** (Ghosh, Hajishirzi, and Schmidt 2023) provides an all-encompassing evaluation across three areas, *i.e.*, semantic fidelity, visual fidelity, and prompt adherence. GenEval is now widely used for evaluations of T2I models (Xie et al. 2025; Esser et al. 2024). Second, **HPSv2** (Wu et al. 2023a) provides a contrasting perspective on image quality assessment, with demonstrated effectiveness in estimating the perceptual preferences of humans about image quality. HPSv2 has been utilised in several recent works such as KOALA (Lee et al. 2024).
Human Preference Metrics: We conduct a user study to assess human preferences for the generated images that are difficult to capture with the quantitative quality metrics (see Appendix A.5 for further details regarding our user study design). Similarly to (Sheynin et al. 2023) and (Dai et al. 2023), we evaluate two aspects: (1) **Text Alignment**: How well a generated image follows the description of the text prompt, (2) **Image Quality**: The overall visual quality considering clarity, colour, composition, and other factors.
Hardware Testing Environment: On the system resource side, we report the measured peak memory usage and latency of running the target models on GPUs. Our evaluation was conducted across three different GPUs, representing deployment scenarios on cloud (A100 with 80GB), high-end consumer (A6000 with 48GB), and resource-constrained low-end GPUs (GTX 3090 with 24GB).

A.5 Detailed User Study Design

Acknowledging the inconsistency of automated metrics, especially in discerning fine aspects of image quality and preferences for aesthetics, we choose to conduct a full user study of methods following the protocol of (Sheynin et al. 2023) and (Dai et al. 2023). In our user evaluation method, we are focused on measuring two aspects of user experience:

- **Text Alignment**: Evaluators measure the extent to which the generated images reflected correctly the content item or detail in the input text prompts. Within this measure, evaluators take into account whether the required object appeared and its spatial relationship with other objects, about detail examined before the invocation of the process, and attributes/stylistic characteristics specified in the prompts.
- **Image Quality**: We ask the evaluator to judge the quality on a multi-factor basis, *e.g.*, clarity of image, colour accuracy and vibrancy, balance of composition (or lack thereof), uniformity of lighting, preservation of texture detail, and the absence of artefacts or distortion.

Experimental Protocol and Interface Design: Based on these evaluation criteria, we designed an online questionnaire that rigorously collects systematic and unbiased data. Figure 7 shows the user study interface, which provides a consistent and structured layout and enables a progressive evaluation workflow.

First (see bottom-left figure), participants were shown six images generated by six different methods using the same text prompt (displayed in grid format with adequate visual separation) and required to select the single best image based on their holistic review of the image quality. This can create an overall perceived quality ranking among the evaluated generation methods. Second (see bottom-right figure), participants rated images individually based on two aspects of user experience as discussed above using structured rating scales. Specifically, each image was rated in isolation, on both the Text Alignment and Image Quality dimensions using a 6-point Likert scale (0-5). Zero = Poor performance, and 5 = Excellent performance.
Randomisation Strategy and Bias Mitigation: Randomisation is necessary for the performance differences to be based on the quality of the generated images, not due to positional bias, order, or pattern recognised by the evaluator. Therefore, to mitigate bias and avoid participants discovering patterns in how we generated the order of images, we randomly ordered the presentation of images across all evaluation sessions. Specifically, each participant would see the images in different random sequences, and there were no fixed positional relationships between methods across prompt sets. Randomisation occurred in the image grid as well as the rating phases, thus maintaining the fair assessment of generated image quality across different methods.
Quality Control: To ensure that our assessment criteria are interpreted in the same way, participants receive introduction materials before starting the formal evaluation. These materials have detailed descriptions of both the Text Alignment and Image Quality dimensions. The questionnaire has some built-in systems to ensure data quality and participant attention across multiple evaluation sessions. The platform sets out clear progress indicators to orientate the participants and maintain their attention. Also, the platform requires fully completed ratings for all images in both dimensions before the participant can move on to the next prompt set.

B Additional Analysis

In this section, we provide additional analysis results that are not included in the main content of the paper due to the page limit.

B.1 SD3.5 Large Turbo

Individual Subcomponent Removal Analysis: In additional to Figure 4, where we showed full-block removal and two types of subcomponent removals, we include in Figure 8 three other types of subcomponents (d-f).

Joint Subcomponent Removal Analysis: To further understand the interdependencies between different MMDiT subcomponents, we conducted an analysis examining the effect of jointly removing multiple subcomponents from SD3.5 Large Turbo. Figure 9 shows the performance degradation patterns when pairs of subcomponents are simultaneously removed from the model.

The results reveal distinct interaction patterns between different subcomponent combinations. Many of the paired subcomponent removals exhibit significant performance degradation concentrated in the earlier blocks of the model, consistent with the critical role of early processing stages in establishing semantic structures for image generation. However, a notable exception emerges with the combination of Context Norm and Context MLP, which demonstrates remarkably minimal impact across the entire network depth. This suggests that these two components may have complementary or redundant functions within the context processing pathway, allowing the model to maintain performance when both are simultaneously removed.

The varied degradation patterns across different subcomponent pairs indicate that the MMDiT architecture contains both critical interdependent components and potentially redundant pathways, providing insights for targeted pruning strategies that preserve model quality while reducing computational overhead.

B.2 FLUX.1-Schnell

Individual Subcomponent Removal Analysis: We extended our analysis to FLUX.1-Schnell, which exhibits a unique two-stage architecture design. The model consists of blocks 0-18 containing transformer blocks with five subcomponents, followed by single transformer blocks with three subcomponents in the latter portion of the network.

Figure 10 presents the contribution analysis results for FLUX.1-Schnell. The architectural heterogeneity of this model provides additional insights into the role of different subcomponents across varying block types. The transformer blocks (Figures 10a-10f) show sensitivity patterns similar to those observed in SD3.5 Large Turbo, especially for Norm subcomponent, while the single transformer blocks (Figures 10g-10h) shows that the performance degradation is largely small.

Joint Subcomponent Removal Analysis: The joint removal analysis for FLUX.1-Schnell reveals architecture-specific patterns that differ markedly between the two block types. As shown in Figure 11, for transformer blocks, the results demonstrate that removing components from earlier network portions leads to substantial quality degradation across all subcomponent combinations. Particularly notable is the impact of removing all normalisation components, which negatively affects performance throughout the network depth, with the exception of Norm + MLP and Context Norm + Context MLP combinations, which show reduced impact.

In contrast, the single transformer blocks exhibit remarkable resilience to subcomponent removal, with minimal impacts on final output quality. This robustness suggests that the single transformer blocks may contain significant redundancy or that their simpler architecture naturally provides more fault tolerance. The differential sensitivity between block types reinforces the hierarchical nature of the FLUX.1-Schnell architecture, where early complex processing is critical while later simplified processing is more robust to subcomponent removal.

C Additional Results

We present additional results that are not included in the main content of the paper due page limit.

C.1 Computational Overhead Analysis

We investigate the computational overhead of using *HierarchicalPrune*. The overhead is primarily composed of two phases: (1) the contribution analysis for establishing hierarchical importance patterns and (2) the subsequent distillation process.

Contribution Analysis Overhead: The contribution analysis phase represents the foundational step of our methodology, systematically evaluating architectural components to establish hierarchical importance patterns. This analysis involves removing individual blocks or subcomponents and measuring the resulting performance degradation. The computational requirements varied across architectures:

- **SD3.5 Large Turbo:** Block-wise analysis required 6.3 hours (400 images per block). Fine-grained subcomponent analysis took an additional 6.3 hours (80 images per subcomponent).
- **FLUX.1-Schnell:** Block-wise analysis required 14.9 hours (400 images per block) and fine-grained analysis took an additional 12.2 hours (80 images per subcomponent).

The contribution analysis takes around 12.6 A100 GPU hours for SD3.5 Large Turbo and 27.1 A100 GPU hours for FLUX.1-Schnell. It systematically examines 38 blocks and 190 subcomponent combinations for SD3.5 Large Turbo, and 57 blocks with multiple subcomponent configurations for FLUX.1-Schnell (e.g., 19 transformer blocks with 5 components each, plus 38 single transformer blocks with 3 components each). The increased overhead for FLUX.1-Schnell is due to its higher inference cost and architectural complexity.

Overall, the contribution analysis phase requires 12.6-27.1 GPU hours for comprehensive architectural profiling. While substantial, this represents a one-time cost that is amortised across all deployments. Furthermore, training small-scale DMs

Table 4: Full quantitative evaluation of image quality with mean and standard error. Note that the standard error is presented only for HPSv2, as GenEval is the metric that reports the total number of correctly classified images with different attributes (standard error is not applicable to the GenEval metric). KOALA and BK-SDM experience substantial degradation of image quality when reducing memory usage by 20-30%. *HierarchicalPrune* achieves significantly reduced memory usage while maintaining the image quality close to the original models.

Model	Method	Memory (%)	GenEval ⇑	HPSv2 ⇑	Reduction ⇓
Linear DiT	SANA-Sprint	(100%)	0.77	29.61 (± 0.071)	-
SD3.5 Large Turbo	Original	(100%)	0.71	30.29 (± 0.074)	-
	KOALA	(79.4%)	0.37	19.99 (± 0.074)	38.5%
	BK-SDM	(79.4%)	0.38	21.21 (± 0.077)	41.2%
	Ours (HPP)	(79.4%)	0.03	11.08 (± 0.042)	79.4%
	Ours (+PWP)	(79.4%)	**0.71**	**29.97** (± 0.077)	**2.5**%
	Ours (+Quant)	(**22.5**%)	0.69	28.15 (± 0.078)	4.8%
	Ours (HPP)	(71.5%)	0.0	7.00 (± 0.041)	88.4%
	Ours (+PWP)	(71.5%)	0.46	21.74 (± 0.075)	31.9%
	Ours (+SGDistill)	(71.5%)	**0.64**	**27.29** (± 0.079)	**10.1**%
	Ours (+Quant, All)	(**20.5**%)	0.62	26.29 (± 0.082)	13.3%
FLUX.1 Schnell	Original	(100%)	0.66	29.71 (± 0.073)	-
	KOALA	(70.5%)	0.56	24.04 (± 0.075)	17.1%
	BK-SDM	(70.5%)	0.57	24.67 (± 0.074)	15.3%
	Ours (All)	(**19.6**%)	**0.64**	**28.69** (± 0.079)	**3.2**%

(1-2B parameter-sized) from scratch requires 140,000 and/or 200,000 A100 GPU hours for smaller models such as SD1.4 and SD2.1, respectively, as reported in (Pernias et al. 2024), let alone large-scale DMs (8-11B parameter-sized), which could require significantly more GPU hours to train from scratch. Yet, our contribution analysis cost equals <0.006% of typical DM training.
Distillation Training Overhead: Following the contribution analysis, the distillation training process represents the most computationally intensive component of our methodology, requiring approximately 603 A100 GPU hours for SD3.5 Large Turbo and 1260 A100 GPU hours for FLUX.1-Schnell. The computational requirement reflects the complexity of preserving image quality during aggressive compression, as our hierarchical distillation approach meticulously balances knowledge transfer between original and compressed models while respecting the positional sensitivity patterns identified through contribution analysis.
Overall Computational Requirement: When considered in its entirety, the contribution analysis phase requires 12.6-27.1 GPU hours for comprehensive architectural profiling across different model architectures, and the subsequent distillation pipeline demands 603-1260 A100 GPU hours. While substantial, these represent one-time costs that amortise across all subsequent deployments. To place this computational investment in perspective, training small-scale DMs (1-2B parameters) from scratch requires 140,000-200,000 A100 GPU hours for SD1.4 and SD2.1 models, respectively, as reported (Pernias et al. 2024), let alone large-scale DMs (8-11B parameters), which would require considerably more computational resources. Notably, our contribution analysis cost equals less than 0.006% of typical DM training overhead, while our distillation training process represents less than 0.30-0.63% of the computational cost required to train comparable models from scratch.

This perspective emphasises the efficiency of our compression approach: rather than training new compact models that could sacrifice quality, *HierarchicalPrune* leverages existing high-quality models and transforms them into deployable versions through targeted compression. Furthermore, as shown in Section 3, the reduced computational requirement of the compressed DMs shows practical benefits, enabling 77.5-80.4% memory reduction and 27.9-38.0% latency improvements while preserving the superior image quality that large-scale DMs provide. Overall, this cost-benefit strongly favours compression-based approaches over training compact models from scratch, particularly when deployment accessibility and quality preservation are paramount considerations.

C.2 Comprehensive Experimental Results

In this subsection, we provide the comprehensive experimental results.

Table 4 shows all the results presented in Section 3, aggregating main quantitative results and ablation study results, regarding quantitative quality metrics such as GenEval and HPSv2. Note that we also present standard error to provide statistical evidence for robust performance evaluation. The standard error is presented only for HPSv2 as GenEval is the metric that reports the total number of correctly classified images with different attributes, hence the standard error is not applicable to the GenEval metric.

Table 5: Comparison of the peak memory and mean and standard error of the per-step latency of the DM Transformer measured on representative hardware from three tiers, *i.e.*, NVIDIA A100 (80 GB VRAM), A6000 (48 GB VRAM) and GTX 3090 24 GB (VRAM).

			A100		**A6000**		**GTX 3090**	
Model	**Method**	**Memory** ⇓	**Latency** ⇓	**Reduction** ⇑	**Latency** ⇓	**Reduction** ⇑	**Latency** ⇓	**Reduction** ⇑
Linear DiT	SANA-Sprint	3.14 GB	37±0.5 ms	-	54±0.4 ms	-	54±0.7 ms	-
SD3.5 Large Turbo	Original	15.8 GB	444±0.4 ms	-	823±0.7 ms		922±3.8 ms	-
	KOALA	11.3 GB	348±0.3 ms	21.6%	637±0.6 ms	22.6%	816±0.7 ms	11.5%
	BK-SDM	11.3 GB	348±0.3 ms	21.6%	633±0.6 ms	23.1%	816±0.7 ms	11.5%
	Ours (All)	**3.24 GB**	**341±0.3 ms**	**23.2%**	**593±0.6 ms**	**27.9%**	**754±0.7 ms**	**18.2%**
FLUX.1 Schnell	Original	22.6 GB	430±2.7 ms	-	756±4.0 ms	-	997±4.6 ms	-
	KOALA	15.9 GB	221±2.0 ms	48.6%	432±3.9 ms	42.9 %	562±4.4 ms	43.6%
	BK-SDM	15.9 GB	221±2.0 ms	48.6%	432±3.9 ms	42.9 %	562±4.4 ms	43.6%
	Ours (All)	**4.44 GB**	245±1.9 ms	43.0%	469±3.6 ms	38.0 %	562±4.4 ms	43.6%

In terms of computational resources, as shown in Tables 2 and 5, our compression pipeline achieves 79.5% reduction in peak memory for SD3.5 Large Turbo i.e., from 15.8 GB to just 3.24 GB, and 80.4% for FLUX.1.Schnell, i.e. from 22.6GB to 4.44GB.

For latency measurements, we used three testing environments as described in Appendix A, and we ran inference on the HPSv2 dataset, using 400 prompts sampled equally from all four categories. We compare *HierarchicalPrune* with KOALA and BK-SDM, each incorporated with SD3.5 Large Turbo and FLUX.1-Schnell model. We have also included SANA-Sprint-1.6B as a strong baseline.

In Table 5, we report the mean and standard error of the per-step latency of the MMDiT inferences, as well as the reduction compare to each base model. We observe that *HierarchicalPrune* constantly achieves the highest latency reduction for both base models across representative hardware platforms for cloud (A100), high-end professional (A6000), and consumer-grade edge (GTX3090) GPUs.

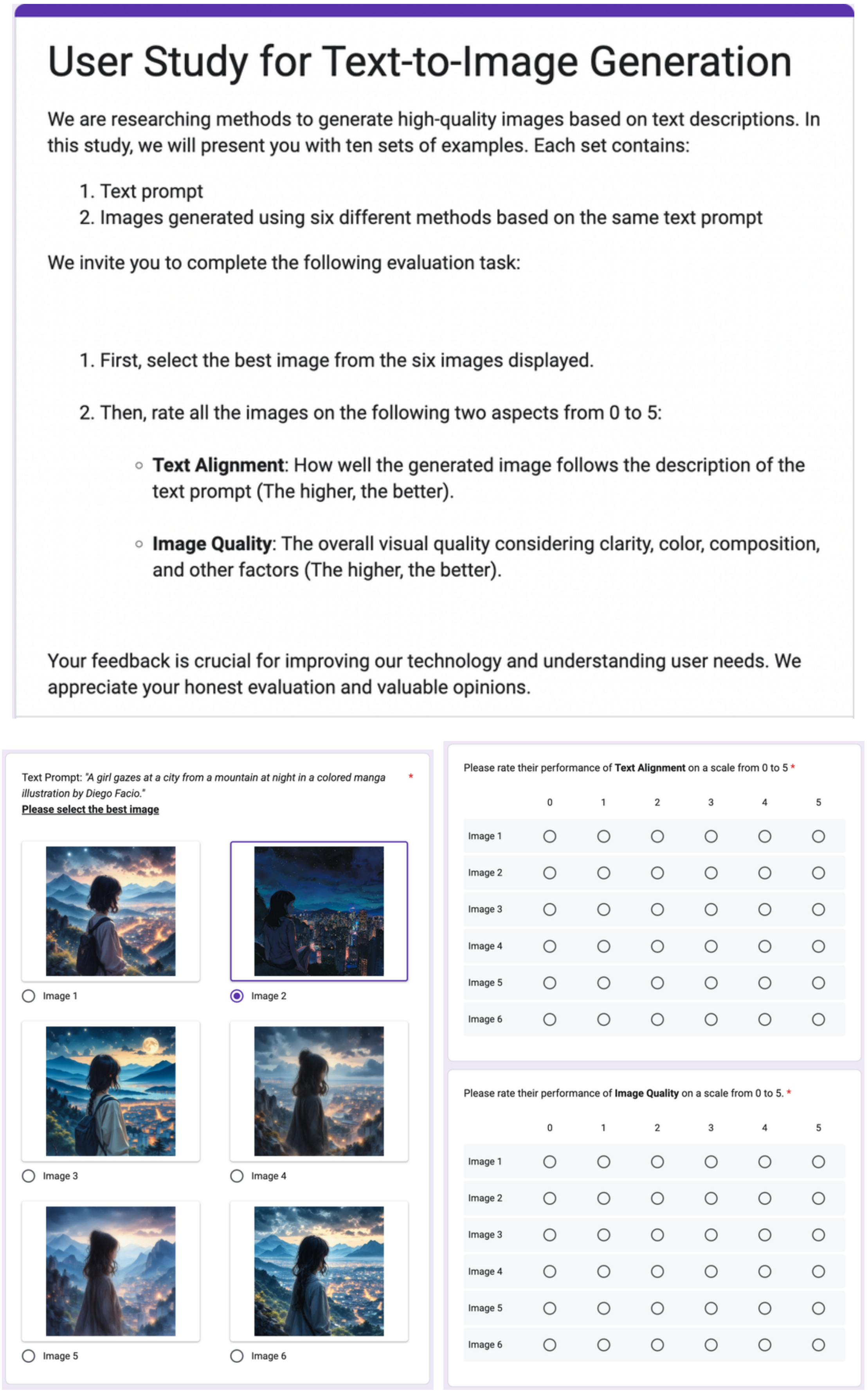
C.3 Additional Qualitative Image Results

In this subsection, we include image samples generated from *HierarchicalPrune* and baselines, to showcase the quality of outputs (see Figures 12 and 13).

D Limitations & Societal Impacts

Technical Limitations: While *HierarchicalPrune* demonstrates impressive results, several considerations remain. First, our approach requires initial profiling to identify hierarchical importance patterns. Yet, this is a one-time cost that gets amortised when applying the pruning/distillation method repeatedly to DMs with similar architectures. Second, while we have validated our dual-hierarchy insights across multiple MMDiT-based architectures (SD3.5 and FLUX), fundamentally different architectures might exhibit different patterns requiring adaptation.

Broader Impact: By reducing hardware requirements, our work enables broader access to state-of-the-art generative tools for researchers, artists, and educators with limited computing resources. Efficient inference aligns with sustainable AI practices, potentially lowering the carbon footprint of large-scale generative tasks. However, automation of creative workflows may impact industries reliant on human-generated art. We emphasise collaboration with policymakers to ensure equitable transitions. Moreover, like any research in efficient foundation models, faster and cheaper generation could amplify harmful applications (*e.g.*, spam and deepfake). We advocate for strict ethical guidelines, developer accountability, educational campaigns to raise awareness of generative AI's capabilities and risks and tools like watermarking to trace AI-generated content.



User Study for Text-to-Image Generation

We are researching methods to generate high-quality images based on text descriptions. In this study, we will present you with ten sets of examples. Each set contains:

1. Text prompt
2. Images generated using six different methods based on the same text prompt

We invite you to complete the following evaluation task:

1. First, select the best image from the six images displayed.
2. Then, rate all the images on the following two aspects from 0 to 5:
   - **Text Alignment**: How well the generated image follows the description of the text prompt (The higher, the better).
   - **Image Quality**: The overall visual quality considering clarity, color, composition, and other factors (The higher, the better).

Your feedback is crucial for improving our technology and understanding user needs. We appreciate your honest evaluation and valuable opinions.

Text Prompt: *"A girl gazes at a city from a mountain at night in a colored manga illustration by Diego Facio."* *

Please select the best image

○ Image 1 [✓ Image 2] ○ Image 3 ○ Image 4 ○ Image 5 ○ Image 6

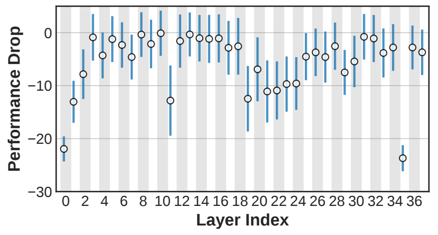
Please rate their performance of **Text Alignment** on a scale from 0 to 5 *

	0	1	2	3	4	5
Image 1	○	○	○	○	○	○
Image 2	○	○	○	○	○	○
Image 3	○	○	○	○	○	○
Image 4	○	○	○	○	○	○
Image 5	○	○	○	○	○	○
Image 6	○	○	○	○	○	○

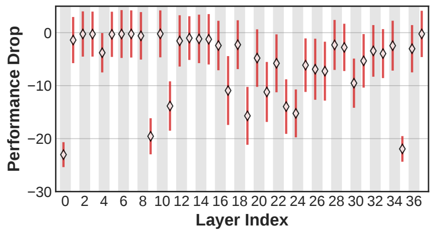
Please rate their performance of **Image Quality** on a scale from 0 to 5. *

	0	1	2	3	4	5
Image 1	○	○	○	○	○	○
Image 2	○	○	○	○	○	○
Image 3	○	○	○	○	○	○
Image 4	○	○	○	○	○	○
Image 5	○	○	○	○	○	○
Image 6	○	○	○	○	○	○

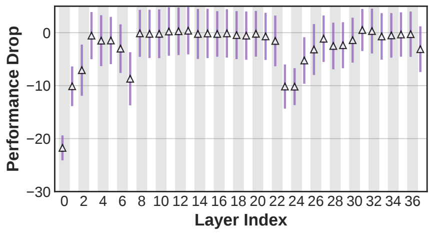
Figure 7: The user study interface that describes the objectives, instructions, evaluation criteria, and example image samples.



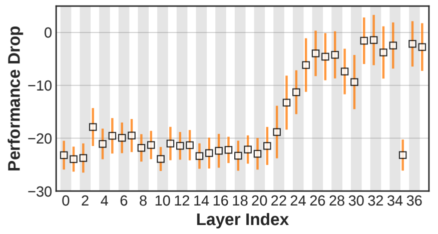
(a) Block Removal



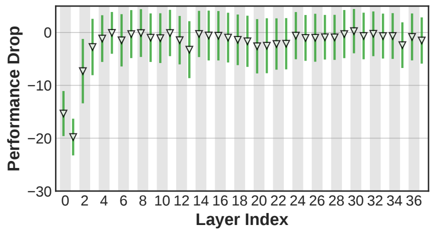
(b) Multi-modal Attention



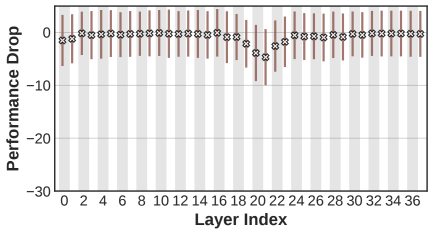
(c) MLP



(d) Norm

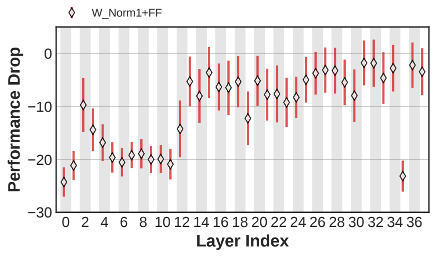


(e) Context Norm

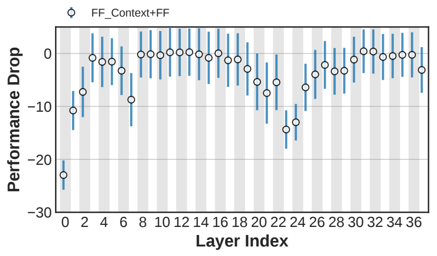


(f) Context MLP

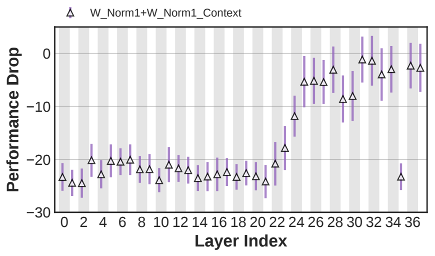
Figure 8: Fine-grained contribution analysis of SD3.5 Large Turbo on the HPSv2 dataset by removing either an entire MMDiT block or an intra-block subcomponent. We report the performance drop compared to the original model. (a) An entire MMDiT block is removed following prior depth pruning approaches (Lee et al. 2024; Kim et al. 2024a; Fang et al. 2024). (b,c,d,e,f) Each subcomponent of a MMDiT block is removed, revealing the different patterns of importance of each subcomponent.



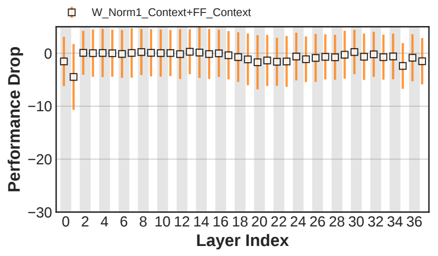
(a) Norm+MLP



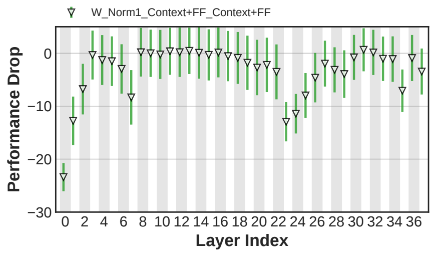
(b) MLP & Context MLP



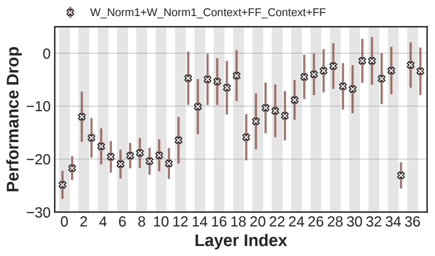
(c) Norm & Context Norm



(d) Context Norm & Context MLP

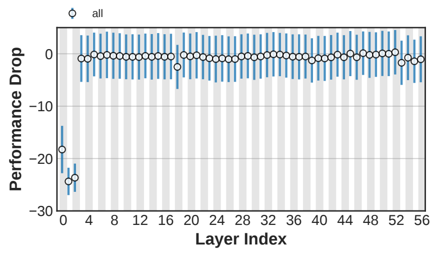


(e) Context Norm & All MLP

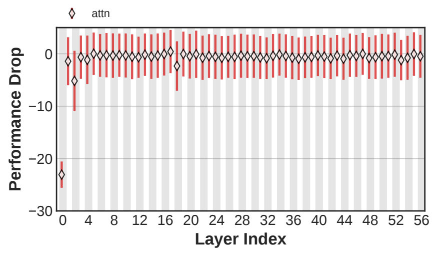


(f) All Norm & All MLP

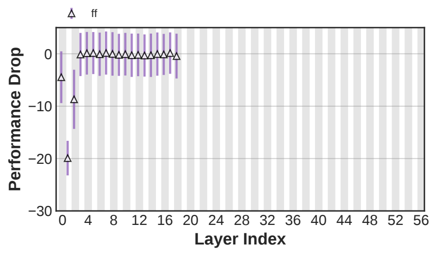
Figure 9: Fine-grained contribution analysis of SD3.5 Large Turbo on the HPSv2 dataset by jointly removing multiple subcomponent types. The performance drop follows different patterns for different pairs, with most type combinations leading to high performance degradation in earlier parts of the network, except for Context Norm + Context MLP (d), which has minimal impact.



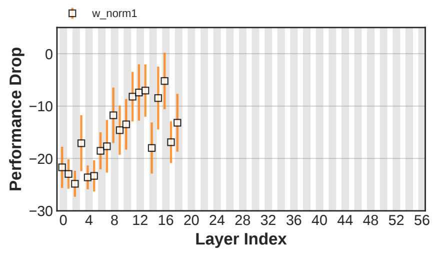
(a) Block Removal



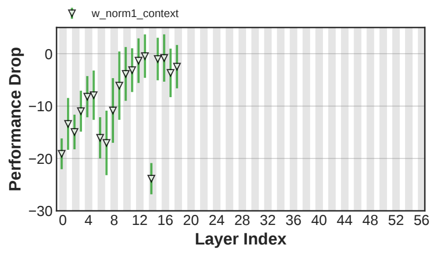
(b) Multi-modal Attention



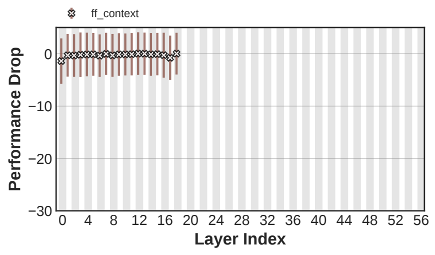
(c) MLP



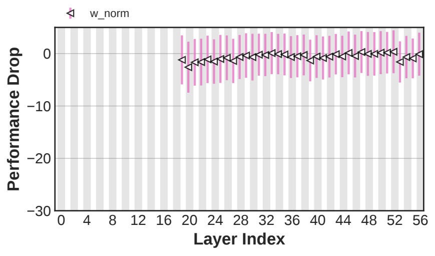
(d) Norm



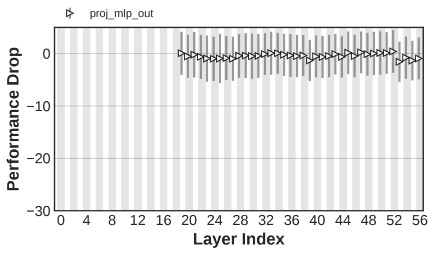
(e) Context Norm



(f) Context MLP

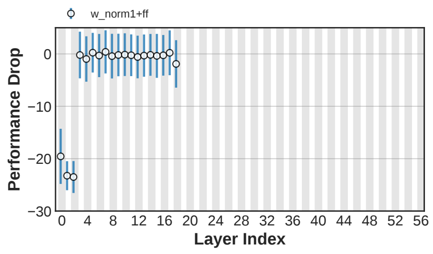


(g) (Single Transformer) Norm

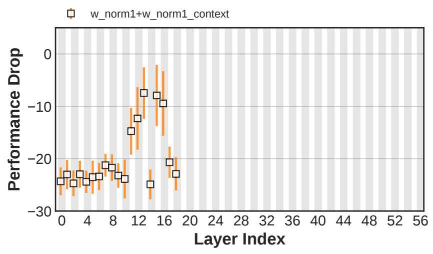


(h) (Single Transformer) MLP & Proj

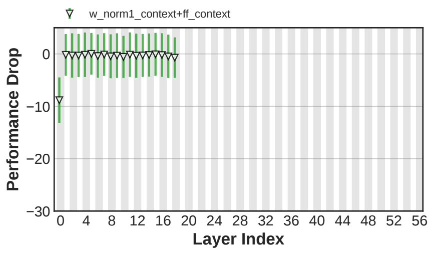
Figure 10: Fine-grained contribution analysis of FLUX.1-Schnell on the HPSv2 dataset by removing either an entire MMDiT block or an intra-block subcomponent. We report the performance drop compared to the original model. (a) An entire MMDiT block is removed following prior depth pruning approaches (Lee et al. 2024; Kim et al. 2024a; Fang et al. 2024). (b,c,d,e,f,g,h) Each subcomponent of a MMDiT block is removed, revealing the different patterns of importance of each subcomponent. Note that the first part of the model from block 0 to 18 is composed of the transformer blocks with five components (b-f) and the second part of the model is made of single transformer blocks with three components (b,g,h).



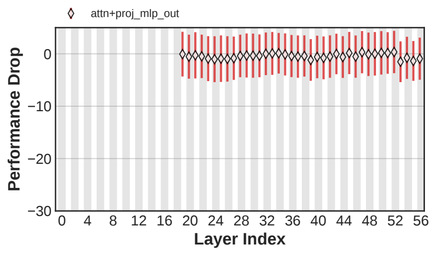
(a) Norm+MLP



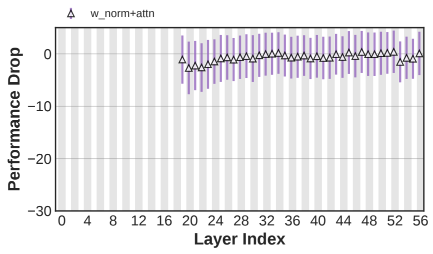
(b) Norm & Context Norm



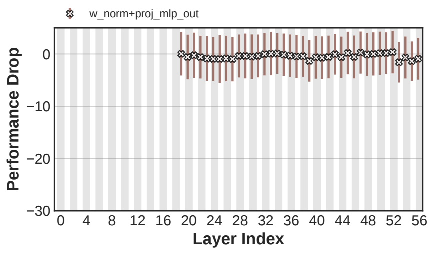
(c) Norm Context & Context MLP



(d) (Single Transformer) Attn & MLP



(e) (Single Transformer) Norm & Attn



(f) (Single Transformer) Norm & MLP

Figure 11: Fine-grained contribution analysis of FLUX.1-Schnell on the HPSv2 dataset by jointly removing multiple subcomponent types. The performance drop follows different patterns for different pairs for the transformer blocks (a,b,c) and for the single transformer blocks (e,d,f). For the transformer blocks (a,b,c), it shows that removing the earlier part of the network leads to huge degradation of the quality for all the combinations of the subcomponents and removing all the norms (b) affects negatively across all the portion of the network, except for Norm + MLP (a) and Context Norm + Context MLP (c), which has smaller impact. For the single transformer blocks, removing subcomponents have minimal impacts on the final outputs as shown in (e,d,f).

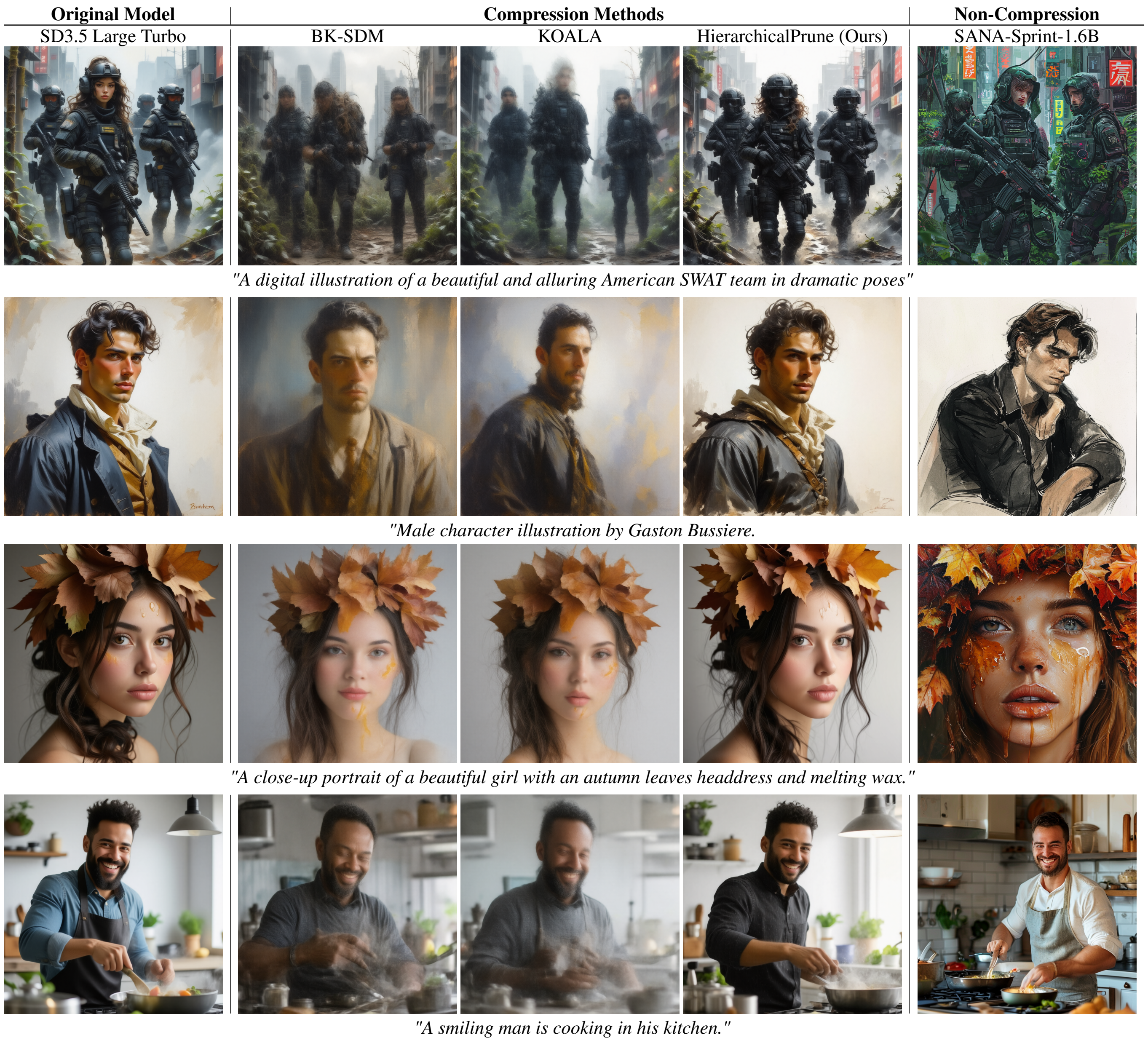

Original Model
Compression Methods
Non-Compression
SD3.5 Large Turbo
BK-SDM
KOALA
HierarchicalPrune (Ours)
SANA-Sprint-1.6B
"A digital illustration of a beautiful and alluring American SWAT team in dramatic poses"
"Male character illustration by Gaston Bussiere.
"A close-up portrait of a beautiful girl with an autumn leaves headdress and melting wax."
"A smiling man is cooking in his kitchen."

Figure 12: Visual comparison demonstrating the quality difference between the original model (column 1), depth pruning based on BK-SDM (column 2), KOALA (column 3), and our proposed *HierarchicalPrune* (column 4), as well as SOTA small-scale diffusion model, SANA-Sprint-1.6B (column 5). Our approach successfully maintains visual quality while delivering 79.5% memory reduction over the original model. Notably, our method preserves the text drawing capability of the original SD3.5 Large Turbo model where SANA-Sprint-1.6B is limited.

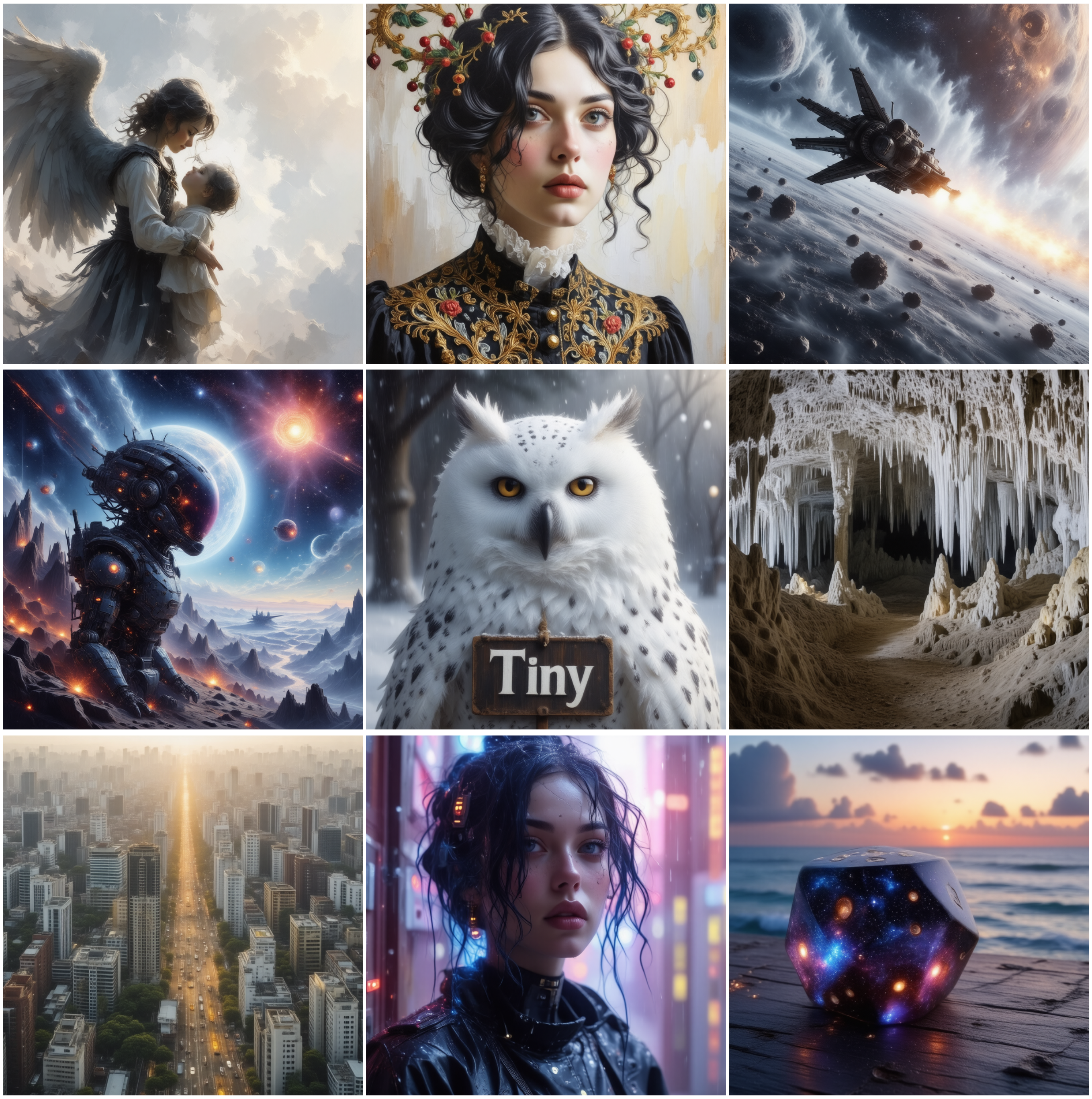


Figure 13: High-resolution image samples generated by our pruned/distilled model using our proposed method, *Hierarchical-Prune*, showcasing its superior visual quality across various visual styles, precisely following text prompts, and preserving the ability to draw typography.