CHAMELEON: DYNAMIC FORMAT ADAPTER FOR EFFICIENT DIFFUSION

Anonymous authors
Paper under double-blind review



Figure 1: Images generated by each per-architecture PTQ baseline and by Chameleon at 4-bit weights (W_4A_8), from the COCO-2014 caption "An old city fire hydrant has 'Love was found here' painted on it."

ABSTRACT

A very popular way to enable image generation on memory-constrained accelerators using modern spatiotemporal generative models is post-training quantization (PTQ). However, every existing diffusion PTQ scheme shares an unstated commitment: the *number format* is fixed in advance, and only the scale, zero point, or the per-layer bit-width is allowed to move. This is unfortunate, because at a fixed bit-width the best format depends on the distribution being encoded, and that distribution differs from tensor to tensor: across weight channels, across layers, and – in a denoising model – along the *diffusion timestep*, where activations slide from heavy-tailed and noise-dominated at $t \to T$ to tightly clustered and structured at $t \to 0$. One format for the whole network is therefore mispriced for most of its tensors, and re-tuning a scale cannot repair a format mismatch. In this work, we propose a PTQ framework, which we call **Chameleon**, that holds the bit-width fixed and treats the number format itself as a discrete variable, chosen per weight channel and per *(layer, timestep bucket)* activation tensor. Chameleon draws activation formats from a palette of {INT8, FP8 E4M3, FP8 E5M2, MXFP8, MXINT8} and weight formats from {INT8, MXINT8} at 8 bits or {INT4, NF4, FP4 E2M1, MXINT4, MXFP4} at 4 bits, choosing activation formats ahead of time from two cheap statistics – empirical activation kurtosis and closed-form diffusion SNR $\bar{\alpha}_t/(1-\bar{\alpha}_t)$ – stored in a per-(layer, timestep bucket) lookup table, and weight formats offline by reconstruction error. An architectural fork adapts the same selection layer to each model family, so each model exercises the palette as far as its own statistics warrant: weights and per-(layer, bucket) activations on multi-step SDXL, weights alone where a single sampler step leaves no timestep axis. We evaluate Chameleon across three diffusion regimes – multi-step SDXL, single-step SDXL-Turbo, and the PixArt-α Diffusion Transformer – on COCO-2014. Chameleon achieves the best FID for all six backbone$\times$bit-width settings, while preserving semantic alignment with the text prompts (CLIP within 0.24 of the FP16 reference in every setting, and the best of all quantized methods at W_4A_8). Adapting the format, rather than only its scale, recovers distributional fidelity at 4-bit weights while holding prompt adherence at the FP16 level.[1]

[1] website code

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

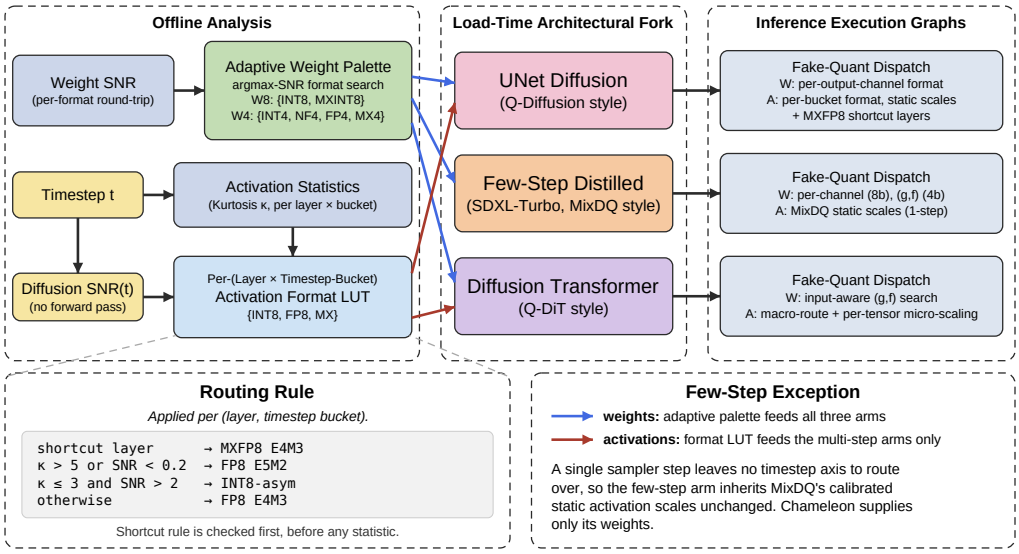


Figure 2: **Chameleon framework:** Offline, a one-pass calibrator accumulates per-(layer, bucket) activation moments for the empirical kurtosis $\kappa_l(t)$, while the diffusion $\mathrm{SNR}(t)$ is read from the noise schedule. The router applies the rule of Section 4.3 to both and writes the format into an integer lookup table $F \in \{0, \ldots, 4\}^{L \times B}$ (code 4 is unused); weights are selected offline (Sections 4.5 and 4.6). At inference, one fake-quantization module reads the `LUT` per layer per step on the multi-step paths and dispatches to the matching routine.

1 Introduction

The frontier of generative AI lies not in the cloud but on the edge (Zhou et al., 2025). Latent diffusion models, few-step distilled models, and diffusion transformers (`DiTs`) all generate images from pure noise by stepping through a learned vector field for tens to hundreds of forward passes through a network holding billions of parameters: `SDXL` alone has roughly 2.6 billion, and text encoding and the variational auto-encoder add another billion. Replacing 16-bit numerics with 8-, 6-, or 4-bit ones – post-training quantization (`PTQ`) – is the standard response, cutting memory footprint and latency while raising throughput (Nagel et al., 2021; Frantar et al., 2023; Xiao et al., 2023). This paper proposes a `PTQ` scheme for the iteratively denoising generators in that list. The key idea is that while weights can be statically quantized, activations should not be, because of the iterative nature of the inference pipeline. We call our **dynamic format adapter** framework Chameleon: at a fixed bit-width it chooses a number format per tensor – per weight channel, per layer, and per timestep bucket where a timestep axis exists – using cheaply computed statistical heuristics.

Prior work has contributed a great deal to this space (Section 2), but the four diffusion `PTQ` baselines we compare against – `Q-Diffusion`, `PTQ4DM`, `MixDQ` and `Q-DiT` – all take quantization to mean rounding to the nearest `INT8` format. This is suboptimal for at least two reasons.

First, at a fixed bit-width the format is still free, and the right choice follows the distribution being encoded: Kuzmin et al. (2022) show that the signal-to-quantization-noise ratio (`SQNR`) at a given width is a function of both that distribution and the format, and Zhang et al. (2024) turn this into a per-layer `INT`-or-`FP` choice for `LLMs`. Tensors within one network differ in shape – across weight channels and across layers – so no single format is right for all of them. Denoising networks add a second axis: the activations of a single layer are themselves a parametric family of distributions indexed by timestep t (Li et al., 2023). At $t \to T$, activation tensors are dominated by Gaussian noise, and the internal activations they induce are heavy-tailed, with kurtosis (κ) above 5; at $t \to 0$ they are dominated by the structured signal of a near-convergence latent, with κ often below 3. The format that is right at $t = 0$ is wrong at $t = T$. Re-tuning the scale of `INT8` cannot make it tolerate heavy tails, any more than re-scaling `FP8 E5M2` can give its logarithmic bins the density to render fine texture. The fix is to *change formats* along the schedule.

Second, microscaling formats – MXFP8, MXINT8, MXINT4 (OCP Microscaling Formats Working Group, 2023; Rouhani et al., 2023) – are now hardware features on Blackwell-class accelerators, with FP8 already native on Hopper and Ada. One shared 8-bit exponent per 32 elements lets a single tensor hold values spanning very different magnitudes without one global scale covering them all. This is exactly what the concatenations at every UNet shortcut need, where shallow skip features meet deep upsampled features. Measured on SDXL, the tensor entering a shortcut conv1 spans a peak-to-median per-channel range of $58\times$ at the median layer and up to $163\times$ (Appendix A.11), so one per-tensor scale must cover outliers that most channels never approach. Q-Diffusion split the computational graph for this; with MX formats, one need not.

These two observations – that the right format depends on the tensor, and on t where a timestep axis exists, and that the format should sometimes be a microscaling block format – are the substance of this paper. Figure 2 sketches the resulting framework end-to-end.

We present **Chameleon**, a PTQ framework that treats the number format as a discrete variable, chosen per weight channel and per (*layer*, *timestep bucket*) activation tensor from the palettes of Section 4.2. Two cheap statistics – the empirical kurtosis $\kappa_l(t)$ and the closed-form diffusion SNR – decide it offline, and the result costs one indexed read per layer per step.

1.1 Motivation

PTQ research for time-stepped image generators is architecture-specific, but the three lines of work converge on one observation: the activation distributions inside a denoising network are non-stationary along the diffusion trajectory, and that non-stationarity is the dominant source of quantization error. What none of those four adjusts is the number format itself. Q-Diffusion, PTQ4DM, MixDQ, and Q-DiT all commit up-front to a single format – invariably INT8 – and route their design effort into scales, zero points, per-layer bit-widths, group sizes, and sampler schedules on top of that fixed commitment. This is an odd restriction to accept when the underlying activation family spans regimes as different as heavy-tailed, noise-dominated activations at $t \to T$ and tightly clustered structured signals at $t \to 0$. If the family of activation distributions is the problem, the family of number formats is the natural response.

Chameleon respects those architecture-specific lessons rather than discarding them: a hard fork at model load selects a UNet, few-step, or DiT execution path (Section 4.6), and because the format choice sits *underneath* that machinery, each path threads Chameleon's LUT through the treatment inherited from prior work. "Which format to use" is thereby decoupled from "which architecture we are quantizing."

This is tractable because the format is a *discrete* variable: the palette is small (five formats for activations, six for weights at 4 bits) and fixed by what accelerators implement natively, so selection is a routing problem rather than a continuous optimization, and both statistics that decide the per-tensor Mean Squared Quantization Error (MSQE) comparison are cheap – kurtosis is a one-pass moment accumulation, and SNR is read off the noise schedule with no forward pass.

2 Related Work

Extended versions of the paragraphs below, and a note on LLM quantization, are in Appendix A.1.

Diffusion-model PTQ: Q-Diffusion (Li et al., 2023) first recognized that activation distributions inside a denoising UNet are not stationary across the schedule, answering with a uniform-timestep calibration sampler and a split quantizer for the shortcut layers. PTQ4DM (Shang et al., 2023) added per-bucket calibration statistics. Both take the format itself – INT8 – as a given and tune scales around it.

Few-step and consistency-model PTQ: Few-step distilled models – by consistency (Luo et al., 2024) or adversarial (Sauer et al., 2024) distillation – are hostile to quantization: each of their one to four steps carries far more of the semantic burden than one step of a 50-step trajectory. MixDQ (Zhao et al., 2024) combines metric-decoupled mixed precision, assigned by integer programming, with a BOS-aware bypass in the cross-attention K and V projections; Q-Sched (Frumkin & Marculescu, 2026), concurrent with and complementary to Chameleon, instead learns a single pair of scalar

preconditioning coefficients that warp the sampling trajectory. Our few-step fork composes with MixDQ rather than replacing it (Section 4.6).

DiT quantization: Diffusion transformers exhibit large variance across the *input* channel dimension of their linear-layer activations, which tensor- or output-channel-wise quantization cannot absorb. Q-DiT (Chen et al., 2025a) introduced *group-wise* weight quantization with an evolutionary search over group sizes $g \in \{32, 64, 128, 192, 288\}$, and *dynamic* on-the-fly scale and zero-point computation for activations – again with INT as the only number system. ViDiT-Q (Zhao et al., 2025) reports W_8A_8 and W_4A_8 on PixArt-α with finer-grained dynamic quantization, again within INT. Our DiT fork keeps both, generalizing the group search to a joint (g, f) search over group size and format (Section 4.6). PTQ4DiT (Wu et al., 2024), FP4DiT (Chen et al., 2025b) and ViDiT-Q are not re-run in our harness and so do not appear in Table 1 (Appendix A.1).

Microscaling formats: The OCP MX specification (OCP Microscaling Formats Working Group, 2023; Rouhani et al., 2023) standardized block-shared-exponent narrow formats: each block of 32 elements carries one shared 8-bit exponent, absorbing intra-tensor magnitude variation without per-channel scale infrastructure. To our knowledge, Chameleon is the first PTQ scheme for diffusion models to use MX formats *adaptively*, routing concatenation-bearing UNet shortcut layers to MXFP8 by topology.

Format selection beyond INT: At a fixed bit-width the format is itself a free variable: MoFQ (Zhang et al., 2024) chooses INT or FP per layer for LLMs by quantization error, and So et al. (2023a) search a per-tensor floating-point encoding, layer by layer, for diffusion. TDQ (So et al., 2023b) comes closest to our temporal axis, but varies the quantization *interval* with t, not the format (Appendix A.1).

Where we sit: Chameleon is the first PTQ scheme for time-stepped image generators to treat the *number format* as a discrete variable indexed by *(layer, timestep bucket)* – per-layer format selection at a fixed width was introduced for LLMs (Zhang et al., 2024), and timestep-dependent quantization *parameters* for diffusion (So et al., 2023b), but not the format along the schedule – and the first to demonstrate the same routing principle across UNet, few-step UNet, and image DiT under one framework. It is composable rather than competitive: where existing methods care about scales (Q-Diffusion, PTQ4DM), bit-widths (MixDQ), group sizes (Q-DiT), or sampling trajectories (Q-Sched), Chameleon cares about formats, and the underlying calibration loop and group search from each of these methods can sit on top of it without modification.

3 Background & Preliminaries

3.1 Diffusion Basics

In the variance-preserving discretization used by Stable Diffusion and its derivatives, the noise schedule is summarized by the cumulative product $\bar{\alpha}_t = \prod_{s \le t}(1 - \beta_s)$, and the *signal-to-noise ratio* admits the closed form

$$\mathrm{SNR}(t) = \frac{\bar{\alpha}_t}{1 - \bar{\alpha}_t}, \tag{1}$$

which falls monotonically in t across five to eight orders of magnitude along the schedule. We use it as one of our two routing statistics: it is known at calibration time and costs nothing to evaluate. Appendix A.2 gives the continuous-time formulation, the per-model SNR ranges, and how DiTs and few-step models fit the same picture.

3.2 Asymmetry in the Statistics of Weights vs. Activations

A trained denoising network is a static set of weights $\{W_l\}$, whose statistics do not depend on t and can be characterized once, offline, per channel. Activations are different: for a fixed layer l, the input tensor $x_l(t)$ depends on both the random latent and the timestep, and its distribution changes dramatically along the schedule. Appendix A.3 works through an illustrative example on SDXL: measured along the schedule, activation kurtosis and $\mathrm{SNR}(t)$ separate three phases – heavy-tailed at

low SNR (Phase 1), settling at mid SNR (Phase 2), and near-Gaussian at high SNR (Phase 3) – each best served by a different format, and layers cross them at different times, which is why Chameleon routes per (layer, bucket) rather than per bucket alone.

4 THE CHAMELEON FRAMEWORK

4.1 PROBLEM STATEMENT

Let $\mathbb{N}$ be a denoising network with quantizable layers $l = 1, \ldots, L$. For each layer let $X_l(t) \in \mathbb{R}^{C_1 \times C_2 \times \cdots}$ denote the input activation tensor at timestep t and $W_l \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times \cdots}$ the weight tensor. We discretize the schedule into B uniform-width buckets indexed $b = 0, \ldots, B-1$ and write $b(t)$ for the bucket containing t.

Given a finite palette $\mathcal{F}$ of number formats – each with its own scaling granularity g (per-tensor or per-block for activations, per-output-channel or grouped for weights; Section 4.6 fixes this per architecture) – we seek for each (layer l, bucket b) the format minimizing the mean-squared quantization error

$$f^{\star}_{l,b} \;=\; \arg\min_{f \in \mathcal{F}} \; \mathbb{E}_{t \in b}\Big[\, \big\| X_l(t) - \mathrm{DQ}_f\big(\mathrm{Q}_f(X_l(t))\big) \big\|_2^2 \,\Big], \tag{2}$$

where Q_f and DQ_f are the quantization and de-quantization operators of format f at its native granularity g. The analogous objective is solved per output channel for weights, with the expectation over t dropped because weights are static.

Solving equation 2 exhaustively means materializing $X_l(t)$ and computing five round-trips per (layer, bucket). That is feasible for one calibration pass, but two cheap scalars – the empirical kurtosis $\kappa_l(t)$ of $X_l(t)$ and the diffusion SNR of equation 1, which depends on t alone – *approximate* the argmin well enough to populate a static lookup table that ships with the model.

4.2 FORMAT PALETTE

The activation palette is

$$\mathcal{F}_a \;=\; \{\texttt{INT8-asym},\ \texttt{FP8 E4M3},\ \texttt{FP8 E5M2},\ \texttt{MXFP8 E4M3},\ \texttt{MXINT8}\},$$

indexed by integer codes $0, 1, 2, 3, 4$ in the runtime LUT. The first three are conventional per-tensor formats; the last two are microscaling formats with block size 32 and a shared 8-bit exponent.

The weight palette is bit-width dependent:

$$\mathcal{F}_w^{(8)} \;=\; \{\texttt{INT8-sym},\ \texttt{INT8-asym},\ \texttt{MXINT8}\},$$
$$\mathcal{F}_w^{(4)} \;=\; \{\texttt{INT4-sym},\ \texttt{INT4-asym},\ \texttt{NF4},\ \texttt{FP4 E2M1},\ \texttt{MXINT4},\ \texttt{MXFP4 E2M1}\}.$$

$\mathcal{F}_w^{(4)}$ above is the full 4-bit palette; each architectural fork searches a subset of it (Section 4.6), and Appendix A.11 describes each code.

We keep the palette small: the MSQE landscape has clear plateaus (Phases 1–3 of Appendix A.3), and extra formats inside a plateau buy nothing while inflating the LUT. MXINT8 is defined in the activation code space for symmetry with the weight palette but is never selected by the routing rules of Section 4.3; the remaining four are the smallest set covering all three phases plus the concatenation case. Alternatives to plain integer and floating-point coding exist in the broader compression literature (Sanyal et al., 2020; Johnson, 2018), but we restrict the palette to formats that map onto upcoming hardware; Appendix A.11 describes each.

4.3 ROUTING THRESHOLDS

For each (layer, bucket) pair we apply the decision rule below, which approximates the argmin of equation 2 under a generalized-Gaussian model. UNet shortcut layers are pinned by topology before

any statistic is consulted; the rest are routed by κ and SNR:

$$\begin{aligned}
\text{If } l \text{ is a UNet shortcut layer:} &\quad f^\star_{l,b} = \texttt{MXFP8 E4M3} && \text{(shortcut concatenation)}\\
\text{Else if } \kappa_l(t) > 5 \text{ or } \mathrm{SNR}(b) < 0.2: &\quad f^\star_{l,b} = \texttt{FP8 E5M2} && \text{(Phase 1, heavy tails)}\\
\text{Else if } \kappa_l(t) \le 3 \text{ and } \mathrm{SNR}(b) > 2: &\quad f^\star_{l,b} = \texttt{INT8-asym} && \text{(Phase 3, clean signal)}\\
\text{Else:} &\quad f^\star_{l,b} = \texttt{FP8 E4M3} && \text{(Phase 2, settling)}
\end{aligned}$$

Here $\mathrm{SNR}(b) \equiv \mathrm{SNR}(t_b)$, evaluated at the bucket midpoint $t_b = bT/B + T/(2B)$, as in Algorithm 1. The SNR gate overrides kurtosis by design: below τ_{low} every non-shortcut layer is routed to FP8 E5M2 (Table 7), so per-layer routing acts in the mid- and high-SNR buckets. The constants $(\kappa_{\text{low}}, \kappa_{\text{high}}) = (3, 5)$ and $(\tau_{\text{low}}, \tau_{\text{high}}) = (0.2, 2.0)$ are fixed a priori rather than tuned per model: the kurtosis gates are where the per-format MSQE curves cross under a generalized-Gaussian model, and the SNR gates mark the macroscopic phases of the noise schedule. Empirically, the routing is insensitive to their exact values: shifting the kurtosis gates from $(3.0, 5.0)$ to $(3.5, 5.5)$ – $+17\%$ and $+10\%$ respectively – changes FID by 0.02, below the 0.04 run-to-run variation we observe between two independent calibrations of an identical configuration (Appendix A.9).

4.4 Calibration: Lockstep Sampling and LUT Construction

Calibration runs as a single pass on top of a Q-Diffusion-style uniform timestep sampler. We collect 128 prompts (COCO captions), bucket the schedule into $B = 10$ uniform-width windows, and generate one 20-step trajectory per prompt. During each forward pass, every Chameleon-instrumented layer accumulates running statistics into an $L \times B$ table, pooling the conditional and unconditional halves of each classifier-free-guidance batch; Algorithm 1 in the supplement gives the full pseudocode.

The accumulator stores raw power sums (n, $\sum x$, $\sum x^2$, $\sum x^3$, $\sum x^4$) plus a running min and max, from which $\kappa_l(t)$ follows in closed form. Memory is $O(1)$ in the number of prompts – seven scalars per (layer, bucket) – so SDXL's 372 quantized layers at $B = 10$ need 26,040 scalars, about 200 KB.

4.5 Per-Channel Weight Selection

Weights are static, so their format selection runs offline and at finer granularity. For each output channel c of each conv or linear, we round-trip $W_l[c, :, \ldots]$ through every format in the appropriate palette and pick the format with the highest quantization signal-to-noise ratio (SQNR) in dB:

$$f^\star_{l,c} \;=\; \arg\max_{f \in \mathcal{F}_w} \; 10 \log_{10} \frac{\mathbb{E}\big[W_l[c]^2\big]}{\mathbb{E}\Big[\big(W_l[c] - \mathrm{DQ}_f(\mathrm{Q}_f(W_l[c]))\big)^2\Big]}. \tag{3}$$

The per-channel grain matters: even within one SDXL conv layer, the kurtosis of weights along the output-channel axis varies enough that a tensor-wise choice would force a compromise no channel benefits from. It costs one byte of format metadata per output channel, about 1 MB across the UNet.

4.6 The Architectural Fork

A model loader inspects the network topology once and dispatches to one of three execution paths. The fork is the only branch in the framework; everything downstream is shared.

UNet diffusion path (SDXL): Activations follow Section 4.3 with static, ahead-of-time-calibrated scales (Q-Diffusion style). Layers matching the regex up_blocks.*.resnets.*.conv1 – the conv layers immediately downstream of a shortcut concatenation – are unconditionally routed to MXFP8 E4M3, bypassing the κ/SNR check, because their inputs are wide-tailed in a way that no per-tensor format can absorb. Weights follow Section 4.5.

Few-step path (SDXL-Turbo): A one-step schedule has no timestep axis to route over, so this path inverts the division of labor: Chameleon owns the *weights* and inherits the *activations*

wholesale. Activations use MixDQ's calibrated static per-tensor scales together with its metric-decoupled mixed-precision assignment, unchanged. Weights are replaced by Chameleon's adaptive palette – per-channel SQNR selection over $\mathcal{F}_w^{(8)}$ at 8 bits, and a per-layer (g, f) search over $\{32, 64, 128, 192, 288\} \times \{\texttt{INT4-asym}, \texttt{NF4}, \texttt{FP4 E2M1}, \texttt{MXINT4}, \texttt{MXFP4 E2M1}\}$ at 4 bits (the MX formats carry one power-of-two scale per 32-element block, so g does not change them and they are searched at $g = 32$) – substituted for MixDQ's symmetric fold at the point where the quantized weight tensor materializes (the nine up-block conv_shortcut layers stay FP16 at both bit-widths; Appendix A.4). The weight palette alone therefore accounts for the improvement over MixDQ at both bit-widths, and MixDQ's BOS-aware bypass becomes unnecessary under it (Appendix A.9), which also removes its batch-size-one inference constraint.

DiT path (PixArt-α): The MX shortcut logic is dropped – DiTs do not have shortcut concatenations of the kind UNets do. Two changes replace the shortcut rule:

(i) Weights use a joint (g, f) search. For each linear layer we evaluate the SQNR – input-aware at 4 bits, i.e. weighting each input channel's error by its activation energy $\mathbb{E}[x_i^2]$ – of every $(g, f) \in \{32, 64, 128, 192, 288\} \times \{\texttt{INT4-asym}, \texttt{NF4}, \texttt{FP4 E2M1}\}$ tuple and pick the argmax. The search is restricted to the non-MX members of $\mathcal{F}_w^{(4)}$, since a searched g already supplies the block granularity MX would contribute. Groups run along the *input-channel* dimension, following Q-DiT's observation that this is where DiT activations have their dominant variance.

(ii) Activations use *macro-routing with dynamic per-sample scaling*: the format choice $f_{l,b}^{\star}$ from the LUT (equation 2) is the macro decision, but the scale is computed dynamically from the current sample's range at inference rather than read from a calibrated table. This follows Q-DiT's observation that DiT activations shift drastically across samples. Applied unchanged to PixArt-α, this rule produces a markedly different assignment from the UNet's. At both bit-widths, FP8 E5M2 wins in 97–100% of the 282 quantized linears in *every* bucket and INT8-asym is never selected, so the three-phase progression of Appendix A.3 does not appear here. Since $\mathrm{SNR}(b)$ is large in the cleanest buckets, the gate that fires must be $\kappa_l(t) > 5$: DiT activations stay heavy-tailed along the entire schedule. The thresholds are identical across both architectures, with no per-model tuning, yet they track the architecture rather than reproducing the UNet's answer.

4.7 Implementation and Runtime

The runtime path is a single nn.Module inserted before each quantizable layer; Algorithm 2 gives the pseudocode and Appendix A.7 two reproducibility notes (LUT buffer registration and FP8 range clamping). Here we say how our MX numbers should be read.

MX block formats are simulated by reshaping to blocks, computing the shared exponent, round-tripping each element, and reshaping back. The activation path follows the OCP specification exactly: the shared scale is the power-of-two $2^{\lfloor \log_2 \max_i |x_i| \rfloor}$ (an E8M0 exponent), so the simulation matches conforming MX hardware. The SDXL weight-side implementation instead uses an unconstrained per-block scale, which is strictly more expressive than the specification allows (the few-step path's MX weights already use power-of-two scales, and the DiT path selects no MX weight format); the numbers we report for MX weight formats should be read accordingly. Appendix A.12 measures what conformance would cost and how it changes the palette. We report fake-quant numbers throughout because real INT8/FP8/MX kernels are not yet uniformly available across the architectures we evaluate; on Blackwell we expect native kernels (e.g., Microsoft's microxcaling) to remove the simulation overhead.

The whole framework is roughly 5.8 kLOC of PyTorch. We will release the code and the per-model LUTs that reproduce Section 5 exactly.

5 Experiments

5.1 Setup

We quantize SDXL base 1.0 (Podell et al., 2024), SDXL-Turbo (Sauer et al., 2024), and PixArt-α XL/2 1024-MS (Chen et al., 2024) through the UNet, few-step, and DiT paths of Section 4.6, and

re-run each family's prior PTQ schemes through one shared harness with identical prompts, seeds, and scoring. We report clean-FID (Parmar et al., 2022) over 24,576 COCO-2014 validation captions (Lin et al., 2014) and CLIP-score (Radford et al., 2021) as a prompt-adherence check; resolutions and samplers differ per family (Table 1), so FID is comparable within a backbone but not across. All runs use fake quantization on a single A100, so we report quality and calibration cost but not latency. Appendix A.4 gives the full setup.

Table 1: Combined evaluation on COCO-2014 (24,576 prompts, one image per caption). Clean-FID is computed against the full val2014 set and CLIP-score (ViT-L/14, cosine $\times 100$) against the matching captions; best FID per backbone (and CLIP for W_4A_8) in **bold**. *Ours. In both SDXL-Turbo rows the nine up-block conv_shortcut layers stay FP16 (Section 4.6).

Setup	Method	Bits	FID ↓	CLIP ↑
SDXL (50 steps, CFG 7.5, 1024×1024)	FP16 baseline	$W_{16}A_{16}$	16.16	26.88
	Q-Diffusion	W_8A_8	14.70	26.77
	PTQ4DM	W_8A_8	14.65	26.68
	Chameleon*	W_8A_8	**14.23**	26.70
	Q-Diffusion	W_4A_8	21.41	26.27
	PTQ4DM	W_4A_8	21.82	26.21
	Chameleon*	W_4A_8	**14.43**	**26.64**
SDXL-Turbo (1 step, no CFG, 512×512)	FP16 baseline	$W_{16}A_{16}$	21.63	26.63
	MixDQ	W_8A_8	21.43	26.64
	Chameleon*	W_8A_8	**20.93**	26.62
	MixDQ	W_4A_8	24.50	25.88
	Chameleon*	W_4A_8	**21.29**	**26.85**
PixArt-α (20 steps, CFG 4.5, 1024×1024)	FP16 baseline	$W_{16}A_{16}$	27.63	25.99
	Q-DiT	W_8A_8	27.69	25.99
	Chameleon*	W_8A_8	**24.33**	25.81
	Q-DiT	W_4A_8	23.53	25.73
	Chameleon*	W_4A_8	**22.24**	**26.09**

5.2 Main Results

Table 1 reports the headline numbers for the three backbones.

In SDXL, Chameleon reaches 14.23 FID at W_8A_8 and 14.43 at W_4A_8, against FP16 16.16 and the best prior results of 14.65 (PTQ4DM) and 21.41 (Q-Diffusion); the 7.0-point margin at 4 bits is the largest in the table. CLIP slips from 26.88 to 26.70 and 26.64, the latter the best of any quantized method by 0.37. Both configurations share the same A_8 routing, so the 0.20 FID difference is attributable to the weight palette alone.

In SDXL-Turbo, FID improves from 21.63 in FP16 to 20.93 (W_8A_8) and 21.29 (W_4A_8), with CLIP holding at 26.62 and rising to 26.85, the latter 0.22 above the FP16 reference. Because this fork inherits MixDQ's activations unchanged, the margin over MixDQ – 0.50 FID at 8 bits, 3.21 at 4 bits – isolates the weight palette: the tighter the weight budget, the more a per-channel format choice buys.

In PixArt-α, Chameleon reduces FID from 27.63 to 24.33 at W_8A_8 and 22.24 at W_4A_8, beating Q-DiT by 3.36 and 1.29 respectively, and exceeding its CLIP by 0.36 at 4 bits. On the weight side the difference is the joint (g, f) search: at 8 bits it settles on INT8-asym everywhere but at $g = 32$ rather than Q-DiT's 128 (MXINT8 is never selected), and at 4 bits on $g = 32$ throughout, with NF4 – unavailable to an INT-only baseline – for 69 of the 282 linears.

Which axes the palette uses differs by backbone, as the statistics dictate: SDXL uses all of them (per-channel weights, per-layer activations, three temporal regimes, the topological MXFP8 pin); PixArt-α uses the weights and a near-uniform activation assignment (FP8 E5M2 in 97–100% of linears); SDXL-Turbo, whose single step leaves no timestep axis, uses the weight palette alone.

Appendix A.9 separates the activation axes on SDXL: against the best single fixed format, the per-layer choice is worth 0.82 FID and the timestep axis a further 0.12.

Two patterns hold across all three backbones. Every Chameleon configuration scores FID below its FP16 reference, and on SDXL and PixArt-α the 4-bit model is competitive with or better than the 8-bit model. We do not have a validated account of either: a low-pass explanation holds on PixArt-α but not on the UNets, as we discuss in Appendix A.8 of the supplement. A qualitative comparison at W_4A_8 is provided in Figure 5 in Appendix A.10.

5.3 Calibration Cost

Chameleon's calibration involves no gradient updates, no auxiliary networks, and no integer program of our own – the few-step path excepted, and only because it consumes MixDQ's precomputed mixed-precision assignment unchanged. Calibration takes at most 45 minutes on a single A100 (45 minutes for SDXL, 7 for PixArt-α and 1 for SDXL-Turbo) and under an hour for all three backbones combined (Table 2 in Appendix A.7). The routing hyperparameters – the kurtosis and SNR thresholds and the bucket count B – are fixed across every model, with no per-model tuning; only the palette is specialized per architecture, and structurally so (Section 4.6).

6 Discussion & Limitations

Our numbers come from fake quantization rather than real low-precision kernels, so we make no efficiency claims: they characterize what the format assignment costs in fidelity, and measuring its latency and memory savings on native FP8 and MX kernels remains future work. The evaluation is limited to image generation; extending the routing to video DiTs, whose temporal-attention sensitivities are uncharacterized, is a natural next step, and sub-4-bit (W_2/W_1) regimes would require quantization-aware training rather than PTQ. The palette also assumes its formats exist in hardware: FP8 and MX need Hopper- or Blackwell-class support, and on INT-only NPUs it reduces to its INT members. Per-channel mixed formats likewise do not map to one native GEMM tile; they would run as same-format sub-GEMMs after a channel permutation, or with weights dequantized in registers as existing W_4A_8 kernels do. Appendix A.13 expands these points, adding the per-channel format-metadata overhead at extreme model scale and calibration generalization to out-of-distribution prompts.

7 Conclusion

We argued that at a fixed bit-width the *number format* is not a design constant but a discrete variable, and that it should be chosen per tensor: per weight channel, per layer, and – where a timestep axis exists – per timestep bucket. The argument is structural: a format is priced for a distribution, and the distributions inside a denoising network differ across channels, across layers, and along the schedule, sliding from heavy-tailed, noise-dominated tensors at $t \to T$ to tightly clustered, structured signals at $t \to 0$, so no single format in the standard 8-bit palette is correctly priced for all of them. Tuning scales cannot fix a format mismatch.

Chameleon makes that choice adaptive. Two ahead-of-time statistics – empirical kurtosis and the closed-form diffusion SNR – populate a format lookup table that the runtime indexes with one tensor read, and a single architectural fork specializes the routing to UNet, few-step, and DiT backbones. Across all three backbones and both bit-widths, the resulting quantized configurations reduce FID below the FP16 reference while holding CLIP-score within 0.24 of it, and lead every quantized baseline on CLIP at W_4A_8 (Table 1). The framework is composable rather than competitive: Q-DiT's group-wise weights and dynamic per-sample scaling consume Chameleon's LUT unmodified, and MixDQ's activation pipeline composes with our weight palette so cleanly that its BOS-aware bypass is no longer required.

The natural next steps – Q-Sched-style sampler adaptation, real-kernel speedups on Hopper and Blackwell, video DiTs, and sub-4-bit QAT – are independent and additive.

REFERENCES

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. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=eAKmQPe3m1.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28306–28315, 2025a. doi: 10.1109/CVPR52734.2025.02636. URL https://doi.org/10.1109/CVPR52734.2025.02636.

Ruichen Chen, Keith G. Mills, and Di Niu. FP4DiT: Towards Effective Floating Point Quantization for Diffusion Transformers. *Transactions on Machine Learning Research*, 2025b. ISSN 2835-8856. URL https://openreview.net/forum?id=CcnH4mSQbP.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient Finetuning of Quantized LLMs. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115. Curran Associates, Inc., 2023. doi: 10.52202/075280-0441. URL https://doi.org/10.52202/075280-0441.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate Quantization for Generative Pre-trained Transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS.

Natalia Frumkin and Diana Marculescu. Q-Sched: Pushing the Boundaries of Few-Step Diffusion Models with Quantization-Aware Scheduling. In *Catch, Adapt, and Operate: Monitoring ML Models Under Drift Workshop*, 2026. URL https://openreview.net/forum?id=5oKzkAqVPq.

Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. PTQD: Accurate Post-Training Quantization for Diffusion Models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=Y3g1PV5R9l.

Yefei He, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. EfficientDM: Efficient Quantization-Aware Fine-Tuning of Low-Bit Diffusion Models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=UmMa3UNDAz.

Jeff Johnson. Rethinking floating point for deep learning, 2018. URL https://arxiv.org/abs/1811.01721.

Andrey Kuzmin, Mart van Baalen, Yuwei Ren, Markus Nagel, Jorn Peters, and Tijmen Blankevoort. FP8 Quantization: The Power of the Exponent. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 14651–14662. Curran Associates, Inc., 2022. URL https://openreview.net/forum?id=H3Gv7XEGzYV.

Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-Diffusion: Quantizing Diffusion Models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17489–17499, 2023. doi: 10.1109/ICCV51070.2023.01608. URL https://doi.org/10.1109/ICCV51070.2023.01608.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. In P. Gibbons, G. Pekhimenko, and C. De Sa (eds.), *Proceedings of Machine Learning and Systems*, volume 6, pp. 87–100, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/file/42a452cbafa9dd64e9ba4aa95cc1ef21-Paper-Conference.pdf.

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 David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision – ECCV 2014*, pp. 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1. URL https://doi.org/10.1007/978-3-319-10602-1_48.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan LI, and Jun Zhu. DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 5775–5787. Curran Associates, Inc., 2022. doi: 10.52202/068431-0418. URL https://doi.org/10.52202/068431-0418.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent Consistency Models: Synthesizing High-Resolution Images with Few-step Inference, 2024. URL https://openreview.net/forum?id=duBCwjb68o.

Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. A White Paper on Neural Network Quantization, 2021. URL https://arxiv.org/abs/2106.08295.

OCP Microscaling Formats Working Group. OCP Microscaling Formats (MX) Specification. Technical report, Open Compute Project, 2023. URL https://www.opencompute.org/documents/ocp-microscaling-formats-mx-v1-0-spec-final-pdf. Version 1.0.

Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On Aliased Resizing and Surprising Subtleties in GAN Evaluation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11400–11410, 2022. doi: 10.1109/CVPR52688.2022.01112. URL https://doi.org/10.1109/CVPR52688.2022.01112.

J.L. Pech-Pacheco, G. Cristobal, J. Chamorro-Martinez, and J. Fernandez-Valdivia. Diatom autofocusing in brightfield microscopy: a comparative study. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 3, pp. 314–317 vol.3, 2000. doi: 10.1109/ICPR.2000.903548. URL https://doi.org/10.1109/ICPR.2000.903548.

Said Pertuz, Domenec Puig, and Miguel Angel Garcia. Analysis of focus measure operators for shape-from-focus. *Pattern Recogn.*, 46(5):1415–1432, May 2013. ISSN 0031-3203. doi: 10.1016/j.patcog.2012.11.011. URL https://doi.org/10.1016/j.patcog.2012.11.011.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=di52zR8xgf.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, Stosic Dusan, Venmugil Elango, Maximilian Golub, Alexander Heinecke, Phil James-Roxby, Dharmesh Jani, Gaurav Kolhe, Martin Langhammer, Ada Li, Levi Melnick, Maral Mesmakhosroshahi, Andres Rodriguez, Michael Schulte, Rasoul Shafipour, Lei Shao, Michael Siu, Pradeep Dubey, Paulius Micikevicius, Maxim Naumov, Colin Verrilli, Ralph Wittig, Doug Burger, and Eric Chung. Microscaling Data Formats for Deep Learning, 2023. URL https://arxiv.org/abs/2310.10537.

Arnab Sanyal, Peter A. Beerel, and Keith M. Chugg. Neural Network Training with Approximate Logarithmic Computations. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3122–3126, 2020. doi: 10.1109/ICASSP40776.2020.9053015. URL https://doi.org/10.1109/ICASSP40776.2020.9053015.

Arnab Sanyal, Gourav Datta, Prithwish Mukherjee, Sandeep P. Chinchali, and Michael Orshansky. EntroLLM: Entropy encoded weight compression for efficient large language model inference on edge devices. In *ICASSP 2026 - 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 19742–19746, 2026. doi: 10.1109/ICASSP55912.2026.11464777.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial Diffusion Distillation. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXXXVI*, pp. 87–103, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-73015-3. doi: 10.1007/978-3-031-73016-0_6. URL https://doi.org/10.1007/978-3-031-73016-0_6.

Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-Training Quantization on Diffusion Models. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1972–1981, 2023. doi: 10.1109/CVPR52729.2023.00196. URL https://doi.org/10.1109/CVPR52729.2023.00196.

Junhyuk So, Jungwon Lee, Daehyun Ahn, Hyungjun Kim, and Eunhyeok Park. Temporal Dynamic Quantization for Diffusion Models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=D1sECc9fiG.

Junhyuk So, Jungwon Lee, Daehyun Ahn, Hyungjun Kim, and Eunhyeok Park. Temporal Dynamic Quantization for Diffusion Models. In *Advances in Neural Information Processing Systems*, volume 36, 2023b. URL https://arxiv.org/abs/2306.02316.

Junyi Wu, Haoxuan Wang, Yuzhang Shang, Mubarak Shah, and Yan Yan. PTQ4DiT: Post-training Quantization for Diffusion Transformers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=NLmAGkN6nn.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

Yijia Zhang, Lingran Zhao, Shijie Cao, Sicheng Zhang, Wenqiang Wang, Ting Cao, Fan Yang, Mao Yang, Shanghang Zhang, and Ningyi Xu. Integer or Floating Point? New Outlooks for Low-Bit Quantization on Large Language Models. In *2024 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, 2024. doi: 10.1109/ICME57554.2024.10688089. URL https://doi.org/10.1109/ICME57554.2024.10688089.

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 *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part XIV*, pp. 285–302, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-72629-3. doi: 10.1007/978-3-031-72630-9_17. URL https://doi.org/10.1007/978-3-031-72630-9_17.

Tianchen Zhao, Tongcheng Fang, Haofeng Huang, Rui Wan, Widyadewi Soedarmadji, Enshu Liu, Shiyao Li, Zinan Lin, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang. ViDiT-Q: Efficient and Accurate Quantization of Diffusion Transformers for Image and Video Generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=E1N1oxd63b.

Xingyu Zheng, Xianglong Liu, Haotong Qin, Xudong Ma, Mingyuan Zhang, Haojie Hao, Jiakai Wang, Zixiang Zhao, Jinyang Guo, and Michele Magno. BinaryDM: Accurate Weight Binarization for Efficient Diffusion Models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=YaeZwhXJ4k.

Chengwei Zhou, Sreetama Sarkar, Yuming Li, Arnab Sanyal, and Gourav Datta. OASIS: Optimized Lightweight Autoencoder System for Distributed In-Sensor computing, 2025. URL https://arxiv.org/abs/2505.02256.