I-DiT: Integer-Only Quantization for Diffusion Transformers

Anonymous authors
Paper under double-blind review

Abstract

Diffusion transformers achieve remarkable generative performance but remain challenging to deploy on resource-constrained hardware due to their high computational and memory costs. These devices often rely on specialized AI accelerators whose efficiency comes primarily from low-precision integer-only datapaths. However, existing diffusion quantization methods still rely heavily on floating-point operations, and integer-only frameworks are not readily applicable to modern diffusion transformers due to fixed quantization scales, lack of tiled attention support, and sensitivity to activation outliers. For this, we present I-DiT, the first framework for fully integer-only inference of diffusion transformers without costly fine-tuning. First, we propose a dynamic scale adjustment method for integer-only frameworks, that adapts to varying activation ranges during runtime. Second, we introduce log2-based online softmax that enables numerically stable and memory-efficient tiled attention in integer. Lastly, we reformulate dyadic requantization to support finer-grained activation quantization, localizing the impact of outliers while preserving integer-only execution. Experiments across PixArt, SD3, and FLUX demonstrate that I-DiT preserves generation quality while enabling efficient fully integer-only inference for large-scale diffusion models.

1 Introduction

Diffusion models have become a standard approach for state-of-the-art generative pipelines (Ho et al., 2020; Song et al., 2020), particularly for high-resolution images and conditional generation (Ho & Salimans, 2021; Rombach et al., 2022; Peebles & Xie, 2023; Chen et al., 2023a; 2024). Among existing architectures, Diffusion Transformers (DiTs, Peebles & Xie (2023)) have been widely adopted for their superior performance, now serving as a representative choice for image generation. Despite their success, DiT models incur significant inference costs due to repeated execution of large transformer blocks, leading to high latency and energy consumption in practice.

This computational burden restricts the deployment of DiTs beyond datacenter GPUs toward resource-constrained hardware, such as FPGAs (Blott et al., 2018), edge NPUs (Sun & Kist, 2021), PIM accelerators (Gómez-Luna et al., 2022), and microcontrollers (Lai et al., 2018). For example, recent demonstrations of on-device diffusion generation on mobile AI accelerators (Li et al., 2023b; Zhao et al., 2023; Chen et al., 2023b) highlight a growing demand for executing diffusion models under tight compute, memory, and power budgets. Such platforms commonly provide highly optimized low-precision integer datapaths for energy efficiency (Arm, 2025; 2024; Qualcomm), making quantization (Nagel et al., 2021) an important enabler for practical deployment.

Unfortunately, existing diffusion quantization methods (He et al., 2023b; So et al., 2023; Li et al., 2025; He et al., 2023a; Li et al., 2023a; Zhao et al., 2024; 2025; Huang et al., 2024; Mills et al., 2025; Chen et al., 2025a) largely overlook nonlinear functions, modulation, and attention, leaving these operations in floating-point while focusing primarily on accelerating GEMMs. Thus, they remain dependent on floating-point hardware, limiting their applicability to integer-centric edge and embedded accelerators where floating-point operations are costly (Lin et al., 2024; Wang et al., 2022; Abdelaziz et al., 2021; Silvano et al., 2025; Alexandridis & Dimitrakopoulos, 2025).

While there exist a few integer-only inference frameworks (Kim et al., 2021; Li & Gu, 2023; Yao et al., 2021), we observe that directly extending them to DiTs exposes three challenges: **1) Pre-calibrated quantization scales cannot adapt to the highly dynamic activation ranges**

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.

of diffusion models. Diffusion model activation distributions vary substantially across denoising timesteps (So et al., 2023; Huang et al., 2024) and conditioning inputs (Ryu et al., 2025; Lee et al., 2025), but existing methods cannot dynamically select scales without the floating-point activation range. **2) Existing integer attention cannot support memory-efficient tiled attention.** Existing integer attention designs force materializing large attention intermediates, requiring large memory often unavailable in resource-constrained devices. **3) Coarse-grained dyadic quantization is vulnerable to activation outliers.** Existing integer-only frameworks adopt coarse-grained activation quantization under dyadic scale constraints, making them sensitive to activation outliers. Together, these limitations hinder integer-only quantization for modern diffusion architectures.

To this end, we introduce I-DiT, the first framework that enables fully integer-only inference for large-scale DiTs without requiring costly full retraining for quantization. First, we propose dynafit, which dynamically adjusts pre-calibrated scales without accessing floating-point activations at runtime. Then, we devise an integer-only online softmax algorithm for fully integer tiled attention, effectively reducing the peak memory needed for each inference. Lastly, we reformulate integer requantization to support finer-grained activation quantization, localizing the influence of outliers. Our results suggest that integer-only diffusion models are not only feasible, but practical, opening new opportunities for deploying generative models on efficient and specialized hardware.

- We propose I-DiT, the first fully integer-only inference framework for diffusion transformers, eliminating floating-point computation throughout the denoising pipeline.
- We devise dynafit, a runtime scale adjustment method that adapts pre-calibrated scales to dynamic activation ranges without floating-point scale computation.
- We introduce an integer-only online softmax algorithm, enabling stable and memory-efficient tiled attention entirely in integer arithmetic.
- We reformulate dyadic requantization to support finer-grained activation quantization, effectively localizing the impact of outliers.

2 BACKGROUND

Quantization reduces computation and memory costs by replacing high-precision operations with low-bit operations. I-DiT uses *uniform symmetric quantization* where values are mapped to a uniformly spaced integer within a symmetric range. Given a tensor X, each $x \in X$ is quantized as:

$$\tilde{x} = Q(x; k, s) = \text{clamp}\Big(\Big\lfloor \frac{x}{s} \Big\rceil, -2^{k-1}, 2^{k-1} - 1\Big), \tag{1}$$

where quantization scale s denotes $2\alpha/(2^k - 1)$, defined for k-bit integer representation, and $\alpha\ (\geq 0)$ denotes the maximum floating-point value in X. Then, dequantization is simply an element-wise scaling $x \approx DQ(\tilde{x}; k, s) = s \cdot \tilde{x}$. For matrix multiplication ($Y = XW$), the quantized operation can be formulated as $Y = XW \approx S_X S_W \odot \tilde{X}\tilde{W}$. After adding the output scale S_Y, the requantization from integer input $\tilde{X}$ to integer output $\tilde{Y}$ can be directly written as:

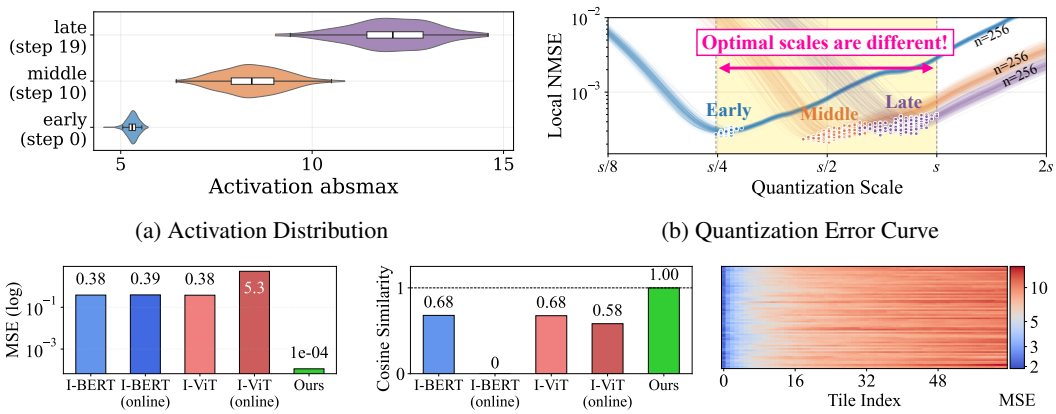
$$\tilde{Y} = Q(Y; k, S_Y) \approx \frac{S_X S_W}{S_Y} \odot \tilde{X}\tilde{W}. \tag{2}$$

In this work, we focus on lightweight *post-training quantization (PTQ)*, which avoids heavy model fine-tuning. For activation, we adopt *fine-grained* quantization where values along the inner dimension share a scale for each outer-dimension index, which we denote as *per-token quantization*.

Integer-Only Quantization enables end-to-end model deployment for integer-only hardware by removing floating-point operations. *Non-linear approximations* are used to replace activation functions with integer operations via low-order polynomials and approximating bit-shifting residuals. In addition to that, *Dyadic quantization* (Yao et al., 2021) converts the quantization scales to dyadic numbers, which are rational numbers of the form $b/2^c$, for nonnegative integers b and c. This allows the quantization process to be done with integer multiplication and bit-shifting during inference. Using this, quantized matrix multiplication output $\tilde{Y}$ in Eq. 2 becomes:

$$\tilde{Y} \approx DN(\frac{S_X S_W}{S_Y}) \odot \tilde{X}\tilde{W} = (b \odot \tilde{X}\tilde{W}) \gg c, \tag{3}$$

where $DN(\cdot)$ approximates a floating-point value as $b/2^c$, allowing requantization to operate directly on integer accumulators via multiply-and-shift without floating-point dequantization.



(a) Activation Distribution (b) Quantization Error Curve

(c) MSE (d) Cosine Similarity (e) Tile-wise Reconstruction Error

Figure 1: (a) activation range of PixArt-Σ, $\mathrm{block.19.attn1}$ (pre-proj.) using 256 inputs sampled from 3 timesteps, (b) quantization error curve of each sample using different quantization scales, (c-d) attention output error across different integer attention variants, using MSE(↓) and cosine similarity(↑), (e) shows reconstruction error in the attention output as more tiles are processed.

Tiled Attention processes the attention matrix in tiles for memory efficiency rather than fully materializing it. Given attention scores $S = QK^\top$, the original row-wise softmax for row i is:

$$\mathrm{Softmax}(S_i)_j = \exp(S_{ij} - m_i) \big/ \textstyle\sum_k \exp(S_{ik} - m_i), \qquad m_i = \max_k S_{ik}. \tag{4}$$

and the attention output is computed as $O_i = \sum_j \mathrm{Softmax}(S_i)_j V_j$. In tiled attention (Dao et al., 2022), matrices are partitioned into tiles (e.g., $Q_i \in \mathbb{R}^{B_r \times d}$ and $K_j, V_j \in \mathbb{R}^{B_c \times d}$). For query tile i and key-value tile j, the running max is updated as $m_i^j = \max(m_i^{j-1}, \max_k S_{ik}^j)$, and the remaining running statistics are updated accordingly:

$$\ell_i^j = \exp(m_i^{j-1} - m_i^j)\, \ell_i^{j-1} + \sum\nolimits_k \exp(S_{ik}^j - m_i^j), \tag{5}$$

and similarly for the accumulated output: $o_i^j = \exp(m_i^{j-1} - m_i^j)\, o_i^{j-1} + \sum_k \exp(S_{ik}^j - m_i^j) V_k^j$. After all T tiles are processed, the final attention output is obtained as $O_i = o_i^T / \ell_i^T$.

3 Challenges

Dynamic Quantization under Integer Inference. Diffusion models repeatedly evaluate the network over different denoising timesteps, causing the activation distribution to change throughout the generation process (Fig. 1a). This mismatch directly affects quantization accuracy. Fig. 1b varies the quantization scale around the fixed calibrated scale (s) and measures the resulting reconstruction error. Each dot denotes an individual sample's minimum error scale. These optimal scales span a wide range, varying substantially both across samples and denoising stages. Consequently, a fixed scale cannot consistently minimize quantization error across different inputs.

Conventional dynamic quantization handles such distribution shifts by determining scales from floating-point activations at runtime (Ryu et al., 2025). This is incompatible with integer-only inference, as activations must remain in integers throughout the computation graph. Thus, integer-only inference requires adapting scales to runtime ranges using only integer information.

Integer-Only Online Softmax. Softmax is particularly challenging for integer-only arithmetic as exponentiation amplifies small perturbations in attention scores. Its wide dynamic range increases overflow and underflow under bounded integer representations. Existing integer-only methods (Kim et al., 2021; Li & Gu, 2023) address this problem with integer approximations ($\exp_{int}(\cdot)$), representing $e^x \approx s_e \exp_{int}(x)$ with an integer value and its quantization scale s_e. While this remedy is sufficient for conventional softmax, directly applying it to online softmax introduces additional difficulty. Online softmax repeatedly renormalizes the accumulated result when the row-wise running

maximum (m_i^j) is updated:

$$\ell_i^j = \underbrace{\exp(m_i^{j-1} - m_i^j)}_{\text{renormalization } r_i^{(j)}} \ell_i^{j-1} + \sum\nolimits_k \exp(S_{ik}^j - m_i^j), \tag{6}$$

This recurrence creates a problem under integer quantization. In specific, the integer approximation $r_i^{(j)} \approx s_r^{(j)} \exp_{int}(m_i^{j-1} - m_i^j)$ repetitively multiplies the accumulator ℓ_i^{j-1} and its scale $s_\ell^{(j-1)}$:

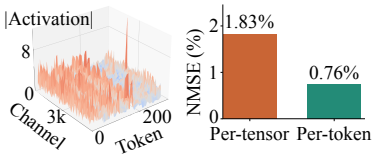
$$r_i^{(j)} \ell_i^{j-1} \approx \left(s_r^{(j)} s_\ell^{(j-1)}\right) \left(\tilde{r}_i^{(j)} \tilde{\ell}_i^{j-1}\right). \tag{7}$$

Thus, as each tile is processed, the scale parameter accumulates along with the accumulator values. This scale accumulation can rapidly drive the accumulator scale toward underflow, compromising numerical stability. Although requantizing the accumulator after each tile can bound the scale, it introduces repeated rounding errors and per-tile requantization overhead, undermining both the numerical accuracy and efficiency. Fig. 1 c-e shows that integer exponentiation with online softmax increases attention output error. I-ViT increases MSE by orders of magnitude (Fig. 1c) and I-BERT reaches zero cosine similarity (Fig. 1d), which both indicate a severe degradation. Fig. 1e further shows the error increase as more tiles are processed, due to the renormalization scale decaying exponentially across tiles. The same difficulty exists for accumulated output (o_i^j). In contrast, our method achieves three orders of magnitude lower MSE and near-perfect cosine similarity.

Quantization Granularity under Dyadic Constraints. Dyadic quantization requires sharing the requantization factor across large groups, making fine-grained quantization difficult. Consider per-token activation quantization combined with per-channel weight quantization. Their scales vary along different dimensions of the GEMM output, such that $DN(S_W S_X / S_Y)$ form an outer product:

$$\tilde{Y}_{ij} = DN\left(S_W[i] S_X[j] / S_Y[j]\right) (\tilde{X}\tilde{W})_{ij}, \qquad S_W \otimes (S_X / S_Y) \in \mathbb{R}^{r \times c}, \tag{8}$$

where $S_W \in \mathbb{R}^r$ and $S_X / S_Y \in \mathbb{R}^c$. Consequently, each output element requires a different dyadic multiplier and shift. Naïvely supporting these fine-grained scales therefore requires either materializing an activation-sized matrix of requantization parameters or generating and applying them elementwise at runtime, introducing substantial storage, bandwidth, and computation overhead. Falling back to per-tensor activation quantization avoids this overhead, but can substantially degrade quantization precision due to large outliers (Chen et al., 2025b; Mills et al., 2025) that expand the shared quantization range. Figure 2 demonstrates the effect of outliers in activation quantization. Per-token quantization reduces error by localizing the outliers to a smaller group of values. As such outliers are persistent in diffusion models, fine-grained activation scaling is important for preserving quantization quality. Nevertheless, its direct combination with dyadic requantization undermines the efficiency that integer-only execution is intended to provide.



(a) Activation Dist. (b) Quantization Error

Figure 2: Activation distribution and quantization error from SD3-medium block. 8. ff_context. proj.

4 METHODOLOGY

4.1 DYNAFIT: INTEGER-ONLY DYNAMIC SCALE ADAPTATION

To address the mismatch between static calibration and runtime activation ranges, we propose dynafit, an integer-only dynamic scale adaptation method. Our key observation is that adapting the quantization scale does not require the actual floating-point activation range. Instead, the calibrated scale provides a reference range, and we only need to determine how far the current activation range deviates from this reference. This discrepancy can be inferred directly from the occupancy of the integer quantization range, without dequantization or floating-point range estimation. We restrict this adjustment to powers of two, such that it can be seamlessly absorbed into dyadic requantization.

Based on the observation, we parameterize the runtime scale of each quantization group g as $S_g = \bar{S}_g 2^{\delta_g}$, where $\bar{S}_g$ is the frozen calibrated scale and the integer exponent δ_g acts as a lightweight runtime correction. $\delta_g > 0$ expands the representable range to prevent clipping, whereas $\delta_g < 0$ contracts it to provide finer quantization intervals when integer bins are underutilized.

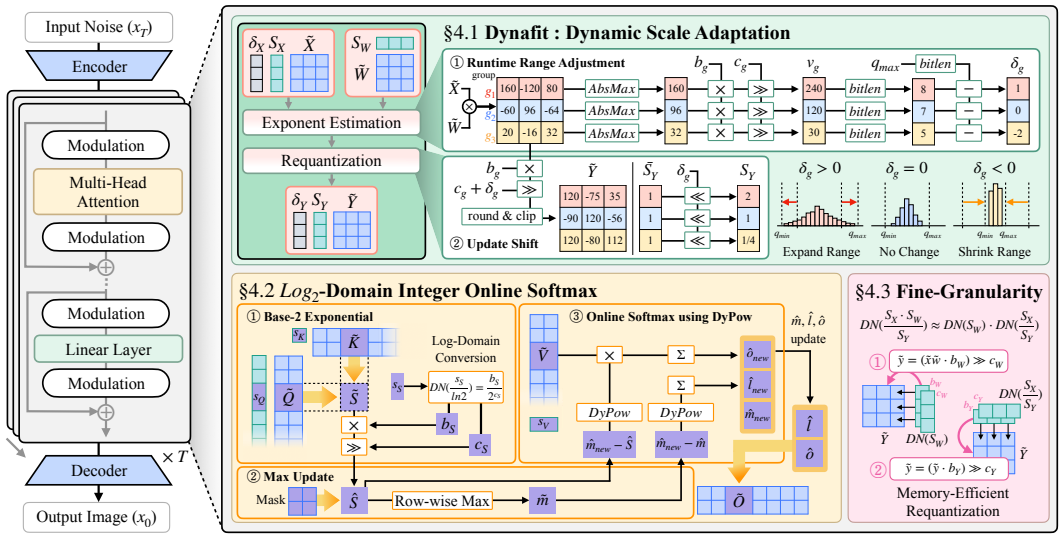


Figure 3: The overall framework of I-DiT. Dynafit dynamically adapts the static range to activation range (§ 4.1). The $\log_2$-domain integer online softmax (§ 4.2) enables fully integer online softmax by replacing exponentials with base-2 log-domain shifts via $\mathrm{DyPow}(x)$. I-DiT operates with finer granularity under dyadic quantization (§ 4.3), preserving memory efficiency and output quality.

To determine δ_g, we estimate integer range occupancy under the calibrated output scale $\bar{S}_g$. For a GEMM with integer input $\tilde{X}$ and weight $\tilde{W}$, let $DN(\bar{S}_X S_W/\bar{S}_Y) = b/2^c$ denote the calibrated dyadic conversion factor (Eq. 3). Before output clamping, we compute

$$v_g = \left\lfloor \max_{i \in g} \left| \widetilde{X}\widetilde{W} \right|_i \cdot b_g 2^{-c_g} \right\rfloor, \tag{9}$$

where v_g is the maximum output magnitude that would result under the calibrated integer grid. Since we use powers of two for scale correction, selecting δ_g requires only the binary-order mismatch between v_g and $q_{\max}$, rather than their exact ratio. This mismatch is directly captured via their bit-length difference, allowing the exponent to be selected using only integer operations:

$$\delta_g = \mathrm{clamp}\left(\mathrm{bitlen}(v_g) - \mathrm{bitlen}(q_{\max}), \delta_{\min}, \delta_{\max}\right), \qquad v_g > 0 \tag{10}$$

The bit-length difference directly measures the binary-order mismatch between the current range and the representable integer range, avoiding division or floating-point scale computation.

Integration with dyadic requantization. The power-of-two restriction provides an additional advantage, where the runtime correction can be absorbed directly into the shift of the existing dyadic requantization. Let δ_X and δ_Y denote the input and output scale-adjustment exponents, respectively. Using the calibrated $DN(\bar{S}_X S_W/\bar{S}_Y) = b/2^c$, Eq. 3 becomes

$$\tilde{Y} \approx \left(b \odot \tilde{X}\tilde{W}\right) \gg (c + \delta_Y - \delta_X), \tag{11}$$

where only the effective shift is changed. Rounding and clamping follow the quantizer Q (Eq. 1).

4.2 Log_2-Domain Integer Online Softmax

As we discussed in Sec. 3, integer online softmax suffers from scale compounding across tiles. Our key idea is to prevent this scale from propagating in the first place. To this end, we align all exponential terms to a common scale that can be removed within each tile update. We specifically choose a power-of-two common scale, allowing it to be eliminated exactly using a bit shift. Based on these principles, we reformulate online softmax in the $\log_2$ domain, where exponentiation naturally admits such a power-of-two representation. Using the identity $\exp(x) = 2^{x/\ln 2}$, we can reformulate the accumulator $\hat{\ell}_i^j$ to be updated using only powers of two with non-positive arguments:

$$\hat{\ell}_i^j = 2^{\hat{m}_i^{j-1} - \hat{m}_i^j}\, \hat{\ell}_i^{j-1} + \sum\nolimits_k 2^{\hat{S}_{ik}^j - \hat{m}_i^j}, \tag{12}$$

where $\hat{S}_i^j \triangleq S_i^j / \ln 2$ are the log-domain attention scores. To evaluate these terms without floating-point arithmetic, we introduce the integer primitive $\mathbf{DyPow}(\boldsymbol{x}) \approx \mathbf{2^{f_w} 2^{-x}}$, which represents the exponential using f_w fractional bits. The approximation decomposes x into its integer and fractional parts, where the integer component is evaluated exactly using a bit shift, and the bounded fractional component is approximated by a low-order integer polynomial. See Appendix B for details.

Importantly, $\mathrm{DyPow}(\cdot)$ always introduces the same power-of-two scale 2^{f_w} for every exponential term. We remove this factor by an exact right shift within each online update, so the exponential scale does not compound across tiles, and the accumulator maintains a fixed scale throughout the streaming computation. Accordingly, Eq. 12 becomes

$$\hat{\ell}_i^j = \left\lfloor \hat{\ell}_i^{j-1} \cdot \mathrm{DyPow}(\hat{m}_i^j - \hat{m}_i^{j-1}) \,/\, 2^{f_w} \right\rfloor + \sum\nolimits_k \mathrm{DyPow}(\hat{m}_i^j - \hat{S}_{ik}^j). \tag{13}$$

The same formulation applies for $\hat{o}_i^j$. After processing all key/value tiles, we compute the normalized attention output $\tilde{O}_i = \hat{o}_i / \hat{\ell}_i$ with a fixed-point integer reciprocal. By aligning the exponential terms to a common power-of-two scale and removing it at each tile update, the accumulator remains under a fixed scale throughout the streaming computation, preserving the tile-wise, memory-efficient computation using only integer arithmetic. Algorithms 1 to 2 and Figure 3 summarize the procedure.

4.3 Fine-Grained Dyadic Requantization

Fine-grained quantization introduces scales along different dimensions of a GEMM, with token-wise activation scales and output-channel-wise weight scales. Naïvely merging these scales for requantization produces a two-dimensional scale matrix, whose size grows quadratically with the token and channel dimensions. For this, we devise efficient dyadic requantization for fine-grained quantization. Our key observation is that this matrix does not have to be materialized. To be specific, we modify the requantization (Equation (3)) by factoring the merged dyadic scale into two components:

$$\tilde{Y} \approx \tilde{X}\tilde{W}\big(DN(S_W)\big)\, DN(S_X/S_Y)\,, \tag{14}$$

where S_W and S_X/S_Y are saved as dyadic numbers. During inference, requantization is implemented as two sequential dyadic scaling operations, each with an integer multiplication and shift.

$$\tilde{Y} = (\tilde{X}\tilde{W} \cdot b_W) \gg c_W, \qquad \tilde{Y} = (\tilde{Y} \cdot b_Y) \gg c_Y, \tag{15}$$

where $DN(S_W) = b_W / 2^{c_W}$ and $DN(S_X/S_Y) = b_Y / 2^{c_Y}$, with $b_W, c_W, b_Y, c_Y \in \mathbb{Z}_{\geq 0}$. Keeping these scales separate avoids the $O(rc)$ merged matrix, reducing scale storage to $O(r + c)$ while enabling fine-grained quantization with memory-efficient integer-only execution.

5 Evaluation

Experiment Settings. We evaluate PixArt-Σ, PixArt-α, SD3-medium, and FLUX.1-schnell on 5K prompts each from MJHQ (Li et al., 2024), sDCI (Urbanek et al., 2024), and COCO2014 (Lin et al., 2014). Baselines comprise integer-only I-BERT (Kim et al., 2021) and I-ViT (Li & Gu, 2023), with five diffusion quantization methods: Q-Diffusion (Li et al., 2023a), Q-DiT (Chen et al., 2025a), TDQ (So et al., 2023), PTQD (He et al., 2023b), and PTQ4DM (Shang et al., 2023). We use the original integer-only setting (W8A8 by default, with higher precision for selected operations). For diffusion baselines, we replace floating-point nonlinear operations with I-ViT counterparts for integer inference. Calibration uses 1024 random COCO2017 prompts. See Appendix A for details.

Visual Quality Comparison. Table 1 and Figure 4 show that I-DiT largely preserves generation quality under fully integer-only inference, while many quantized baselines degrade. On MJHQ, I-DiT achieves an FID of 17.92 on PixArt-Σ (FP16: 16.58; best baseline: 41.96) and 23.38 on SD3-medium (FP16: 22.12; best baseline: 49.10). Its FID advantage over quantized baselines on these two models also holds for sDCI and COCO. Beyond distribution-level quality, I-DiT achieves the lowest LPIPS among quantized methods across all evaluated settings, indicating stronger consistency with the corresponding floating-point outputs. This advantage remains clear on PixArt-α, where several baselines are competitive in distribution-level metrics but show substantially worse reference similarity. Qualitatively, competing method often introduce semantic failures, noise, or visual artifacts, whereas I-DiT consistently produces more faithful and visually coherent images.

Table 1: Quality comparison on MJHQ, sDCI, and COCO datasets

Model	Method	MJHQ				sDCI				COCO			
		IR↑	FID↓	sFID↓	LPIPS↓	IR↑	FID↓	sFID↓	LPIPS↓	IR↑	FID↓	sFID↓	LPIPS↓
PixArt-Σ (20 Steps)	FP16	0.95	16.58	14.38	–	0.96	24.85	17.19	–	0.93	34.34	19.04	–
	I-BERT	0.04	41.96	20.67	0.60	0.14	47.51	22.31	0.62	0.25	44.45	22.92	0.62
	I-ViT	0.26	50.40	22.53	0.61	0.37	49.60	27.95	0.64	0.50	51.03	26.03	0.64
	Q-Diffusion	-0.80	94.21	37.98	0.70	-0.75	96.66	40.78	0.73	-0.66	93.75	36.94	0.72
	Q-DiT	0.24	51.02	25.23	0.61	0.34	49.98	31.72	0.63	0.42	53.34	28.30	0.64
	TDQ	-0.64	102.41	32.83	0.74	-0.85	118.43	40.69	0.76	-0.65	119.09	38.02	0.76
	PTQD	-0.74	94.44	38.39	0.70	-0.68	97.19	41.69	0.73	-0.57	92.81	37.60	0.72
	PTQ4DM	0.02	57.25	23.09	0.66	0.04	59.33	28.06	0.69	0.16	59.57	26.62	0.69
	Ours	**0.83**	**17.92**	**15.79**	**0.49**	**0.83**	**27.07**	**17.27**	**0.52**	**0.79**	**33.07**	**19.40**	**0.53**
PixArt-α (20 Steps)	FP16	0.99	16.03	15.09	–	0.83	27.51	17.57	–	0.85	37.73	22.50	–
	I-BERT	0.53	22.91	19.82	0.48	0.43	31.69	20.52	0.49	0.51	35.15	29.05	0.49
	I-ViT	0.78	26.37	17.47	0.54	**0.72**	29.39	19.32	0.51	**0.77**	31.94	22.26	0.50
	Q-Diffusion	-0.09	75.98	28.10	0.61	-0.44	96.02	30.31	0.68	-0.10	78.54	34.80	0.63
	Q-DiT	0.65	26.70	**16.08**	0.62	0.66	**27.97**	19.49	0.59	0.67	**30.80**	22.01	0.58
	TDQ	0.59	52.48	25.16	0.74	0.44	62.28	32.72	0.81	0.60	56.35	30.56	0.79
	PTQD	0.01	75.43	26.14	0.60	-0.33	93.62	28.97	0.67	-0.01	77.12	32.43	0.62
	PTQ4DM	0.70	29.28	17.98	0.59	0.69	30.83	20.32	0.56	0.75	32.02	**21.84**	0.54
	Ours	**0.83**	**19.17**	17.68	**0.40**	0.66	34.02	**19.11**	**0.42**	0.72	40.77	24.73	**0.41**
SD3 -medium (28 Steps)	FP16	0.91	22.12	17.39	–	0.98	23.05	18.05	–	1.02	32.88	22.19	–
	I-BERT	-2.20	191.49	52.76	0.84	-2.19	176.40	54.51	0.83	-2.23	192.87	59.13	0.84
	I-ViT	-1.85	121.54	41.13	0.80	-1.75	102.19	41.16	0.78	-1.89	115.17	41.90	0.80
	Q-Diffusion	-1.90	117.35	45.90	0.82	-1.81	103.14	45.17	0.82	-1.92	122.47	45.54	0.81
	Q-DiT	-2.08	164.68	52.10	0.83	-2.02	145.22	52.88	0.83	-2.09	158.27	53.24	0.85
	TDQ	-1.54	49.10	25.80	0.79	-1.36	38.70	24.20	0.76	-1.55	50.68	27.62	0.77
	PTQD	-1.89	116.94	46.01	0.82	-1.81	103.98	45.83	0.82	-1.91	122.25	45.95	0.81
	PTQ4DM	-1.59	63.25	29.80	0.83	-1.39	52.99	28.93	0.79	-1.57	63.75	29.93	0.82
	Ours	**0.91**	**23.38**	**16.50**	**0.73**	**0.99**	**20.34**	**17.46**	**0.72**	**1.02**	**28.97**	**20.17**	**0.74**
FLUX.1 -schnell (4 Steps)	BF16	0.97	19.22	16.72	–	0.98	20.93	17.94	–	1.03	28.55	21.21	–
	I-BERT	0.56	26.20	35.47	0.60	0.76	26.78	26.17	0.60	0.79	32.56	37.31	0.62
	I-ViT	0.51	27.01	37.79	0.57	0.70	27.64	30.67	0.55	0.76	33.64	39.40	0.56
	Q-Diffusion	0.74	22.32	26.64	0.53	0.86	25.19	25.20	0.52	0.92	31.79	31.80	0.54
	Q-DiT	0.60	25.22	33.94	0.55	0.76	26.54	28.93	0.53	0.81	32.86	37.22	0.55
	TDQ	0.74	22.20	29.65	0.55	0.87	24.38	26.55	0.54	0.94	31.05	33.54	0.56
	PTQD	0.74	22.48	26.77	0.53	0.85	25.35	25.27	0.52	0.92	31.86	31.76	0.54
	PTQ4DM	0.64	24.11	33.10	0.54	0.78	25.75	28.61	0.52	0.83	32.58	36.15	0.54
	Ours	**0.90**	**19.12**	**18.73**	**0.44**	**0.96**	**22.37**	**19.80**	**0.46**	**1.02**	**29.39**	**24.39**	**0.47**

Table 2: Ablation study of proposed components using MJHQ dataset

Method	PixArt-Σ				SD3-medium			
	IR ↑	FID ↓	Lat. ↓	Mem. ↓	IR ↑	FID ↓	Lat. ↓	Mem. ↓
Baseline	0.26	50.40	41.48 s	15.94 GB	-1.85	121.54	76.92 s	28.48 GB
+ Int. Online Softmax	0.35	29.44	37.47 s	15.94 GB	-1.58	60.25	57.53 s	28.48 GB
+ Fine-grained Quant.	0.73	20.71	37.48 s	21.58 GB	0.71	30.80	59.60 s	34.09 GB
+ Dynafit	0.83	17.92	39.86 s	21.61 GB	0.91	23.38	61.95 s	34.10 GB

Ablation Study. Table 2 and Figure 6 assess the cumulative effects of I-DiT's three components. We measure latency and peak GPU memory for single-batch image generation on an A6000 GPU. Adding $\log_2$-domain integer online softmax lowers FID from 50.40 to 29.44 on PixArt-Σ and from 121.54 to 60.25 on SD3-medium while reducing latency. Fine-grained quantization further lowers FID to 20.71 and 30.80, showing that activation range mismatch persists after attention is stabilized. Finally, dynafit reaches FIDs of 17.92 and 23.38 with little added peak memory, showing that finer granularity alone cannot resolve runtime range variation.

Comparison on Quantization Error. Figure 5 shows the quantization error of two DiT layers across three denoising timesteps (early, middle, and late), using 256 samples per timestep. The red horizontal line denotes mean error across samples using statically calibrated scale, which is far higher than each sample's lowest achievable error from choosing its optimal quantization scales. On the other hand, the mean error using ours (green) shows comparable magnitude of error to each sample's minimum, successfully choosing the near-optimal scale for each sample.

Efficiency Comparison on Integer-Only Attention Table 3 compares integer tiled attention using our proposed method (§4.2) with FP-oriented and integer-only attention baselines. We report attention latency, peak memory, and full-denoiser latency. To isolate the effect of attention, we re-

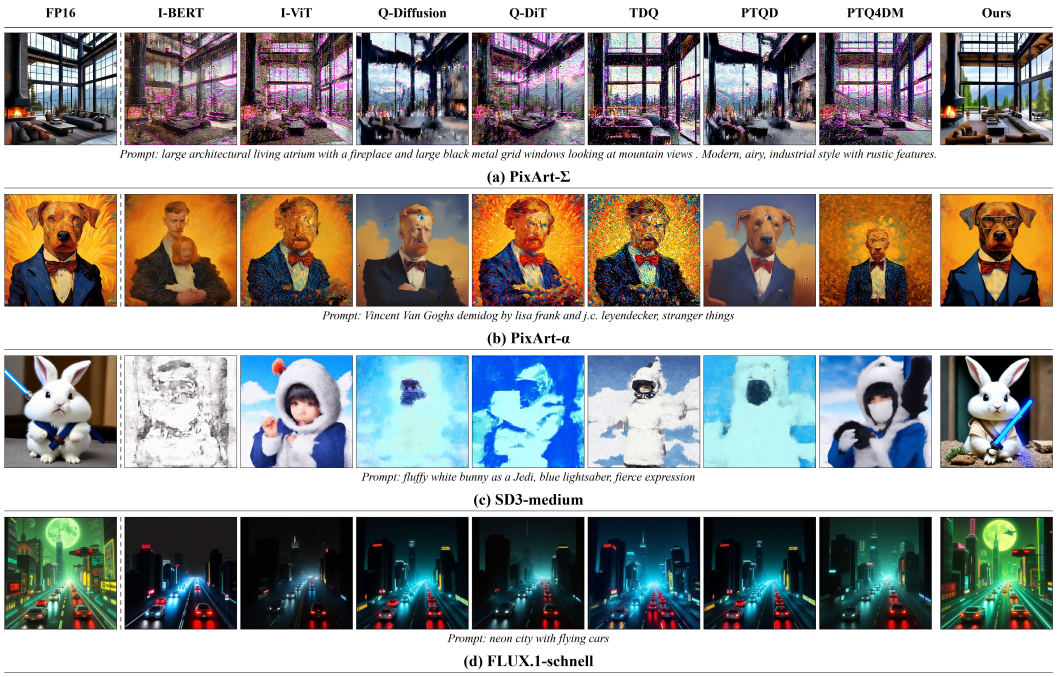


Figure 4: Qualitative evaluation of generated images using the MJHQ dataset.

Table 3: Latency and memory comparison across two models on A6000 GPU

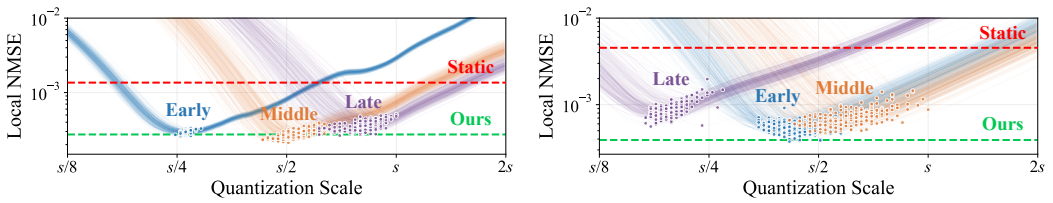
	Method	Online Softmax	PixArt-Σ Time (Attn)	PixArt-Σ Time (Model)	PixArt-Σ Peak Memory	SD3-medium Time (Attn)	SD3-medium Time (Model)	SD3-medium Peak Memory
FP-oriented	Naive PyTorch	✗	18.77 ms	1.42 s	2.12 GB	32.51 ms	2.20 s	3.64 GB
	FlashAttn	✓	2.18 ms	0.82 s	0.07 GB	2.73 ms	1.02 s	0.10 GB
	SageAttn	✓	10.93 ms	1.02 s	0.13 GB	6.68 ms	1.12 s	0.16 GB
	INT-FlashAttn	✓	3.29 ms	0.79 s	0.11 GB	3.68 ms	1.05 s	0.16 GB
	QAttn	✓	3.33 ms	0.88 s	0.12 GB	3.41 ms	1.14 s	0.16 GB
INT-only	I-BERT	✗	38.31 ms	1.83 s	6.21 GB	67.59 ms	2.53 s	10.88 GB
	I-ViT	✗	37.31 ms	1.79 s	4.23 GB	65.78 ms	2.54 s	7.38 GB
	FlashAttn (INT)	✓	385.42 ms	27.48 s	0.31 GB	670.99 ms	16.94 s	0.44 GB
	Ours	✓	34.17 ms	1.79 s	0.07 GB	28.75 ms	1.85 s	0.10 GB

place only the attention module while keeping other components unchanged. FP-oriented methods use tiling but retain floating-point operations, while I-BERT and I-ViT are integer-only but materialize the full attention matrix. FlashAttn (INT) is our implemented integer version of FlashAttention (Dao, 2024) that keeps tiled execution at the cost of wide fixed-point arithmetic and substantially higher latency (Appendix H). In contrast, $\log_2$-domain integer online softmax achieves 0.07/0.10 GB peak attention memory and $11.3\times$/$23.3\times$ lower attention latency than FlashAttn (INT) on PixArt-Σ/SD3-Medium. Its peak memory matches floating-point FlashAttention, showing that tiled memory efficiency can be retained under fully integer execution.

Hardware Implementation on FPGA. To assess feasibility on custom integer-only hardware, we implement I-DiT and three integer-only attention baselines in SystemVerilog as Vitis RTL kernels on an Alveo U280. For a controlled comparison, all designs use three SLR-pinned compute units, dedicated HBM pseudochannels through 512-bit AXI interfaces, common 140/350 MHz compute/memory-interface clocks, and approximately matched DSP allocations. For online softmax, ping-pong K/V buffers overlap transfers, QK, and normalization/PV under AXI backpressure, while I-BERT/I-ViT retain full-row softmax. Further details are in subsection J.1. Table 4 reports the latency, where I-DiT achieves 250.55ms latency on PixArt-Σ, showing $2.22\times$ speedup over FlashAttn (INT). It also requires substantially less on-chip memory than all compared designs. The results provide hardware validation that our integer-only attention formulation maps efficiently to a pure-integer datapath.

Table 4: Latency comparison on an FPGA (AMD U280) using PixArt-Σ

Method	Online Softmax	Latency	On-chip Mem.
FlashAttn (INT)	✓	557.34ms	1730.25KiB
I-ViT	✗	1053.80ms	866.25KiB
I-BERT	✗	1267.86ms	812.25KiB
Ours	✓	250.55ms	110.25KiB



(a) PixArt-Σ, block. 19. attn1 (pre-proj.) (b) SD3-medium, block. 20. ff_context (post-GeLU)

Figure 5: Quantization error curve within a layer across samples. Curves denote individual samples, the red and green lines denote mean error of all samples using static scale and dynafit, respectively.



Figure 6: Qualitative ablation of I-DiT on (a) PixArt-Σ and (b) SD3-medium. From left to right, we attach $\log_2$-domain integer online softmax, fine-grained quantization, and dynafit to the baseline.

6 Related Work

Integer-only quantization enables efficient inference on resource-constrained hardware by eliminating floating-point dependency (Gholami et al., 2021). Following the foundational integer-arithmetic framework (Jacob et al., 2018), HAWQ-V3 (Yao et al., 2021) introduced dyadic quantization to replace divisions with bit-shifts. For Transformers, subsequent works focused on integer-friendly nonlinear operations: Softermax (Stevens et al., 2021) proposed hardware-efficient softmax, I-BERT (Kim et al., 2021) employed integer polynomial approximations, and I-ViT (Li & Gu, 2023) introduced shift-based operations for ViTs. More recently, I-LLM (Hu et al., 2024) extended integer-only inference to LLMs with dynamic integer matrix multiplication for highly varying activations.

Diffusion quantization usually requires specialized remedies to handle non-stationary activations across timesteps. PTQD (He et al., 2023b), Q-Diffusion (Li et al., 2023a), PTQ4DM (Shang et al., 2023), TDQ (So et al., 2023), and TFMQ-DM (Huang et al., 2024) address this through timestep-aware calibration, scaling, or temporal consistency. For Transformer backbones, ViDiT-Q (Zhao et al., 2025) and Q-DiT (Chen et al., 2025a) use sensitivity-aware quantization, while QuEST (Wang et al., 2025) selectively tunes critical parameters. EfficientDM (He et al., 2023a) employs low-bit distillation, whereas MixDQ (Zhao et al., 2024) and SVDQuant (Li et al., 2025) use mixed precision or low-rank decomposition to mitigate sensitive activations and outliers. Nevertheless, these methods retain floating-point computation and do not support integer-only deployment.

7 Conclusion

In this work, we presented I-DiT, the first framework to enable integer-only quantization for diffusion transformers. First, we introduced dynafit, a dynamic scale adaptation that adjusts pre-calibrated scales at runtime without floating-point computation. We further reformulated online softmax in the $\log_2$ domain, enabling stable and memory-efficient tiled attention entirely with integer arithmetic. In addition, we support fine-grained quantization for activation without materializing element-wise scale tensors, preserving memory efficiency under dyadic constraints. Experiments demonstrate that I-DiT preserves generation quality while enabling efficient integer-only execution.

AI USE STATEMENT

In this work, we used generative AI tools for [implement methods, design or provide feedback on research methodology or experiments, interpret results, assist with translation]. We have not used generative AI tools for [help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, support qualitative and thematic data analysis], and [generate synthetic data sets, assist in the writing of proofs, propose or refine hypotheses, clean and reformat dataset] are not applicable to this work. Additionally, we used generative AI tools for [draft parts of a research paper, summarize or analyse existing literature, edit a research paper to improve readability, identify relevant literature]. We have reviewed all AI-assisted work. [We checked LLM-generated summary of prior works and verified that they were all existing works.] We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed explanation for each proposed method (section I), using mathematical formulas (section B), pseudocode (Algorithm 1), and triton kernel implementation (Listing 1). Appendix also includes implementation details on baselines (section H). Since our experiment includes FPGA implementation, we include the details in the Appendix (subsection J.1). The experiment setup is detailed in both the main body of the manuscript and the appendix (section A), including hyperparameter settings and bitwidth selection.

REFERENCES

Hamzah Abdelaziz, Jong Hoon Shin, Ardavan Pedram, Joseph Hassoun, et al. Rethinking floating point overheads for mixed precision dnn accelerators. *Proceedings of Machine Learning and Systems*, 3:223–239, 2021.

Kosmas Alexandridis and Giorgos Dimitrakopoulos. H-fa: A hybrid floating-point and logarithmic approach to hardware accelerated flashattention. *IEEE Transactions on Circuits and Systems for Artificial Intelligence*, 2025.

Arm. Ethos-U85: High-performance NPU for edge AI. `https://www.arm.com/products/silicon-ip-cpu/ethos/ethos-u85`, 2024. Accessed: 2026-09-17.

Arm. Scaling on-device AI across different Arm backends with ExecuTorch. `https://developer.arm.com/community/arm-community-blogs/b/ai-blog/posts/scaling-on-device-ai-across-different-arm-backends-with-executorch`, 2025. Accessed: 2026-09-17.

Michaela Blott, Thomas B Preußer, Nicholas J Fraser, Giulio Gambardella, Kenneth O'brien, Yaman Umuroglu, Miriam Leeser, and Kees Vissers. Finn-r: An end-to-end deep-learning framework for fast exploration of quantized neural networks. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 11(3):1–23, 2018.

Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis, 2023a.

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*, 2024.

Lei Chen, Yuan Meng, Chen Tang, Xinzhu Ma, Jingyan Jiang, Xin Wang, Zhi Wang, and Wenwu Zhu. Q-dit: Accurate post-training quantization for diffusion transformers. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 28306–28315, 2025a.

Ruichen Chen, Keith G Mills, and Di Niu. Fp4dit: Towards effective floating point quantization for diffusion transformers. *arXiv preprint arXiv:2503.15465*, 2025b.

Yu-Hui Chen, Raman Sarokin, Juhyun Lee, Jiuqiang Tang, Chuo-Ling Chang, Andrei Kulik, and Matthias Grundmann. Speed is all you need: On-device acceleration of large diffusion models via gpu-aware optimizations. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 4651–4655. IEEE, 2023b.

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

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 2022.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*, 2021.

Juan Gómez-Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F Oliveira, and Onur Mutlu. Benchmarking a new paradigm: Experimental analysis and characterization of a real processing-in-memory system. *IEEE Access*, 10:52565–52608, 2022.

Yefei He, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Efficientdm: Efficient quantization-aware fine-tuning of low-bit diffusion models. *arXiv preprint arXiv:2310.03270*, 2023a.

Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Ptqd: Accurate post-training quantization for diffusion models. *Advances in Neural Information Processing Systems*, 36:13237–13249, 2023b.

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

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

Xing Hu, Yuan Cheng, Dawei Yang, Zhihang Yuan, Jiangyong Yu, Chen Xu, and Sifan Zhou. I-llm: Efficient integer-only inference for fully-quantized low-bit large language models. *arXiv preprint arXiv:2405.17849*, 2024.

Yushi Huang, Ruihao Gong, Jing Liu, Tianlong Chen, and Xianglong Liu. Tfmq-dm: Temporal feature maintenance quantization for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2704–2713, 2018.

Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. I-bert: Integer-only bert quantization. In *International conference on machine learning*, pp. 5506–5518. PMLR, 2021.

Liangzhen Lai, Naveen Suda, and Vikas Chandra. Cmsis-nn: Efficient neural network kernels for arm cortex-m cpus. *arXiv preprint arXiv:1801.06601*, 2018.

Hongjae Lee, Myungjun Son, Dongjea Kang, and Seung-Won Jung. Text embedding knows how to quantize text-guided diffusion models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15426–15436. IEEE, 2025.

Daiqing Li, Aleks Kamko, Ehsan Akhgari, Ali Sabet, Linmiao Xu, and Suhail Doshi. Playground v2. 5: Three insights towards enhancing aesthetic quality in text-to-image generation. *arXiv preprint arXiv:2402.17245*, 2024.

Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. Svdquant: Absorbing outliers by low-rank components for 4-bit diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17535–17545, 2023a.

Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. *Advances in Neural Information Processing Systems*, 36:20662–20678, 2023b.

Zhikai Li and Qingyi Gu. I-vit: Integer-only quantization for efficient vision transformer inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17065–17075, 2023.

Cheng-Chen Lin, Wei Lu, Po-Tsang Huang, and Hung-Ming Chen. A 28nm 343.5 fps/w vision transformer accelerator with integer-only quantized attention block. In *2024 IEEE 6th International Conference on AI Circuits and Systems (AICAS)*, pp. 80–84. IEEE, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, 2014.

Keith G Mills, Mohammad Salameh, Ruichen Chen, Negar Hassanpour, Wei Lu, and Di Niu. Qua^{2}sedimo: Quantifiable quantization sensitivity of diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295*, 2021.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Qualcomm. Qualcomm Hexagon NPU. `https://www.qualcomm.com/processors/hexagon`. Accessed: 2026-09-17.

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.

Hyogon Ryu, NaHyeon Park, and Hyunjung Shim. Dgq: Distribution-aware group quantization for text-to-image diffusion models. In *International Conference on Learning Representations*, volume 2025, pp. 66857–66879, 2025.

Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1972–1981, 2023.

Cristina Silvano, Daniele Ielmini, Fabrizio Ferrandi, Leandro Fiorin, Serena Curzel, Luca Benini, Francesco Conti, Angelo Garofalo, Cristian Zambelli, Enrico Calore, et al. A survey on deep learning hardware accelerators for heterogeneous hpc platforms. *ACM Computing Surveys*, 57 (11):1–39, 2025.

Junhyuk So, Jungwon Lee, Daehyun Ahn, Hyungjun Kim, and Eunhyeok Park. Temporal dynamic quantization for diffusion models. *Advances in neural information processing systems*, 36:48686–48698, 2023.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020.

Jacob R Stevens, Rangharajan Venkatesan, Steve Dai, Brucek Khailany, and Anand Raghunathan. Softermax: Hardware/software co-design of an efficient softmax for transformers. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pp. 469–474. IEEE, 2021.

Yipeng Sun and Andreas M Kist. Deep learning on edge tpus. *arXiv preprint arXiv:2108.13732*, 2021.

Jack Urbanek, Florian Bordes, Pietro Astolfi, Mary Williamson, Vasu Sharma, and Adriana Romero-Soriano. A picture is worth more than 77 text tokens: Evaluating clip-style models on dense captions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26700–26709, 2024.

Haoxuan Wang, Yuzhang Shang, Zhihang Yuan, Junyi Wu, Junchi Yan, and Yan Yan. Quest: Low-bit diffusion model quantization via efficient selective finetuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15542–15551, 2025.

Maolin Wang, Seyedramin Rasoulinezhad, Philip HW Leong, and Hayden K-H So. Niti: Training integer neural networks using integer-only arithmetic. *IEEE Transactions on Parallel and Distributed Systems*, 33(11):3249–3261, 2022.

Zhewei Yao, Zhen Dong, Zhangcheng Zheng, Amir Gholami, Jiali Yu, Eric Tan, Leyuan Wang, Qijing Huang, Yida Wang, Michael Mahoney, et al. Hawq-v3: Dyadic neural network quantization. In *International Conference on Machine Learning*, pp. 11875–11886. PMLR, 2021.

Tianchen Zhao, Xuefei Ning, Tongcheng Fang, Enshu Liu, Guyue Huang, Zinan Lin, Shengen Yan, Guohao Dai, and Yu Wang. Mixdq: Memory-efficient few-step text-to-image diffusion models with metric-decoupled mixed precision quantization. In *European Conference on Computer Vision*, pp. 285–302. Springer, 2024.

Tianchen Zhao, Tongcheng Fang, Haofeng Huang, Enshu Liu, Rui Wan, Widyadewi Soedarmadji, Shiyao Li, Zinan Lin, Guohao Dai, Shengen Yan, et al. Vidit-q: Efficient and accurate quantization of diffusion transformers for image and video generation. *International Conference on Learning Representations*, 2025.

Yang Zhao, Yanwu Xu, Zhisheng Xiao, and Tingbo Hou. Mobilediffusion: Subsecond text-to-image generation on mobile devices. *arXiv preprint arXiv:2311.16567*, 2(3):4, 2023.

A Extended Experimental Settings

All experiments were conducted using A6000 GPUs with 48GB VRAM. Our implementation of I-DiT is built on the Hugging Face `diffusers` library by replacing its layers with quantized counterparts. We use classifier-free guidance (CFG) scales of 4.5 for PixArt-α and PixArt-Σ, 7.0 for SD3-medium, and 0 for FLUX.1-schnell, following their official document's default parameter setting. For SD3-medium and FLUX.1-schnell, we used the `cpu_offload` option to avoid out-of-memory errors. All diffusion quantization baselines and ours is built on top of I-ViT's nonlinear approximations. However, due to missing SiLU implementation we implement our own Int-SiLU using their integer GeLU: $\mathrm{SiLU}(x) = 1.702\,\mathrm{GELU}(x/1.702)$ Note that our method is orthogonal to the specific choice of the integer nonlinear approximations.

A.1 Datasets.

We use prompts from COCO Captions 2017 for calibration, and evaluate our method on three benchmark prompt sets. Two benchmarks with distinct characteristics, MJHQ-30K and Densely Captioned Images (DCI), are used to assess generalization across diverse prompt styles. Additionally, we evaluate on COCO Captions 2014, which has a distribution similar to the calibration set, following common settings used in diffusion quantization works.

- **COCO** (Lin et al., 2014). We use **COCO Captions 2017** for calibration. This dataset provides natural image captions derived from the MS-COCO benchmark and is widely used for evaluating text-to-image generation models. Following prior works, we randomly sample prompts from this dataset to construct the calibration set.
- **MJHQ-30K** (Li et al., 2024). MJHQ-30K consists of 30K samples collected from Midjourney, covering 10 common categories with 3K samples per category. The dataset is designed to evaluate artistic and stylized image generation. Following prior works, we uniformly sample 5K prompts from MJHQ-30K for benchmarking.
- **Densely Captioned Images (DCI)** (Urbanek et al., 2024). DCI is a dataset containing approximately 8K images paired with detailed human-annotated captions, where each caption is substantially longer and more descriptive than those in conventional captioning benchmarks. In our experiments, we use the summarized version (**sDCI**), where the original captions are condensed into shorter prompts (77 tokens) suitable for diffusion models. We randomly sample 5K prompts from this dataset for evaluation.

A.2 Baselines.

We compare I-DiT against two representative integer-only quantization methods and five diffusion quantization methods:

- **I-BERT** (Kim et al., 2021). I-BERT enables integer-only Transformer inference by approximating nonlinear functions with integer-friendly low-order polynomials. In particular, it reformulates *GELU* and the exponential term in *Softmax* using second-order polynomial approximations that can be evaluated with integer arithmetic, and computes *LayerNorm* using an integer square-root algorithm. In addition, I-BERT performs matrix multiplications with INT8 multiplication and INT32 accumulation, and requantizes the nonlinear outputs back to INT8. For attention, it follows the standard scaled dot-product attention pipeline and applies its quantized Softmax approximation after materializing the full attention matrix, rather than using an online softmax formulation.
- **I-ViT** (Li & Gu, 2023). I-ViT extends integer-only inference to Vision Transformers using bit-shift-based approximations for nonlinear functions. Specifically, it introduces *Shiftmax*, which approximates the exponential in Softmax through base conversion and shift operations, and *ShiftGELU*, which uses a shift-based approximation derived from the sigmoid form of GELU. For *LayerNorm*, I-ViT uses integer iterative square-root computation. The method adopts the simplest symmetric uniform quantization and targets INT8 integer-only inference. Similar to I-BERT, attention is computed with the standard softmax pipeline rather than an online softmax algorithm.

- **Q-Diffusion** (Li et al., 2023a). Q-Diffusion addresses timestep-dependent quantization in diffusion models by sampling calibration inputs from the full-precision denoising trajectory at uniformly spaced timesteps. It then performs block-wise reconstruction to optimize weight rounding and activation quantization scales. For U-Net skip connections, split shortcut quantization separately quantizes deep and shallow feature groups and their corresponding weight partitions.
- **Q-DiT** (Chen et al., 2025a). Q-DiT targets diffusion transformers with fine-grained group quantization and sample-wise dynamic activation quantization. It groups weights and activations along the input-channel dimension and selects layer-wise group sizes through evolutionary search under a bit-operation budget. During inference, each activation group computes its scale and zero point from its current minimum and maximum values. The original implementation uses asymmetric quantization and GPTQ for weights.
- **TDQ** (So et al., 2023). Temporal Dynamic Quantization (TDQ) adapts activation quantization to diffusion timesteps by replacing static intervals with timestep-conditioned ones. It encodes the timestep using Fourier features and feeds them to a lightweight MLP that predicts the quantization interval for each activation quantizer. The predictor can be optimized with quantization-aware training or reconstruction-based post-training calibration, and the resulting intervals can be precomputed for a fixed inference schedule.
- **PTQD** (He et al., 2023b). PTQD corrects quantization error in diffusion noise prediction by decomposing it into correlated and residual components. The correlated component is corrected by rescaling the quantized prediction using an estimated correlation coefficient, while the residual component is handled with channel-wise bias correction and variance schedule calibration. It also applies step-aware mixed precision, varying activation bit widths while keeping the weight bit width fixed across timesteps.
- **PTQ4DM** (Shang et al., 2023). PTQ4DM constructs calibration data to cover the multi-timestep diffusion process using Normally Distributed Time-step Calibration (NDTC). NDTC samples timesteps from a normal distribution biased toward later denoising steps and obtains the corresponding intermediate inputs from the full-precision model. Quantization parameters are selected using an MSE-based calibration criterion, while nonlinear operations such as SiLU and Softmax remain in full precision.

A.3 Evaluation Metrics.

We evaluate image quality using the following commonly used metrics:

- **FID**. Fréchet Inception Distance (FID) measures the distribution distance between generated images and real images using features extracted from a pretrained Inception network. It compares the mean and covariance of feature distributions. Lower values indicate that generated images are closer to the real image distribution.
- **IS**. Inception Score (IS) evaluates both the quality and diversity of generated images using the conditional label distribution predicted by an Inception classifier. High scores indicate images that are both semantically meaningful and diverse.
- **SFID**. Spatial Fréchet Inception Distance (SFID) extends FID by comparing spatial feature statistics rather than only global pooled features. This allows the metric to better capture structural differences and spatial artifacts in generated images. Lower values indicate better image fidelity.
- **CLIP Score**. CLIP Score measures text-image alignment by computing the cosine similarity between CLIP embeddings of the generated image and the input text prompt. Higher scores indicate stronger semantic alignment between the prompt and generated image.
- **CLIP-FID**. CLIP-FID replaces the Inception feature extractor used in FID with CLIP image embeddings. This provides a distributional comparison that is more aligned with semantic representations learned from large-scale vision-language training. Lower values indicate better similarity to real images.
- **CLIP-IQA**. CLIP-IQA is a no-reference image quality assessment metric derived from CLIP representations. It estimates perceptual image quality without requiring a reference

Algorithm 1 Integer Online Log_2-Softmax Attention

Require: $\tilde{Q}, \tilde{K}, \tilde{V}$; s_q, s_k, s_v; α; $Mask$; f_y, f_w; B_N ; F_{div}
Ensure: $\tilde{O}$
$s_S \leftarrow \frac{s_q s_k}{\sqrt{d}}, \quad \hat{s}_S \leftarrow DN(\frac{s_S}{\ln 2})$
$\tilde{S} \leftarrow \tilde{Q}\tilde{K}^\top$
$\hat{S} \leftarrow \hat{s}_S \odot \tilde{S} + Mask$
for each query row i **do**
  $\hat{m}_i \leftarrow -\infty, \quad \hat{l}_i \leftarrow 0, \quad \hat{o}_i \leftarrow 0$
  for each key tile j **do**
    $(\hat{m}_i, \hat{l}_i, \hat{o}_i) \leftarrow$
      $\mathrm{UpdateStats}(\hat{S}_i^j, \tilde{V}^j, \hat{m}_i, \hat{l}_i, \hat{o}_i)$
  end for
  $\hat{l}_i^{inv} \leftarrow \lfloor 2^{F_{\text{div}}} / \max(\hat{l}_i, 1) \rceil$
  $\tilde{O}_i \leftarrow \lfloor o_i \cdot \hat{l}_i^{inv} / 2^{F_{\text{div}}} \rceil$
end for
return $\tilde{O}$, s_v

Algorithm 2 UPDATESTATS (Single Tile)

Require: tile logits $S = \{S_k\}_{k=1}^{B_N}$; tile values $V = \{V_k\}_{k=1}^{B_N}$; current state (m, l, o)
Ensure: updated (m, l, o)
$m_{\text{tile}} \leftarrow \max_{1 \le k \le B_N} S_k$
$m_{\text{new}} \leftarrow \max(m, m_{\text{tile}})$
$r \leftarrow \mathrm{DyPow}(m_{\text{new}} - m)$
$l \leftarrow \lfloor l \cdot r / 2^{f_w} \rceil$
$o \leftarrow \lfloor o \cdot r / 2^{f_w} \rceil$
for $k = 1$ **to** B_N **do**
  $\omega_k \leftarrow \mathrm{DyPow}(m_{\text{new}} - S_k)$
end for
$l \leftarrow l + \sum_{k=1}^{B_N} \omega_k$
$o \leftarrow o + \sum_{k=1}^{B_N} \omega_k V_k$
$m \leftarrow m_{\text{new}}$
return (m, l, o)

image, correlating with human judgments of visual quality. Higher scores indicate better perceptual quality.

- **ImageReward**. ImageReward is a learned reward model trained on large-scale human preference data for text-to-image generation. It evaluates how well a generated image aligns with the given prompt and human aesthetic preferences. Higher scores indicate better text-image alignment and overall perceptual quality.
- **PSNR**. Peak Signal-to-Noise Ratio (PSNR) measures pixel-level reconstruction fidelity between a generated image and a reference image. It is computed from the mean squared error between the two images. Higher values indicate smaller reconstruction error.
- **LPIPS**. Learned Perceptual Image Patch Similarity (LPIPS) measures perceptual similarity between images using deep neural network feature representations. Unlike pixel-wise metrics, it correlates well with human perceptual similarity judgments. Lower values indicate higher perceptual similarity.
- **SSIM**. Structural Similarity Index Measure (SSIM) evaluates image similarity by comparing luminance, contrast, and structural information between two images. Higher values indicate greater structural similarity.

B DETAILED EXPLANATION OF LOG_2-DOMAIN INTEGER ONLINE SOFTMAX

To evaluate Equation (5) using integers only, we decompose x into its integer and fractional components: the integer component is evaluated exactly using a bit shift, while the bounded fractional component is approximated by a low-order integer polynomial. Consequently, DyPow requires only integer multiplication, addition, and bit shifts.

$$\mathrm{DyPow}(x) \approx 2^{f_w} 2^{-x}.$$

Base-2 exponential using fixed-point arithmetic. We define an integer primitive $\mathrm{DyPow}(x) \approx 2^{-x}$, which evaluates base-2 exponential using fixed-point arithmetic. We split x into an integer part n and fractional remainder f via f_y fractional bits:

$$x = n + \frac{f}{2^{f_y}}. \tag{16}$$

This yields $2^{-x} = 2^{-n} \cdot 2^{-f/2^{f_y}}$, where $f \in [0, 2^{f_y})$. Since 2^{-n} can be computed exactly using bit shift, we approximate the remaining fractional term. Let $t = f/2^{f_y}$, which lies in the bounded interval $t \in [0, 1)$. Using quadratic polynomial:

$$2^{-t} \approx 1 - At + Bt^2. \tag{17}$$

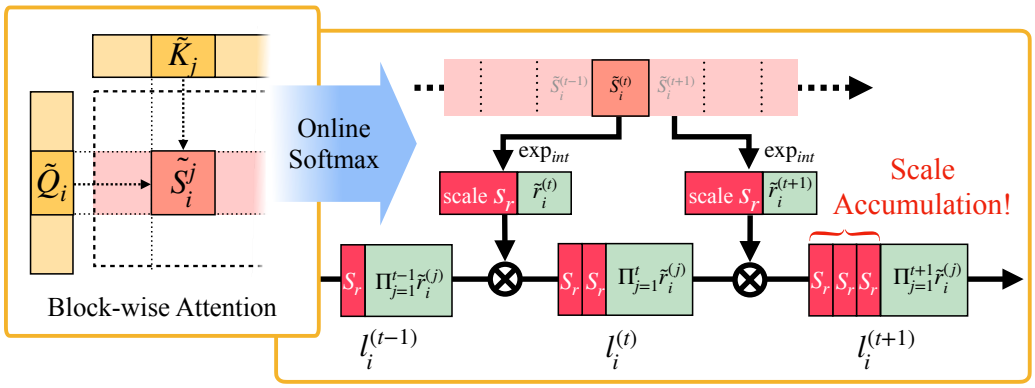


Figure 7: Visualization of online softmax using integer exponential ($\exp_{int}$). The scale output s_r from $\exp_{int}$ gets accumulated after processing each tile, eventually causing underflow and numerical instability in the final row wise sum l_i.

However, since $0 < 2^{-t} \leq 1$ and cannot be represented as an integer, we scale the approximation by f_w bits to preserve precision:

$$2^{-t} \cdot 2^{f_w} \approx 2^{f_w} - (A \cdot 2^{f_w})t + (B \cdot 2^{f_w})t^2, \tag{18}$$

where (A, B) are precomputed coefficients which we set as $A = 0.6711, B = 0.1711$. Let $a_0 = 2^{f_w}$, $a_1 = \lfloor A2^{f_w} \rceil$, and $a_2 = \lfloor B2^{f_w} \rceil$. Substituting $t = f/2^{f_y}$ and evaluating the polynomial using integer arithmetic gives

$$F(f) = a_0 - (a_1 f) \gg f_y + (a_2 f^2) \gg (2f_y), \tag{19}$$

This produces an integer approximation of $2^{f_w}2^{-t}$. Combining this result with the shift 2^{-n} yields the final implementation

$$\mathrm{DyPow}(x) = F(f) \gg n. \tag{20}$$

After processing all tiles, the final attention output $\tilde{O}_i$ is obtained by dividing the output accumulator by the normalization term using a fixed-point reciprocal with precision F_{div}:

$$\hat{\ell}_i^{\mathrm{inv}} = \left\lfloor 2^{F_{\mathrm{div}}} \,/\, \max\left(\hat{\ell}_i,\, 1\right) \right\rceil, \tag{21}$$

$$\tilde{O}_i = \left\lfloor \hat{o}_i \cdot \hat{\ell}_i^{\mathrm{inv}} \,/\, 2^{F_{\mathrm{div}}} \right\rceil. \tag{22}$$

The $\max(\cdot, 1)$ guard prevents division by zero in degenerate tiles.

Triton Kernel Implementation. To ensure practical efficiency, we implement the proposed integer online softmax attention as a custom GPU kernel using Triton. The entire forward pass is fused into a single Triton GPU kernel where the arithmetic consists exclusively of `int8×int8→int32` Tensor Core dot products, integer additions and multiplications, and bit-shifts.

C Extended Motivational Studies

Figure 7 visualizes the scale accumulation issue mentioned in section 3 in detail. Baseline integer-only methods implement softmax by approximating the exponential function using an integer operator ($\exp_{int}$). However, these methods were originally designed for relatively smaller transformer models such as BERT and ViT, where the attention computation can be performed without memory-reduction techniques such as tile-wise attention. In contrast, modern diffusion models require tile-wise attention with online softmax to control memory usage during large-scale attention computation. When the integer exponential approximation ($\exp_{int}$) used in prior work is directly combined with the online softmax formulation required by tile-wise attention, the repeated rescaling steps introduce severe numerical instability, leading to a scale vanishing problem. As illustrated in Figure 7,

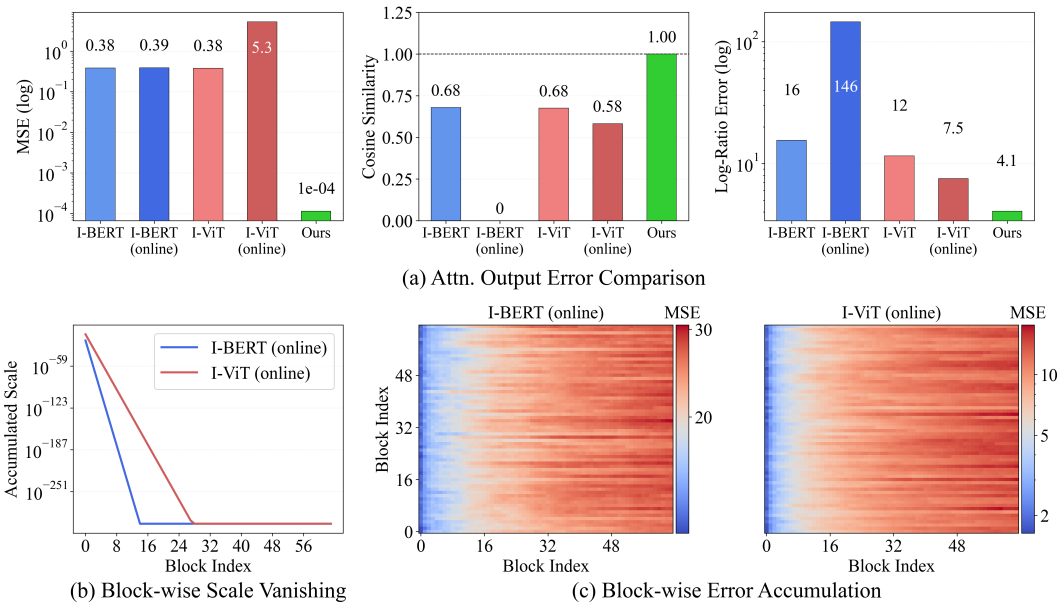


Figure 8: The first row compares the attention output error across different integer attention variants, measured by MSE(↓), cosine similarity(↑), and log-ratio error(↓). In the second row, the left plot illustrates the vanishing of the quantization scale as online softmax processes each attention tile, while the two heatmaps visualize how the error accumulates across tiles when using online softmax variants.

each tile processed by $\exp_{int}$ introduces a new quantization scale s_r. When $s_r < 1$, these scale factors accumulate multiplicatively across T tiles, causing $(s_r)^T$ to rapidly approach zero, which leads to severe attenuation of the attention scores.

We conduct additional experiments to demonstrate how scale accumulation introduces numerical errors in the attention outputs, which is demonstrated in Figure 8. The first three bar plots compare integer attention output error across five variants: I-BERT, I-BERT (online), I-ViT, I-ViT (online), and Ours. I-BERT (online) and I-ViT (online) are implemented by directly plugging their proposed integer exponential ($\exp_{int}$) into the online softmax algorithm for tile-wise attention. The three plots each measure MSE, cosine similarity, and log-ratio error between the integer attention output and a full-precision FP32 reference. Notice how the online version of the baseline methods increase the error and lower the cosine similarity. I-BERT (online) shows lower degradation in terms of error magnitude (MSE), but the cosine similarity degrades to 0 and the log-ratio error increases more than 9×. Similarly, while I-ViT's degradation is not captured through log-ratio error, but both the MSE and cosine similarity show increased error. On the other hand, our proposed method achieves an MSE three orders of magnitude lower than all baselines and near-perfect cosine similarity (0.9999), indicating that it closely approximates the floating-point baseline in both magnitude and direction.

Fig. 8b reveals the root cause of increased error in online variants: the accumulated normalization scale factor in both online variants decays exponentially toward zero as tile index increases. This is due to repetitive multiplication of $\exp_{int}$ necessary for renormalization after processing each tile of QK^T. Fig. 8c visualizes the resulting tile-wise error accumulation as heatmaps of MSE between the integer and FP32 online softmax outputs after processing each tile. The I-ViT (online) heatmap shows a clear left-to-right gradient of increasing error, confirming that quantization error compounds progressively across tiles. The I-BERT (online) heatmap exhibits uniformly high raw MSE, reflecting its more severe and immediate collapse. Our method avoids this failure mode entirely by eliminating the need for iterative scale correction in the softmax computation.

Table 5: Quality and similarity comparison on the MJHQ dataset

Model	Method	Quality							Similarity		
		FID↓	IS↑	sFID↓	CLIP↑	IR↑	C-FID↓	C-IQA↑	LPIPS↓	PSNR↑	SSIM↑
PixArt-Σ (20 Steps)	FP16	16.58	22.02	14.38	28.87	0.95	4.16	0.94	–	–	–
	I-BERT	41.96	16.13	20.67	26.35	0.04	11.15	0.64	0.60	14.01	0.48
	I-ViT	50.40	15.09	22.53	26.56	0.26	11.50	0.77	0.61	13.42	0.47
	Q-Diffusion	94.21	8.66	37.98	22.73	-0.80	28.18	0.31	0.70	11.71	0.44
	Q-DiT	51.02	14.47	25.23	26.61	0.24	12.05	0.80	0.61	13.17	0.45
	TDQ	102.41	10.98	32.83	23.98	-0.64	34.04	0.51	0.74	10.21	0.31
	PTQD	94.44	8.90	38.39	22.96	-0.74	27.87	0.33	0.70	11.36	0.43
	PTQ4DM	57.25	13.90	23.09	25.70	0.02	15.38	0.75	0.66	12.24	0.40
	Ours	17.92	21.39	15.79	28.17	0.83	5.01	0.91	0.49	13.96	0.54
PixArt-α (20 Steps)	FP16	16.03	21.38	15.09	28.56	0.99	4.35	0.94	–	–	–
	I-BERT	22.91	19.81	19.82	27.66	0.53	4.98	0.89	0.48	15.19	0.54
	I-ViT	26.37	21.58	17.47	27.76	0.78	5.39	0.90	0.54	14.90	0.50
	Q-Diffusion	75.98	11.31	28.10	26.16	-0.09	17.49	0.51	0.61	14.78	0.54
	Q-DiT	26.70	20.89	16.08	27.16	0.65	6.98	0.85	0.62	13.41	0.46
	TDQ	52.48	18.12	25.16	27.00	0.59	13.19	0.85	0.74	13.46	0.35
	PTQD	75.43	11.37	26.14	26.34	0.01	15.91	0.54	0.60	14.83	0.53
	PTQ4DM	29.28	21.59	17.98	27.64	0.70	6.21	0.88	0.59	14.46	0.48
	Ours	19.17	21.35	17.68	28.22	0.83	5.41	0.91	0.40	16.35	0.61
SD3 -medium (28 Steps)	FP16	22.12	27.68	17.39	28.31	0.91	6.93	0.93	–	–	–
	I-BERT	191.49	7.02	52.76	13.71	-2.20	58.19	0.45	0.84	7.05	0.22
	I-ViT	121.54	8.54	41.13	16.45	-1.85	33.47	0.55	0.80	9.03	0.34
	Q-Diffusion	117.35	8.07	45.90	17.32	-1.90	40.58	0.40	0.82	8.07	0.35
	Q-DiT	164.68	6.04	52.10	15.41	-2.08	47.74	0.44	0.83	8.72	0.29
	TDQ	49.10	16.64	25.80	17.01	-1.54	17.35	0.74	0.79	9.45	0.37
	PTQD	116.94	8.15	46.01	17.29	-1.89	39.71	0.41	0.82	8.20	0.37
	PTQ4DM	63.25	14.28	29.80	17.29	-1.59	21.14	0.60	0.83	8.87	0.32
	Ours	23.38	27.47	16.50	28.19	0.91	6.99	0.93	0.73	8.33	0.30
FLUX.1 -schnell (4 Steps)	BF16	19.22	22.13	16.72	28.71	0.97	5.43	0.94	–	–	–
	I-BERT	26.20	21.96	35.47	27.07	0.56	8.92	0.92	0.60	11.95	0.50
	I-ViT	27.01	22.41	37.79	26.99	0.51	8.04	0.90	0.57	12.87	0.52
	Q-Diffusion	22.32	22.52	26.64	27.74	0.74	6.14	0.92	0.53	13.56	0.55
	Q-DiT	25.22	22.59	33.94	27.24	0.60	7.46	0.91	0.55	13.14	0.53
	TDQ	22.20	22.78	29.65	27.73	0.74	6.28	0.91	0.55	13.04	0.53
	PTQD	22.48	22.23	26.77	27.75	0.74	6.24	0.91	0.53	13.56	0.55
	PTQ4DM	24.11	23.44	33.10	27.29	0.64	7.20	0.91	0.54	13.26	0.54
	Ours	19.12	23.75	18.73	28.14	0.90	5.19	0.94	0.44	14.75	0.60

D Extended Quantitative Analysis

We provide extended quantitative results on MJHQ, sDCI, and COCO in Tables 5 to 7, covering a broader set of quality and similarity metrics. In addition to the metrics reported in the main paper (ImageReward, FID, sFID, and LPIPS), we further evaluate Inception Score (IS), CLIP, CLIP-FID (C-FID), CLIP-IQA (C-IQA), PSNR, and SSIM. Overall, our method consistently preserves the full-precision outputs more faithfully than existing integer-only baselines, while maintaining competitive generation quality across different model architectures and datasets.

Our method consistently maintains strong perceptual quality across different models and datasets. On PixArt-Σ with MJHQ, for instance, C-IQA improves from 0.64 for I-BERT and 0.77 for I-ViT to 0.91, while ImageReward increases from 0.04 and 0.26 to 0.83. On SD3-medium, where conventional integer-only baselines exhibit substantial quality degradation, the difference becomes more pronounced. On MJHQ, I-BERT and I-ViT obtain ImageReward scores of -2.20 and -1.85, respectively, whereas our method achieves 0.91. Our method also reduces C-FID to 6.99, compared with 58.19 for I-BERT and 33.47 for I-ViT.

Table 6: Quality and similarity comparison on the sDCI dataset

Model	Method	Quality							Similarity		
		FID↓	IS↑	sFID↓	CLIP↑	IR↑	C-FID↓	C-IQA↑	LPIPS↓	PSNR↑	SSIM↑
PixArt-Σ (20 Steps)	FP16	24.85	20.45	17.19	26.39	0.96	12.63	0.97	–	–	–
	I-BERT	47.51	13.71	22.31	25.15	0.14	24.28	0.66	0.62	13.41	0.42
	I-ViT	49.60	12.88	27.95	25.25	0.37	21.51	0.78	0.64	12.66	0.40
	Q-Diffusion	96.66	7.06	40.78	22.31	-0.75	42.07	0.32	0.73	11.23	0.37
	Q-DiT	49.98	12.82	31.72	25.37	0.34	23.23	0.83	0.63	12.46	0.38
	TDQ	118.43	9.09	40.69	23.52	-0.85	50.38	0.47	0.76	9.41	0.26
	PTQD	97.19	7.11	41.69	22.45	-0.68	41.33	0.34	0.73	10.95	0.36
	PTQ4DM	59.33	11.73	28.06	24.63	0.04	26.50	0.75	0.69	11.46	0.34
	Ours	27.07	19.39	17.27	26.34	0.83	13.54	0.93	0.52	13.05	0.47
PixArt-α (20 Steps)	FP16	27.51	21.00	17.57	26.02	0.83	16.44	0.97	–	–	–
	I-BERT	31.69	17.34	20.52	25.51	0.43	18.44	0.93	0.49	14.60	0.50
	I-ViT	29.39	17.98	19.32	25.85	0.72	14.35	0.96	0.51	14.47	0.49
	Q-Diffusion	96.02	8.63	30.31	23.98	-0.44	38.61	0.50	0.68	14.28	0.49
	Q-DiT	27.97	17.38	19.49	25.42	0.66	15.31	0.94	0.59	13.10	0.45
	TDQ	62.28	13.41	32.72	25.30	0.44	26.27	0.90	0.81	12.08	0.27
	PTQD	93.62	8.54	28.97	24.02	-0.33	36.63	0.53	0.67	14.38	0.49
	PTQ4DM	30.83	17.88	20.32	25.91	0.69	14.77	0.95	0.56	14.28	0.48
	Ours	34.02	19.69	19.11	25.96	0.66	18.37	0.95	0.42	15.57	0.56
SD3 -medium (28 Steps)	FP16	23.05	23.22	18.05	26.48	0.98	6.10	0.92	–	–	–
	I-BERT	176.40	6.81	54.51	15.47	-2.19	61.75	0.43	0.83	7.80	0.24
	I-ViT	102.19	8.30	41.16	18.92	-1.75	33.78	0.53	0.78	9.80	0.35
	Q-Diffusion	103.14	7.83	45.17	19.60	-1.81	44.52	0.45	0.82	8.11	0.33
	Q-DiT	145.22	5.95	52.88	18.00	-2.02	52.06	0.43	0.83	9.32	0.28
	TDQ	38.70	14.74	24.20	19.58	-1.36	15.22	0.74	0.76	9.82	0.34
	PTQD	103.98	7.95	45.83	19.47	-1.81	44.03	0.46	0.82	8.21	0.34
	PTQ4DM	52.99	13.42	28.93	20.02	-1.39	19.86	0.62	0.79	9.80	0.33
	Ours	20.34	23.07	17.46	26.26	0.99	5.36	0.92	0.72	8.75	0.24
FLUX.1 -schnell (4 Steps)	BF16	20.93	20.79	17.94	26.50	0.98	5.54	0.93	–	–	–
	I-BERT	26.78	19.29	26.17	26.45	0.76	7.81	0.93	0.60	11.50	0.50
	I-ViT	27.64	19.56	30.67	26.20	0.70	7.79	0.91	0.55	12.43	0.51
	Q-Diffusion	25.19	19.94	25.20	26.33	0.86	6.87	0.92	0.52	12.85	0.52
	Q-DiT	26.54	20.04	28.93	26.27	0.76	7.32	0.91	0.53	12.62	0.52
	TDQ	24.38	20.48	26.55	26.35	0.87	6.58	0.91	0.54	12.43	0.51
	PTQD	25.35	20.19	25.27	26.33	0.85	6.94	0.92	0.52	12.85	0.52
	PTQ4DM	25.75	20.19	28.61	26.28	0.78	7.00	0.91	0.52	12.75	0.52
	Ours	22.37	20.75	19.80	26.32	0.96	6.03	0.93	0.46	13.67	0.54

Beyond generation quality, our method shows particularly consistent improvements in similarity to the full-precision outputs. Across all three datasets, our method achieves the lowest LPIPS among the quantized methods for PixArt-Σ, PixArt-α, SD3-medium, and FLUX.1-schnell. For example, on PixArt-α, our method reduces LPIPS to 0.40, 0.42, and 0.41 on MJHQ, sDCI, and COCO, respectively, compared with the best competing results of 0.48, 0.49, and 0.49. A similar trend is observed for PSNR, where our method achieves 16.35, 15.57, and 15.58 on the same three datasets, consistently exceeding the other integer-only baselines. These results indicate that our quantization scheme better preserves the outputs of the full-precision model throughout the denoising process.

The extended results also reveal that the advantage of our method is not limited to a single quality metric. For PixArt-α, where several distribution-based metrics such as FID and C-FID are less favorable in some settings, our method still maintains strong semantic and perceptual quality while providing substantially better similarity to the full-precision outputs. For example, on COCO, our method achieves a CLIP score of 26.11 and C-IQA of 0.96, while simultaneously obtaining the best LPIPS and PSNR among all quantized methods. Taken together, the results across the ten metrics show that our method provides a favorable balance between preserving generation quality and maintaining high fidelity to the full-precision model under integer-only inference.

Table 7: Quality and similarity comparison on the COCO dataset

Model	Method	Quality							Similarity		
		FID↓	IS↑	sFID↓	CLIP↑	IR↑	C-FID↓	C-IQA↑	LPIPS↓	PSNR↑	SSIM↑
PixArt-Σ (20 Steps)	FP16	34.34	33.30	19.04	26.46	0.93	20.40	0.97	–	–	–
	I-BERT	44.45	23.84	22.92	25.30	0.25	20.11	0.72	0.62	13.33	0.50
	I-ViT	51.03	22.18	26.03	25.43	0.50	18.85	0.81	0.64	12.70	0.46
	Q-Diffusion	93.75	12.54	36.94	22.88	-0.66	33.23	0.31	0.72	11.05	0.44
	Q-DiT	53.34	22.27	28.30	25.45	0.42	20.42	0.83	0.64	12.37	0.44
	TDQ	119.09	13.51	38.02	24.04	-0.65	41.97	0.49	0.76	9.82	0.30
	PTQD	92.81	12.95	37.60	22.92	-0.57	32.35	0.33	0.72	10.74	0.43
	PTQ4DM	59.57	19.32	26.62	24.79	0.16	21.97	0.77	0.69	11.66	0.38
	Ours	33.07	32.77	19.40	26.47	0.79	17.32	0.94	0.53	12.84	0.53
PixArt-α (20 Steps)	FP16	37.73	33.40	22.50	26.04	0.85	24.32	0.97	–	–	–
	I-BERT	35.15	31.07	29.05	25.72	0.51	23.36	0.94	0.49	14.41	0.55
	I-ViT	31.94	33.60	22.26	26.11	0.77	19.89	0.95	0.50	14.29	0.54
	Q-Diffusion	78.54	17.69	34.80	24.75	-0.10	33.86	0.59	0.63	13.92	0.54
	Q-DiT	30.80	31.47	22.01	26.01	0.67	19.75	0.93	0.58	12.49	0.49
	TDQ	56.35	24.87	30.56	25.90	0.60	25.18	0.89	0.79	12.39	0.29
	PTQD	77.12	18.18	32.43	24.73	-0.01	33.24	0.61	0.62	13.85	0.53
	PTQ4DM	32.02	32.75	21.84	26.15	0.75	19.77	0.95	0.54	14.16	0.53
	Ours	40.77	31.52	24.73	26.11	0.72	24.74	0.96	0.41	15.58	0.61
SD3 -medium (28 Steps)	FP16	32.88	36.03	22.19	26.67	1.02	13.09	0.93	–	–	–
	I-BERT	192.87	6.80	59.13	14.70	-2.23	64.13	0.39	0.84	7.91	0.27
	I-ViT	115.17	11.91	41.90	18.02	-1.89	34.49	0.48	0.80	9.48	0.39
	Q-Diffusion	122.47	9.57	45.54	19.04	-1.92	43.12	0.40	0.81	8.50	0.40
	Q-DiT	158.27	7.77	53.24	17.01	-2.09	53.13	0.35	0.85	8.87	0.31
	TDQ	50.68	20.90	27.62	18.77	-1.55	16.60	0.72	0.77	9.74	0.40
	PTQD	122.25	9.76	45.95	19.08	-1.91	42.79	0.40	0.81	8.61	0.41
	PTQ4DM	63.75	19.09	29.93	19.39	-1.57	19.68	0.58	0.82	9.33	0.36
	Ours	28.97	37.38	20.17	26.52	1.02	10.85	0.92	0.74	8.44	0.31
FLUX.1 -schnell (4 Steps)	BF16	28.55	34.77	21.21	26.58	1.03	13.95	0.94	–	–	–
	I-BERT	32.56	31.18	37.31	26.80	0.79	13.18	0.93	0.62	10.88	0.53
	I-ViT	33.64	32.45	39.40	26.40	0.76	13.23	0.92	0.56	12.07	0.56
	Q-Diffusion	31.79	34.33	31.80	26.50	0.92	13.58	0.93	0.54	12.60	0.57
	Q-DiT	32.86	32.41	37.22	26.46	0.81	13.20	0.93	0.55	12.30	0.57
	TDQ	31.05	35.17	33.54	26.51	0.94	13.19	0.93	0.56	12.15	0.55
	PTQD	31.86	34.15	31.76	26.49	0.92	13.78	0.93	0.54	12.60	0.57
	PTQ4DM	32.58	33.55	36.15	26.42	0.83	13.33	0.93	0.54	12.46	0.57
	Ours	29.39	35.26	24.39	26.40	1.02	13.57	0.94	0.47	13.66	0.59

E Additional Qualitative Analysis

We provide additional qualitative comparisons in Figures 9 to 12 using multiple prompts sampled from the MJHQ dataset. For each prompt, we compare images generated by full-precision inference, the integer-only baselines I-BERT and I-ViT, five diffusion quantization baselines (Q-Diffusion, Q-DiT, TDQ, PTQD, and PTQ4DM), and our method. Across all four models, the compared quantization methods often introduce substantial deviations from the corresponding full-precision outputs, including changes in visual structure, object appearance, texture, and color composition. In contrast, our method consistently produces images that more closely resemble the full-precision outputs while preserving overall visual quality and semantic content.

For PixArt-Σ and PixArt-α, I-BERT and I-ViT as well as the diffusion quantization baselines generally retain the high-level prompt semantics, but their outputs frequently exhibit noticeable changes in fine-grained appearance and image composition compared with the full-precision references. Some methods additionally exhibit pronounced visual artifacts or structural degradation for particular prompts. Our method better preserves these visual characteristics and produces outputs that remain perceptually closer to the corresponding full-precision generations. This observation is con-

sistent with the quantitative similarity results, where our method achieves substantially lower LPIPS and higher PSNR across both PixArt models.

The difference becomes more pronounced on SD3-medium, which is particularly sensitive to integer-only quantization. The compared baselines frequently produce severely distorted or semantically degraded outputs, with substantial changes in object structure, texture, and overall scene appearance. In comparison, our method largely preserves the content and visual structure of the full-precision generations across diverse prompts. These qualitative results are consistent with the large improvements observed in the quantitative quality metrics for SD3-medium and further demonstrate the robustness of our method under challenging integer-only inference settings.

Overall, the qualitative comparisons complement the quantitative evaluation by showing that the improvements in similarity metrics correspond to meaningful preservation of the full-precision generation characteristics. Our method effectively reduces quantization-induced visual degradation while maintaining both perceptual quality and semantic consistency across different diffusion transformer architectures.

Table 8: Full ablation study of proposed components on the MJHQ dataset

Model	Method	Dynafit (§4.1)	Int. Log_2 (§4.2)	Fine-grained (§4.3)	IR ↑	FID ↓	Latency ↓	Peak Mem. ↓
PixArt-Σ	Baseline	✗	✗	✗	0.26	50.40	41.48 s	15.94 GB
		✓	✗	✗	0.24	52.12	73.08 s	15.94 GB
		✗	✓	✗	0.35	29.44	37.47 s	15.94 GB
		✗	✗	✓	0.11	55.41	44.61 s	21.58 GB
		✓	✓	✗	-0.01	63.45	46.27 s	15.94 GB
		✗	✓	✓	0.73	20.71	37.48 s	21.58 GB
		✓	✗	✓	0.45	34.65	68.91 s	21.61 GB
	Ours	✓	✓	✓	0.83	17.92	39.86 s	21.61 GB
SD3-medium	Baseline	✗	✗	✗	-1.85	121.54	76.92 s	28.48 GB
		✓	✗	✗	-1.73	53.18	134.37 s	31.05 GB
		✗	✓	✗	-1.58	60.25	57.53 s	28.48 GB
		✗	✗	✓	0.65	33.81	83.26 s	34.09 GB
		✓	✓	✗	-1.62	70.92	72.77 s	28.48 GB
		✗	✓	✓	0.71	30.80	59.60 s	34.09 GB
		✓	✗	✓	0.86	24.01	123.02 s	36.68 GB
	Ours	✓	✓	✓	0.91	23.38	61.95 s	34.10 GB

F Additional Ablation Studies

We provide a full ablation study in Table 8 to further analyze the individual and combined effects of our three proposed components: dynafit (§4.1), integer-only $\log_2$-domain online softmax (§4.2), and fine-grained quantization (§4.3). We evaluate all combinations of the three components on PixArt-Σ and SD3-medium using the MJHQ dataset, reporting both generation quality and inference efficiency.

The results show that the three components provide complementary benefits in all settings. Log_2-domain integer online softmax improves both generation quality and latency when applied individually, while dynafit and fine-grained quantization become particularly effective when combined with the other components. For example, adding fine-grained quantization to $\log_2$-domain integer online softmax improves FID from 29.44 to 20.71 on PixArt-Σ and from 60.25 to 30.80 on SD3-medium. With all three components enabled, FID further improves to 17.92 and 23.38, respectively, while IR reaches 0.83 and 0.91. Overall, the full configuration achieves the strongest generation quality on both models, demonstrating that the proposed components complement one another in mitigating different sources of error in integer-only inference.

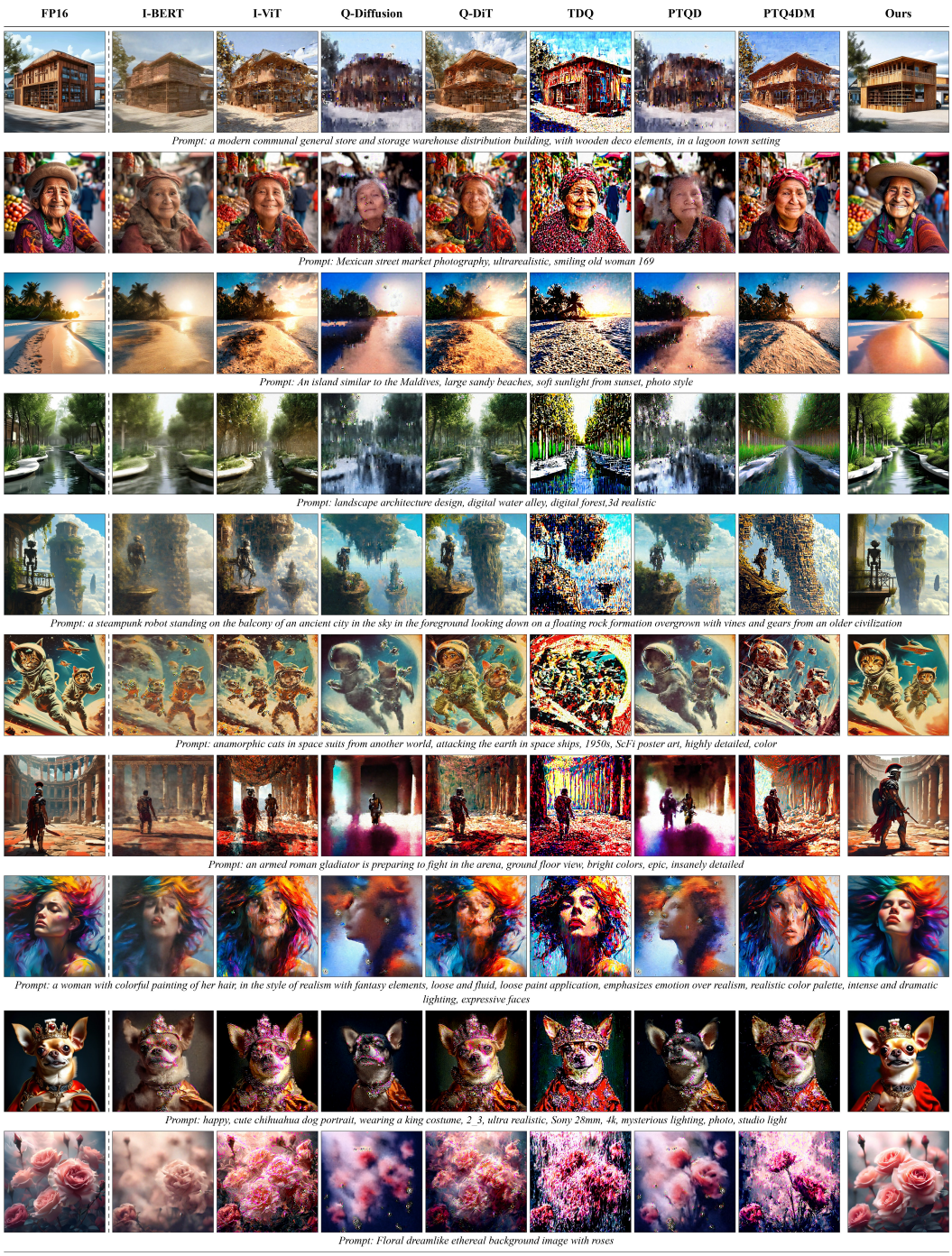


Figure 9: Additional qualitative evaluation of generated images using PixArt-Σ on MJHQ dataset. Compared to other baselines, our method produces images with noticeably improved visual quality and closer semantic alignment to the FP16 results, which is consistent with the quantitative improvements in both image quality and similarity to FP16 outputs.

G Additional Analysis Results on Quantization Error

In this section we provide additional analysis results (Fig. 15) on the quantization error curve using 256 samples from 3 different timesteps. Each plot shows different layer in the model architecture,

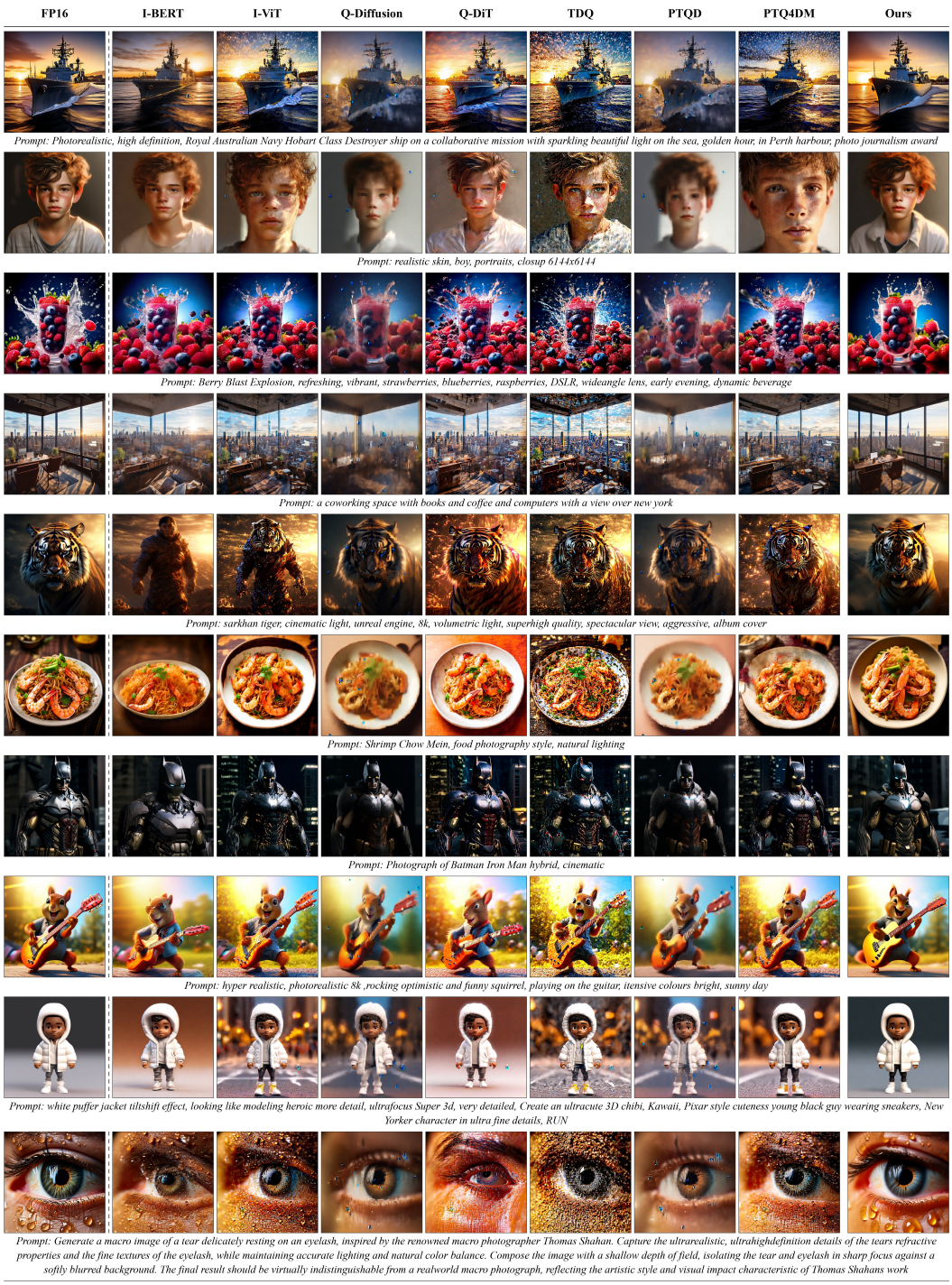


Figure 10: Additional qualitative evaluation of generated images using PixArt-α on MJHQ dataset. Compared to integer-only baselines, our method produces images with noticeably improved visual quality and closer semantic alignment to the FP16 results, which is consistent with the quantitative improvements in both image quality and similarity to FP16 outputs.

with each curve drawing the error of a single sample. Red horizontal line denotes the global mean error of all samples using pre-calibrated static scale, while the green horizontal line denotes the same global mean error but using our method.

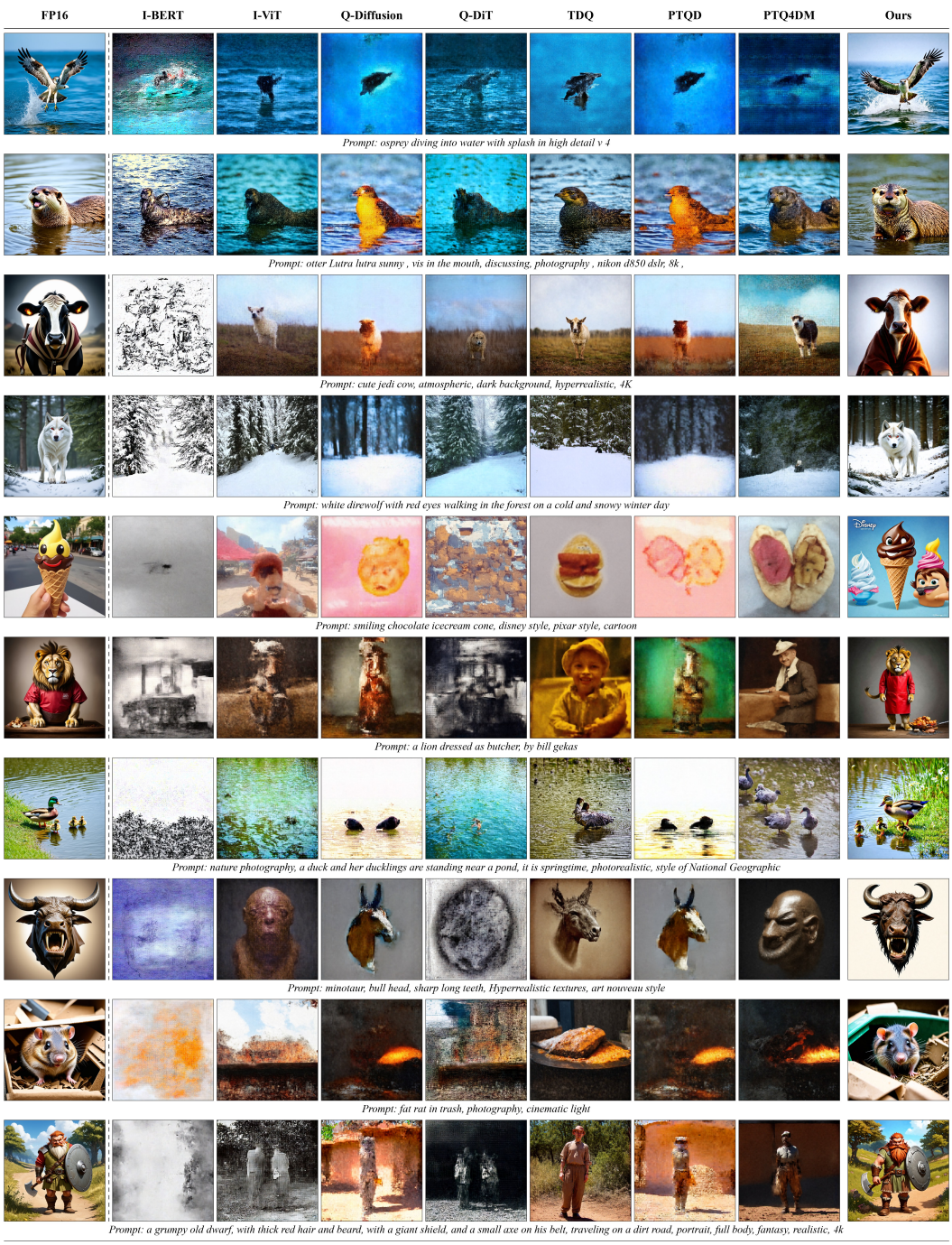


Figure 11: Additional qualitative evaluation of generated images using SD3-medium on MJHQ dataset. Compared to integer-only baselines, our method produces images with noticeably improved visual quality and closer semantic alignment to the FP16 results, which is consistent with the quantitative improvements in both image quality and similarity to FP16 outputs.

H Implementation Details of Integer FlashAttention

We implement a fixed-point baseline that preserves the tiled online-softmax structure of FlashAttention (Dao et al., 2022), but not its original floating-point arithmetic or optimized GEMM. Instead,

it uses fixed-point representations and integer arithmetic within the attention core to retain integer inference pipeline. In specific, the implementation processes 16 query tokens and 32 key/value rows per tile, maintaining a running row maximum, normalization sum, and weighted-value accumulator without materializing the full attention matrix. Q and K are represented as INT8 values with fixed-point scales using 20 fractional bits, and their scaled products are accumulated in INT64 and rounded to logits with 24 fractional bits. Exponentiation is approximated using a cached integer lookup table for e^{-x}, sampled at intervals of 2^{-10}, with linear interpolation and 15 fractional bits for the resulting weights. Contributions with $x \geq 16$ are set to zero. Whenever the running maximum increases, the previous normalization sum and output accumulator are rescaled using the same approximation. Values use a fixed-point representation with 24 fractional bits, and the final output is obtained through rounded integer division by the accumulated normalization sum.

The tiled computation is fused into a single Triton kernel launch, reducing intermediate-memory requirements and avoiding Python-level dispatch between tiles. However, this implementation retains wide fixed-point operands to limit precision loss without introducing additional intermediate requantization for INT8 matrix-multiplication primitives. Consequently, QK is evaluated through an explicit loop over head dimensions, while probability–value multiplication uses an explicit loop over key positions, both with INT64 multiply-add arithmetic. These operands cannot be mapped directly to INT8 tensor-core GEMMs. Doing so would require factoring compatible scales outside the products, decomposing wide operands into multiple narrow-integer products, or introducing further requantization. The resulting implementation is therefore memory-efficient but computationally expensive.

I Implementation Details of Log_2-Domain Integer Online Softmax Kernel

We provide a simplified pseudocode of our integer online $\log_2$-softmax attention in Listing 1. The routine first prepares the fixed-point score scale S_Y by folding the per-token query scale, the per-tensor key scale, and the $1/\sqrt{d}$ normalization into a single integer tensor with f_y fractional bits, and converts the floating-point attention mask into the integer domain.

The kernel then performs tile-wise online softmax entirely in integer arithmetic. For each query tile, it computes int8 score matrix multiplications between Q and K, converts the scores into the $\log_2$ domain using the precomputed scale S_Y, and updates the running maximum and denominator following the online softmax recurrence. The base-2 exponential and accumulator rescaling are implemented through the helper routine `DyPow`. The resulting f_w-bit attention weights are decomposed into two signed bytes in order to utilize INT8 Tensor Core matrix multiplications for the weighted value accumulation.

After all key tiles are processed, the output is normalized using a fixed-point reciprocal of the denominator, producing the integer attention output $\tilde{O}$ (`int32`) together with the value quantization scale s_v, which is later merged into the dyadic requantization scale of the subsequent layer.

The helper function `DyPow` approximates the negative-base-2 exponential in fixed-point form. It decomposes the input into integer and fractional parts, applies a quadratic approximation on the fractional interval, and then performs a rounded power-of-two shift for the integer part.

J FPGA Implementation and Evaluation

Table 11 compares our integer-only RTL attention implementation with i-BERT, i-ViT, and integer-forced FlashAttention on the AMD Alveo U280 for PixArt-Sigma and SD3-medium, using batch size one throughout. The resource utilization is reported in Tables 9 and 10. All designs use three compute units, identical HBM-bank allocations, and common compute and memory-interface clocks of 140 and 350 MHz, respectively. DSP allocations are approximately matched; integer-forced FlashAttention uses 11.8% more DSPs than ours. All implementations pass capture-driven FPGA correctness checks before measurement. Our implementation achieves latencies of 250.55 ms on PixArt-Sigma and 319.40 ms on SD3-medium. On PixArt-Sigma, this corresponds to 5.06×, 4.21×, and 2.22× speedups over i-BERT, i-ViT, and integer-forced FlashAttention, respectively; the corresponding SD3-medium speedups are 4.05×, 4.05×, and 2.07×. These measurements cover complete

Table 9: FPGA resource utilization for PixArt-Σ

Method	Online	PixArt-Σ CU	LUT	FF	DSP	RAMB36	RAMB18	URAM	LUTRAM/ SRL
FlashAttn (INT)	✓	3	315474	252643	3174	383	3	0	61462
I-ViT	✗	3	615747	441109	3093	191	3	0	17014
I-BERT	✗	3	544053	439306	2703	179	3	0	17014
Ours	✓	3	459800	388959	2838	23	3	0	51130

Table 10: FPGA resource utilization for SD3-medium

Method	Online	SD3-medium CU	LUT	FF	DSP	RAMB36	RAMB18	URAM	LUTRAM/ SRL
FlashAttn (INT)	✓	3	299149	246835	3174	311	3	0	57910
I-ViT	✗	3	564039	419472	2802	215	15	0	17014
I-BERT	✗	3	602375	481846	2943	215	15	0	17014
Ours	✓	3	439085	362856	2838	23	3	0	50650

Table 11: FPGA latency, on-chip memory, and energy comparison on an AMD U280 FPGA

Method	Online Softmax	PixArt-Σ Time	On-chip Mem.	Energy	SD3-Medium Time	On-chip Mem.	Energy
FlashAttn (INT)	✓	557.34 ms	1,730.25 KiB	17,476.23 mJ	660.53 ms	1,406.25 KiB	21,034.58 mJ
I-ViT	✗	1,053.80 ms	866.25 KiB	28,556.97 mJ	1,293.28 ms	1,001.25 KiB	34,626.61 mJ
I-BERT	✗	1,267.86 ms	812.25 KiB	34,146.58 mJ	1,293.40 ms	1,001.25 KiB	34,933.81 mJ
Ours	✓	250.55 ms	110.25 KiB	8,043.38 mJ	319.40 ms	110.25 KiB	10,477.52 mJ

all-head attention execution with device-resident inputs, excluding host–device transfers and the remaining model layers.

Our implementation also reduces measured energy by 54.0–76.4% on PixArt-Sigma and 50.2–70.0% on SD3-medium relative to the evaluated baselines. Energy is measured by integrating board power over execution windows, rather than multiplying peak power by latency. Furthermore, ours requires only 110.25 KiB of allocated block memory on either model, compared with 812.25–1,730.25 KiB for the baselines, an 86.4–93.6% reduction. This metric includes BRAM/URAM capacity and parity but excludes LUTRAM, shift-register storage, and registers. Ours uses fewer LUTs and flip-flops than i-BERT and i-ViT on both models, although integer-forced FlashAttention uses fewer of these logic resources. These results demonstrate a favorable latency–energy–block-memory trade-off at comparable DSP allocation, rather than uniform reductions across every resource category. The comparison evaluates implemented designs under common operating conditions, not an equal-area or algorithm-only ablation.

J.1 FPGA Implementation Details

Implementation. We implement our integer online attention and three baselines: I-BERT Kim et al. (2021), I-ViT Li & Gu (2023), and integer-forced FlashAttention, in handwritten SystemVerilog RTL on an AMD Alveo U280 FPGA. The baseline designs are our RTL implementations of their corresponding attention algorithms rather than FPGA implementations provided by the original authors. Each accelerator receives quantized INT8 query, key, and value tensors and computes the complete attention operation, including QK multiplication, normalization, value aggregation, and INT8 output conversion. All custom datapaths use finite-width integer arithmetic without general-purpose floating-point arithmetic units.

Each compute unit (CU) implements an independent attention engine with dedicated 512-bit AXI interfaces for Q, K, V, and output. All designs use three CUs distributed across the three SLRs of the U280, with dedicated HBM banks assigned to each CU. Key/value tensors are loaded in tiles of 64

keys, with banked and double-buffered storage to overlap memory accesses with computation. Our design processes 16 query rows and 16 channels in parallel per CU and performs online normalization as each tile is processed. The integer-forced FlashAttention baseline similarly performs tiled online normalization, but uses its own integer exponential approximation. In contrast, I-BERT and I-ViT preserve their original full-row normalization: all scores for a query are first generated and buffered, followed by maximum reduction, exponential approximation, normalization, and value accumulation. Thus, these designs require row-sized intermediate storage rather than the streaming accumulators used by the online implementations.

Build and comparison. We use Vivado/Vitis 2023.2 and XRT 2.16.204 with the `xilinx_u280_gen3x16_xdma_1_202211_1` platform. To provide a common operating point, all reported implementations use three CUs at 140 MHz compute and 350 MHz HBM-interface clocks. The designs are approximately DSP-matched rather than area-matched; differences in LUT, register, and on-chip memory utilization reflect their respective architectures. Resource utilization in Tables 9 and 10 are obtained from the final routed design and includes the generated transport and control logic in the dynamic region.

Validation and measurement. We validate the FPGA implementations using quantized tensors and scales captured from actual GPU forward passes. Each implementation is checked against an independent arithmetic reference and its corresponding GPU implementation. FPGA outputs exactly match the independent references; our implementation also exactly matches the captured GPU output codes, while all baseline outputs differ by at most one INT8 output-code LSB. For latency, complete attention heads are distributed across the three CUs and we report the median device-resident execution time after warmup. The measured interval includes FPGA memory accesses, attention computation, and INT8 output conversion, but excludes host–device transfers and host-side input preparation. Energy is obtained by integrating the U280 board input-power telemetry over the same execution interval and is reported as mean gross board energy.

K LIMITATIONS

In this work, we focus on integer-only quantization of the diffusion transformer backbone, which dominates inference cost due to repeated execution across denoising timesteps. The VAE decoder and text encoder are not quantized, following common practice in diffusion quantization literature. In practical edge systems, the one-time text encoding and VAE decoding can execute on an FP-capable host processor, while the repeatedly invoked diffusion backbone is offloaded to an integer accelerator. Also, our evaluation adopts the W8A8 setting for all matrix multiplications, following the standard in integer-only quantization (Kim et al., 2021; Li & Gu, 2023). However, extending this to lower bitwidths such as W4A8 or W4A4 remains an important direction, as these settings are more relevant for memory-constrained edge deployment.



Figure 12: Additional qualitative evaluation of generated images using FLUX.1-schnell on MJHQ dataset. Compared to integer-only baselines, our method produces images with noticeably improved visual quality and closer semantic alignment to the BF16 results, which is consistent with the quantitative improvements in both image quality and similarity to BF16 outputs.

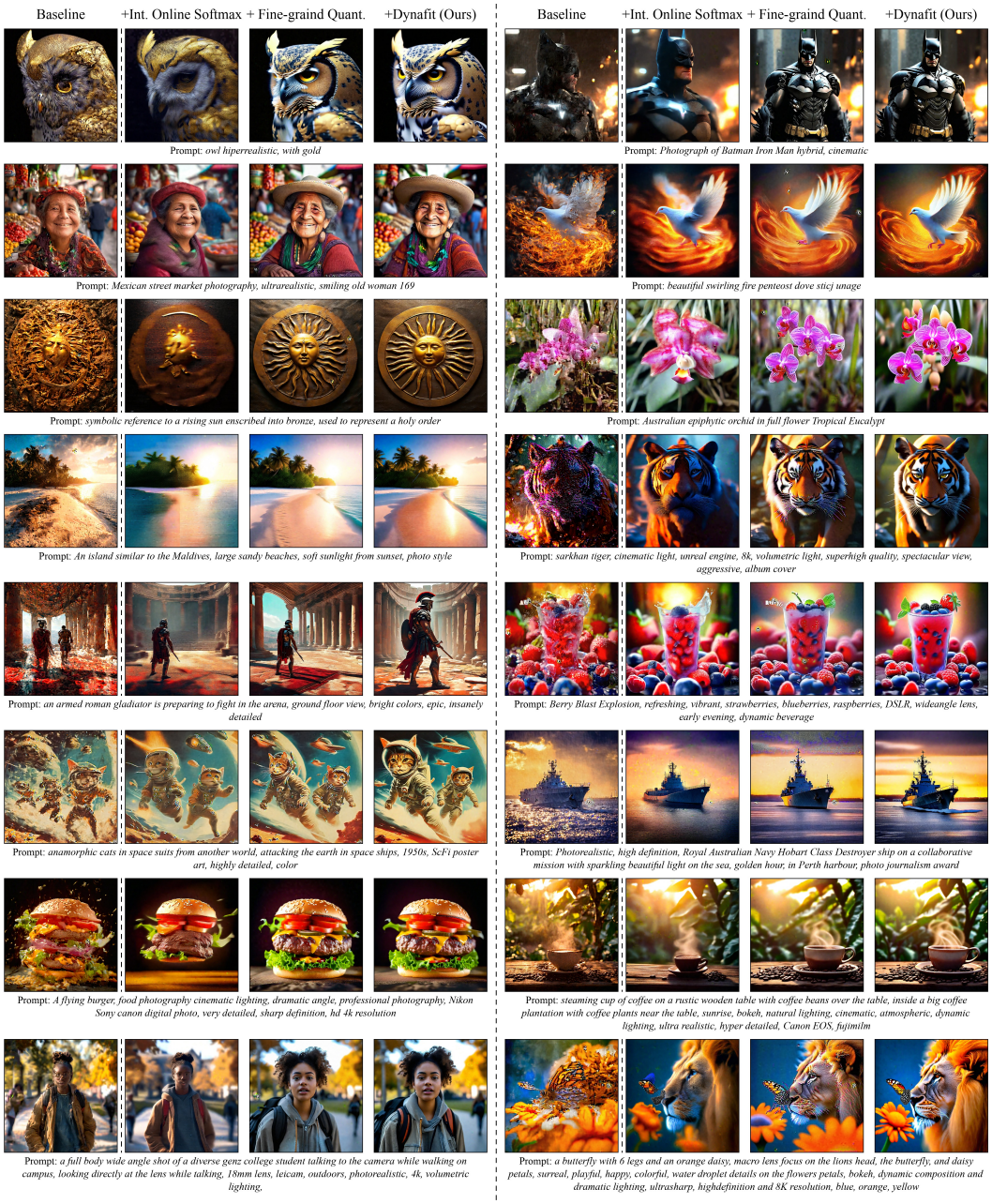


Figure 13: Qualitative ablation study on the MJHQ dataset using PixArt-Σ. From left to right: Baseline, + integer online softmax (§4.2), + fine-grained dyadic requantization (§4.3), and + Dynafit (§4.1; Ours).

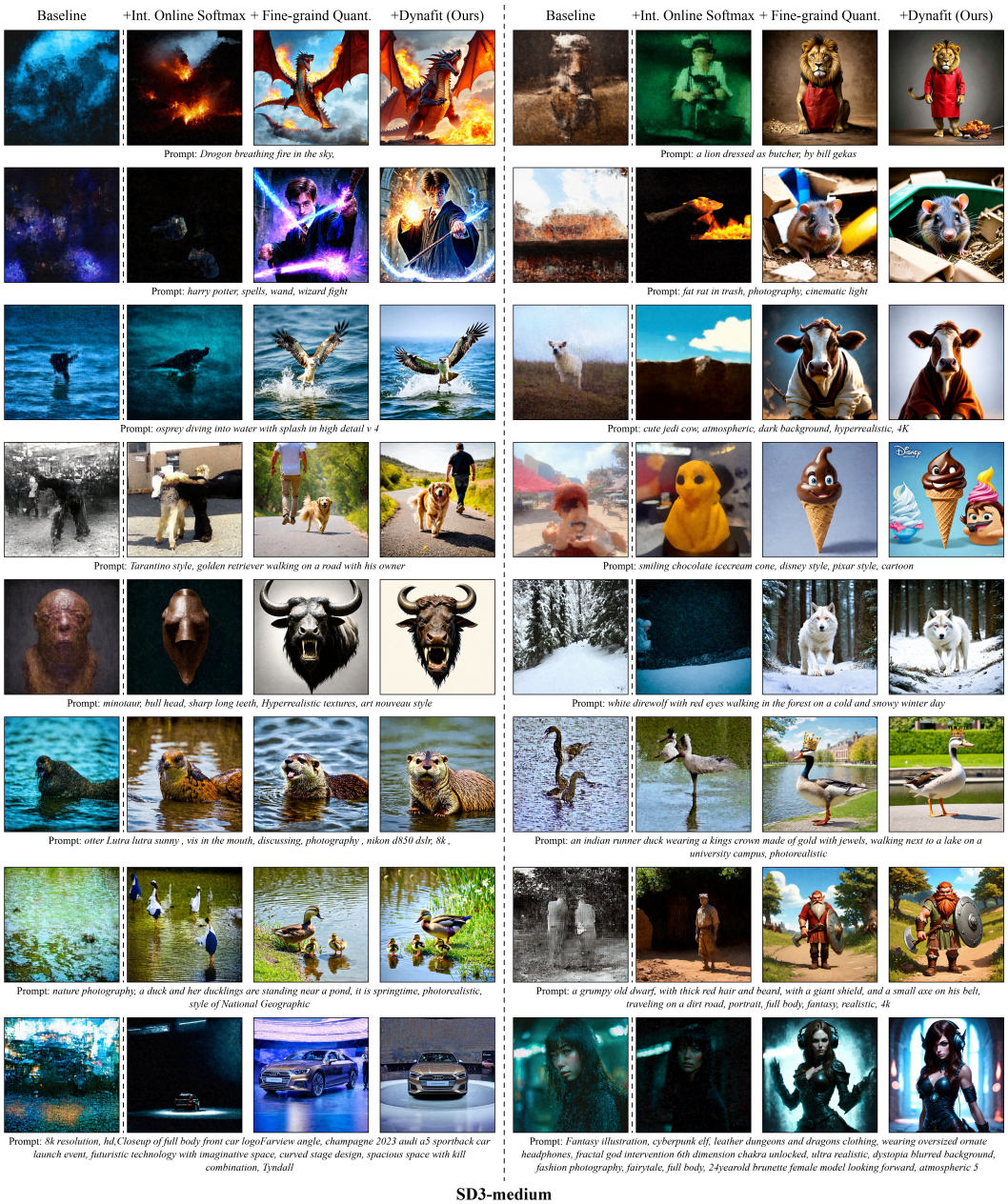
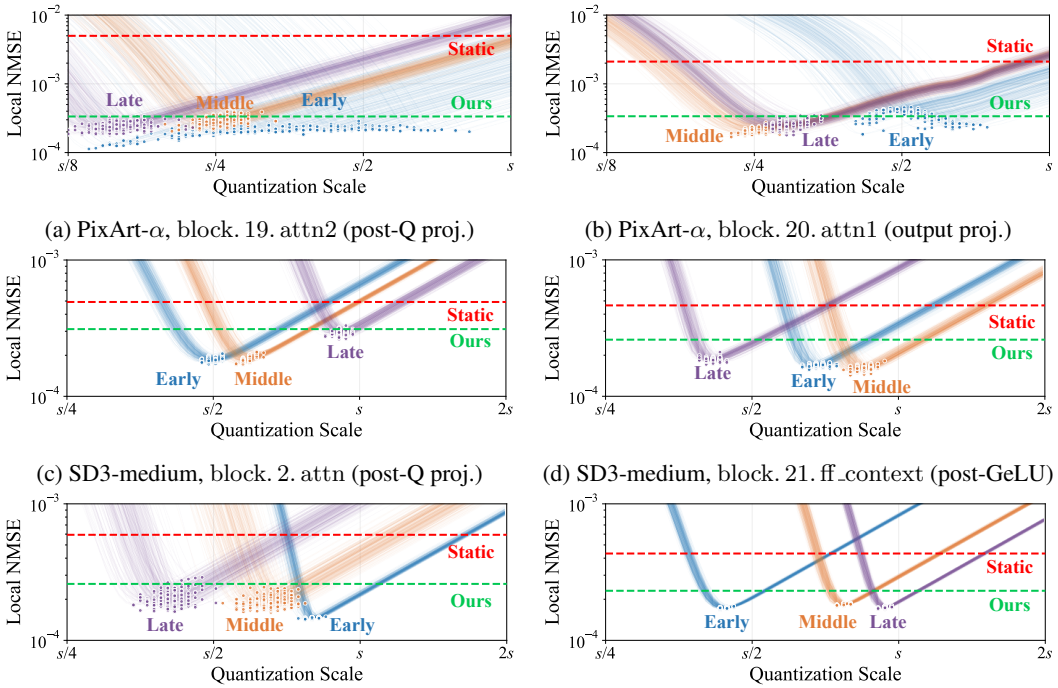


Figure 14: Qualitative ablation study on the MJHQ dataset using SD3-medium. From left to right: Baseline, + integer online softmax (§4.2), + fine-grained dyadic requantization (§4.3), and + Dynafit (§4.1, Ours).



(a) PixArt-α, block. 19. attn2 (post-Q proj.)

(b) PixArt-α, block. 20. attn1 (output proj.)

(c) SD3-medium, block. 2. attn (post-Q proj.)

(d) SD3-medium, block. 21. ff_context (post-GeLU)

(e) PixArt-Σ, block. 22. attn2 (post-Q proj.)

(f) PixArt-Σ, block. 3. attn1 (post-Q proj.)

Figure 15: Quantization error curve within a layer across samples. Curves denote individual samples, the red and green lines denote mean error of all samples using static scale and dynafit, respectively.

Listing 1: Pseudocode for integer online $\log_2$-softmax attention. The helper function `DyPow` approximates the fixed-point base-2 exponential, and the main routine performs block-wise integer-only online softmax attention with reciprocal normalization

```
def DyPow(z_int, f_y, f_w, a0, a1, a2):
    u  = -z_int
    n  = u >> f_y                          # integer part
    f  = u & ((1 << f_y) - 1)              # fractional part
    F  = a0 - (a1 * f) >> f_y + (a2 * f * f) >> (2 * f_y)
    return round_shift(F, n)               # (F + 2^(n-1)) >> n
```

```
def integer_attention(Q, K, V, s_q, s_k, s_v, alpha, mask, f_y, f_w):
    a0 = 1 << f_w
    a1 = round(0.6711 * (1 << f_w))
    a2 = round(0.1711 * (1 << f_w))

    S_Y = round((s_q * s_k * alpha / ln2) * (1 << f_y)).to(int32)
    mask_int = where(mask < -1, -1e9, 0).to(int64)

    for each query block i:
        m = -INF
        l = 0
        o = zeros(d)

        for each key block j:

            # Score computation (int8 Tensor Core matmul)
            S_tile = dot_int8(Q[i], K[j].T)
            Y      = int64(S_tile) * S_Y[i, j]
            Y      = Y + mask_int[j]

            # Online max update
            m_new  = max(m, row_max(Y))

            # Rescale old accumulators via DyPow
            R  = DyPow(-(m_new - m), f_y, f_w, a0, a1, a2)
            l  = round_shift(l * R, f_w)
            o  = round_shift(o * R, f_w)

            # Tile attention weights
            W  = DyPow(Y - m_new, f_y, f_w, a0, a1, a2)
            l += row_sum(W)

            # int8 decomposition for Tensor Core W*V
            W_hi, W_lo = signed_byte_split(W)
            o += (dot_int8(W_hi, V[j]) << 8) \
               +  dot_int8(W_lo, V[j])

            m  = m_new

        # Fixed-point reciprocal normalization
        l_inv = round_div(1 << F_DIV, l)
        O[i]  = round_shift_signed(o * l_inv, F_DIV)

    return O, s_v
```