AccuQuant: Simulating Multiple Denoising Steps for Quantizing Diffusion Models

Seunghoon Lee* **Jeongwoo Choi*** **Byunggwan Son** **Jaehyeon Moon** **Jeimin Jeon** **Bumsub Ham†**

School of Electrical and Electronic Engineering, Yonsei University
https://cvlab.yonsei.ac.kr/projects/AccuQuant

Abstract

We present in this paper a novel post-training quantization (PTQ) method, dubbed AccuQuant, for diffusion models. We show analytically and empirically that quantization errors for diffusion models are accumulated over denoising steps in a sampling process. To alleviate the error accumulation problem, AccuQuant minimizes the discrepancies between outputs of a full-precision diffusion model and its quantized version within a couple of denoising steps. That is, it simulates multiple denoising steps of a diffusion sampling process explicitly for quantization, accounting the accumulated errors over multiple denoising steps, which is in contrast to previous approaches to imitating a training process of diffusion models, namely, minimizing the discrepancies independently for each step. We also present an efficient implementation technique for AccuQuant, together with a novel objective, which reduces a memory complexity significantly from $\mathcal{O}(n)$ to $\mathcal{O}(1)$, where n is the number of denoising steps. We demonstrate the efficacy and efficiency of AccuQuant across various tasks and diffusion models on standard benchmarks.

1 Introduction

Diffusion models [17, 49] have shown the effectiveness for various generation tasks, including text-to-image generation [41, 43], audio generation [29], and video generation [2, 1, 12]. In the context of image generation, diffusion models train neural networks to denoise images progressively, corrupted by Gaussian noise, reversing the noise-adding process, and recovering original images. At test time, starting from a random noise, diffusion models perform a sampling process, where the trained neural networks denoise the corrupted image gradually. To generate realistic images, the sampling process typically involves lots of denoising steps, which is computationally demanding. To overcome this problem, many approaches attempt to reduce the number of denoising steps [35, 55], providing an efficient image generation process. Another line of research focuses on compressing neural networks themselves (e.g., using network quantization [25, 19] or pruning [4]) to reduce the computational cost for each denoising step.

Network quantization lowers bit-widths of full-precision weights and activations into lower ones, enabling a fixed-point computation for efficient inference. There are mainly two approaches to quantizing neural networks: Quantization-aware training (QAT) and post-training quantization (PTQ). QAT optimizes network weights and quantization parameters (e.g., step-sizes and zero-points) jointly using entire training samples, which is computationally expensive, making it hard to apply QAT for large models (e.g., ViTs [8]). PTQ has recently gained significant attention across various models [37, 26, 34, 36] due to its efficiency. In contrast to QAT, PTQ calibrates quantization parameters only without retraining the network weights, using a small subset of training samples.

*Equal contribution.
†Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

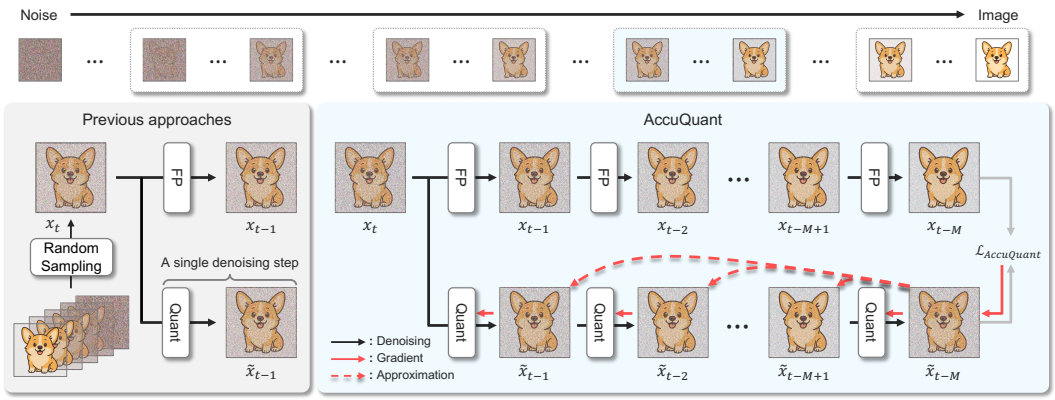


Figure 1: Calibration processes of previous approaches and AccuQuant. **Left**: Previous methods minimize the quantization error at each denoising step individually, failing to account for accumulated quantization errors during calibration. **Right**: AccuQuant addresses this problem effectively and efficiently by simulating multiple denoising steps of diffusion models, that is, aligning generated images of full-precision and quantized models over multiple denoising steps, with a memory complexity of $\mathcal{O}(1)$, independent of the number of steps.

Albeit efficient, it is challenging to directly apply PTQ methods, designed for neural networks for discriminative tasks, such as image classification, to diffusion models, since they involve a sequential process to denoise images. As the sampling process goes on, the quantization error at each denoising step is propagated. An overall quantization error can then be split into two parts: Quantization error at the current denoising step and the error accumulated along the previous steps. Most methods [45, 25, 48, 19] neglect the accumulated error, and try to minimize the quantization error at each denoising step (Fig. 1(**Left**)). Specifically, they exploit the same error-free image as inputs at every step for both full-precision and quantized models, while not considering that the input of the quantized model could contain quantization error accumulated along previous sampling steps. They then calibrate quantization parameters to minimize the difference between the outputs of these models at every step. This is not enough to reduce the overall quantization error, resulting in substantial performance degradation. Although recent works attempt to address the accumulated errors, they are able to handle single-step errors only [50] or require additional parameters for error correction [54, 14].

We introduce in this paper a novel PTQ method for diffusion models, dubbed AccuQuant, that minimizes an overall quantization error, including the ones accumulated over previous denoising steps effectively and efficiently (Fig. 1(**Right**)). To this end, AccuQuant groups a couple of denoising steps, and minimizes the difference between the outputs of quantized and full-precision models for each group to calibrate quantization parameters. That is, AccuQuant considers multiple denoising steps in the diffusion process, in contrast to previous approaches to simulating a single denoising step only [25, 14, 48, 19], namely, calibrating each step independently. This enables considering the accumulated quantization errors explicitly within each group to minimize an overall quantization error. A naive implementation of AccuQuant, however, requires substantial memory storing intermediate activations for all denoising steps within each group in order to compute gradients, resulting in a memory complexity of $\mathcal{O}(n)$ w.r.t. the number of denoising steps in each group. To address this, we propose a novel gradient approximation technique, together with a new objective, that reduces the memory complexity significantly from $\mathcal{O}(n)$ to $\mathcal{O}(1)$, enabling applying AccuQuant to large-scale diffusion models efficiently. We show that our approach achieves state-of-the-art performance across various settings, especially in terms of FID2FP32 [50], which evaluates how closely the outputs of the quantized model match those of the full-precision model. This suggest that AccuQuant aligns the quantized model with its full-precision counterpart more effectively than previous methods [25, 48, 19, 14, 50].

We summarize the main contributions as follows:

- We introduce a novel calibration method for quantizing diffusion models that aligns multiple denoising steps in the sampling processes of quantized and full-precision diffusion models, reducing accumulated quantization errors effectively.

- We present a gradient approximation technique for an efficient implementation of AccuQuant, together with a new objective, reducing the memory complexity significantly to $\mathcal{O}(1)$. We also provide a detailed analysis on quantization errors of diffusion models.
- We demonstrate the effectiveness of AccuQuant through extensive experiments across various models on standard benchmarks [16, 50, 15, 59, 38, 44].

2 Related work

2.1 Quantization for neural networks

Network quantization reduces the bit-width of weights and/or activations in a neural network. QAT methods [5, 53, 24, 21] simulate the quantization process at training time by converting the full-precision weights/activations into lower-precision representations through a rounding function. This requires retraining the neural network to quantize, which is computationally expensive. On the other hand, PTQ calibrates quantization parameters only with a small number of calibration samples. It quantizes neural networks efficiently, without involving a retraining process, but it is limited to handle outliers in weights/activations. This problem can be alleviated by clipping the outliers, adopting a outlier channel splitting technique [60], or assigning different quantization step sizes for weights/activations with large magnitudes [3, 9]. Recent PTQ methods have demonstrated the effectiveness across various architectures, including CNNs [30, 46, 20] and transformers [57, 27, 36]. In particular, they optimize a rounding function for network weights (*i.e.*, determining each weight to be rounded up or down), by exploiting the output differences of each layer [37, 51] or a Hessian-based reconstruction metric [26], before and after quantization.

2.2 Quantization for diffusion models

Most approaches to quantizing diffusion models adopt a PTQ technique, mainly due to its efficiency. Architectural characteristics of diffusion models make it difficult to directly apply existing PTQ methods, especially for extremely low bit levels. In particular, Q-Diffusion [25] shows that residual connections in diffusion models, such as U-Net [42], cause significantly different distributions for concatenated activations, and introduces a split quantization technique that performs quantization prior to the concatenation. TFMQ-DM [19] shows that previous PTQ methods, which are not designed for diffusion models, could disturb temporal features along denoising steps from original ones, and proposes to quantize temporal embedding layers of diffusion models separately.

Diffusion models apply a denoising operation iteratively over time steps to generate images, providing different distributions of activations across the denoising steps. In order to consider the time-varying characteristics of diffusion models for quantization, training samples in calibration datasets would be carefully chosen. To this end, Q-Diffusion [25] proposes to sample images uniformly along denoising steps. PTQ4DM [45] exploits a skewed normal distribution to sample more images at later denoising steps, which typically provide more realistic images.

Related to ours, PTQD [14], TAC [54] and PCR [50] attempt to alleviate the effect of accumulated quantization errors. PTQD [14] and TAC [54] analyze the relationship between the outputs of a full-precision network and its quantized counterpart. Assuming that the outputs from these networks are related at each denoising step, they correct the accumulated errors by computing the correlation coefficient and bias [14] or the reconstruction coefficient and bias [54]. However, these approaches require additional memory to store these parameters for each denoising setp, and incur computational overhead for the error-correction stage. PCR [50] instead tries to reduce the accumulated quantization error directly in a calibration phase. Similar to ours, PCR [50] calibrates quantized diffusion model at each denoising step progressively, but it exploits the generated image of a quantized model in a previous step as inputs for both full-precision and quantized models. This could not account for the differences between quantized and full-precision models across multiple denoising steps effectively, mitigating quantization errors within a single denoising step only. On the contrary, our approach exploits separate inputs for full-precision and quantized models. That is, the inputs for full-precision and quantized models come from the corresponding models in a previous denoising step, respectively. This enables minimizing the discrepancies between the quantized and full-precision models across multiple denoising steps, reducing accumulated quantization errors explicitly. Although a memory complexity of our approach is $\mathcal{O}(n)$ w.r.t. the number of denoising steps, we provide an efficient alternative with a complexity of $\mathcal{O}(1)$.

3 Method

In this section, we briefly describe diffusion models and network quantization (Sec.3.1). We then provide an analysis on quantization error in detail (Sec.3.2). Finally, we present a detailed description of AccuQuant, including a gradient approximation technique (Sec.3.3).

3.1 Preliminaries

Diffusion models. During a forward diffusion process, Gaussian noise ϵ, sampled from a normal distribution $\mathcal{N}(0, 1)$, is added to an input image x_0 progressively over time steps. Specifically, a noisy image x_t at step t can be represented as follows:

$$x_t = \sqrt{\alpha_t} x_0 + \sqrt{1 - \alpha_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \tag{1}$$

where α_t is a noise scheduling coefficient at the denoising step t. Diffusion models reverse this process, gradually removing the noise from the noisy image x_t to recover the original one x_0. To this end, a neural network estimates and removes the noise ϵ from the corrupted image x_t iteratively, until a clean image is obtained. For example, for a deterministic sampling of the DDIM sampler [49], a sampled image of x_{t-1} at step $t-1$ is computed as:

$$x_{t-1} = \sqrt{\alpha_{t-1}} \left(\frac{x_t - \sqrt{1 - \alpha_{t-1}}\epsilon_\theta(x_t, t)}{\sqrt{\alpha_t}} \right) + \sqrt{1 - \alpha_{t-1}}\epsilon_\theta(x_t, t), \tag{2}$$

where we denote by $\epsilon_\theta(x_t, t)$ the noise predicted by the neural network, parameterized by θ, for the image x_t at step t.

Network quantization. Given a floating-point value v and a target bit-width b, network quantization converts the value v into a low-precision integer $\bar{v}$ as follows:

$$\bar{v} = \text{clip}\left(\text{round}\left(\frac{v}{s} \right) + z, 0, 2^b - 1 \right), \tag{3}$$

where we denote by s and z a step-size and a zero-point, respectively. $\text{round}(\cdot)$ is a rounding function (e.g., nearest rounding or adaptive rounding [37, 26]), and $\text{clip}(\cdot, v_{\text{min}}, v_{\text{max}})$ is a clipping function that maps an input value within a range of $[v_{\text{min}}, v_{\text{max}}]$. The integer value $\bar{v}$ is then re-scaled to obtain a quantized value $\hat{v}$ as follows:

$$\hat{v} = s(\bar{v} - z). \tag{4}$$

Note that the quantization parameters of s and z are trained with all training samples for QAT, while they are computed using a small set of calibration samples for PTQ.

3.2 Quantization errors for diffusion models

In this section, we show that quantized diffusion models suffer from an error accumulation problem, where quantization error at each step is accumulated along the sampling process progressively, degrading the quality of generated images.

Let us denote by x_t and ϵ_θ an image and an estimated noise at the denoising step t for a full-precision model, respectively. We define $\tilde{x}_t$ and $\tilde{\epsilon}_\theta$ similarly for a quantized diffusion model. We first categorize the overall quantization error into two parts: a step error δ_t representing the quantization error for the estimated noise $\tilde{\epsilon}_\theta$ at step t, and an accumulated error Δ_t representing the accumulated quantization error for the image $\tilde{x}_t$ along all the previous steps. We can then represent $\tilde{x}_t$ and $\tilde{\epsilon}_\theta$ as follows:

$$\tilde{x}_t = x_t + \Delta_t, \tag{5}$$

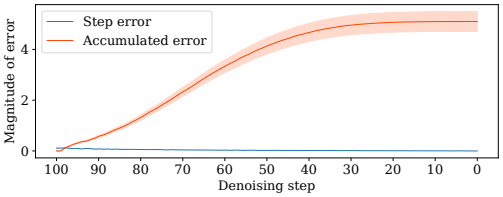
$$\tilde{\epsilon}_\theta(\tilde{x}_t, t) = \epsilon_\theta(x_t, t) + \delta_t. \tag{6}$$

By plugging Eqs. (5) and (6) into Eq. (2), we can compute the output image $\tilde{x}_{t-1}$ of the quantized model at step of $t-1$ as follows:

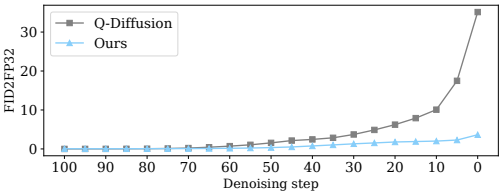
$$\tilde{x}_{t-1} = x_{t-1} + c_t\delta_t + d_t\Delta_t, \tag{7}$$

where

$$c_t = -\frac{\sqrt{\alpha_{t-1}}\sqrt{1 - \alpha_t}}{\sqrt{\alpha_t}} + \sqrt{1 - \alpha_{t-1}}, \quad d_t = \frac{\sqrt{\alpha_{t-1}}}{\sqrt{\alpha_t}}. \tag{8}$$



(a) Quantization errors on CIFAR-10 [23].



(b) FID2FP32 [50] comparisons on CIFAR-10 [23].

Figure 2: (a) We show the magnitude of the step error $c_t\delta_t$ and the accumulated error $d_t\Delta_t$ at each denoising step. The accumulated error increases drastically, while the step error remains relatively constant, suggesting that reducing the accumulated error is crucial for generating better images, compared to the step error. (b) Q-Diffusion accumulates quantization errors according to denoising steps, resulting in a very high FID2FP32 score. On the contrary, our method effectively mitigates the problem, maintaining a low score.

We show in Fig. 2a the step error of $c_t\delta_t$ and the accumulated error of $d_t\Delta_t$ in Eq. (7) at each denoising step of DDIM [49] on CIFAR-10 [23]. Specifically, we assume that δ_t is a Gaussian noise with a zero mean and unit variance, and calculate the accumulated error Δ_{t-1} recursively, *i.e.*, $\Delta_{t-1} = c_t\delta_t + d_t\Delta_t$, starting from Δ_T being 0, where T is a total number of denoising steps. We can see that the accumulated error $d_t\Delta_t$ increases drastically according to denoising steps, while the step error $c_t\delta_t$ does not. This suggests that the accumulated error for the image $\tilde{x}_t$ have a much greater impact on the quality of a generated image, compared with the step error for the estimated noise, as the sampling process goes on.

To further validate our observation, we simulate in Fig. 2b the sampling process of DDIM [49] with Q-Diffusion [25], and compute FID2FP32 [50] with generated images from full-precision and quantized models at every 5 steps. Note that FID2FP32 [50] measures the FID score [16] with the outputs of quantized and full-precision models, evaluating how closely the quantized model approximates the output from its full-precision counterpart. We can see from the blue line in Fig. 2b that FID2FP32 [50] increases accordingly along denoising steps, demonstrating once again that quantization errors are accumulated in the sampling process. Note that all diffusion models, generating images through multiple denoising steps, suffer from the error accumulation, similar to DDIM [49] in Fig. 2b, regardless of types of sampling methods. Therefore, reducing the accumulated error would be a key to maintain the quality of generated images for quantized diffusion models.

3.3 AccuQuant

To address the error accumulation problem, we introduce a novel PTQ method for quantizing diffusion models, dubbed AccuQuant, that imitates multiple denoising steps of a full-precision diffusion model, reducing the accumulated error effectively. Unlike previous approaches to focusing on individual denoising steps [45, 25, 48, 19, 54], our framework simulates the multiple denoising steps in sampling process of full-precision diffusion model for quantization. This enables considering the accumulated error during a calibration process (*i.e.*, optimizing quantization parameters), reducing the error for quantizing diffusion models.

Specifically, we group M consecutive denoising steps, splitting an entire denoising sequence into a total of T/M groups. That is, a denoising step for the l^{th} group starts with step of $T - M(l-1)$. Given an image obtained from the full-precision model at step t, denoted by x_t, we apply a denoising process M times to generate images of x_{t-M} and $\tilde{x}_{t-M}$ from full-precision and quantized models, respectively, as follows:

$$x_{t-M} = D_M(x_t, t), \quad \tilde{x}_{t-M} = \tilde{D}_M(x_t, t; s_l), \tag{9}$$

where D_M represents a denoising process for the full-precision model over M steps. $\tilde{D}_M$ is defined similarly for the quantized model with the step size of s_l for quantization. To calibrate the step-size s_l for the l^{th} group, we minimize the mean squared error between x_{t-M} and $\tilde{x}_{t-M}$ as follows:

$$s_l^* = \arg\min_{s_l} \mathcal{L}_{\text{MSE}}(x_t, t; s_l), \tag{10}$$

where

$$\mathcal{L}_{\text{MSE}}(x_t; s_l; t) = \left\| D_M(x_t, t) - \tilde{D}_M(x_t, t; s_l) \right\|_2^2. \tag{11}$$

The calibration process is then repeated sequentially for each group. By doing so, AccuQuant accounts for the overall quantization error (*i.e.*, both quantization errors at individual steps and accumulated ones within multiple steps), without introducing additional parameters or extra computational costs during a sampling process.

Gradient approximation. In order to compute gradients for calibrating quantization parameters, AccuQuant requires lots of memory to store feature maps for all denoising steps within a group. For the l^{th} group, the gradient of the objective $\mathcal{L}_{\text{MSE}}$ w.r.t. the step-size s_l is computed as follows:

$$\begin{aligned}\frac{\partial \mathcal{L}_{\text{MSE}}}{\partial s_l} &= \frac{\partial \mathcal{L}_{\text{MSE}}}{\partial \tilde{x}_{t-M}} \frac{\partial \tilde{x}_{t-M}}{\partial s_l} \\ &= \frac{\partial \mathcal{L}_{\text{MSE}}}{\partial \tilde{x}_{t-M}} \left[\sum_{m=1}^{M} g_m \frac{\partial \tilde{x}_{t-m}}{\partial s_l} \right], \end{aligned} \tag{12}$$

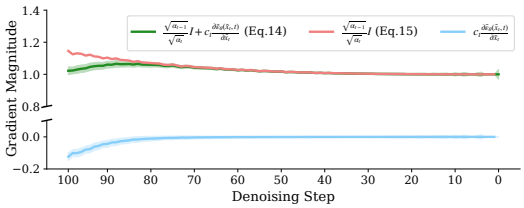


Figure 3: Plots of the gradient in Eq. (14) and its components over denoising steps. The gradients are calculated during a calibration process of AccuQuant for DDIM [49] on CIFAR-10 [23]. Shade indicates the standard deviation.

where g_m is a cumulative product of partial derivatives, across multiple denoising steps, between input and output images, as follows:

$$g_m = \begin{cases} 1 & m = M \\ \prod_{j=1}^{M-m} \frac{\partial \tilde{x}_{t-M-1+j}}{\partial \tilde{x}_{t-M+j}}, & m \neq M. \end{cases} \tag{13}$$

This suggests that we should store all intermediate feature maps, calculated across denoising steps within each group, to compute g_m. This results in a memory complexity of $\mathcal{O}(M)$ w.r.t. the number of denoising steps, which limits the scalability of AccuQuant.

To address this, we propose an efficient implementation technique to reduce the memory complexity. We can represent the gradient between consecutive denoising steps for the quantized diffusion model as follows:

$$\frac{\partial \tilde{x}_{t-1}}{\partial \tilde{x}_t} = \frac{\sqrt{\alpha_{t-1}}}{\sqrt{\alpha_t}} I + c_t \frac{\partial \tilde{\epsilon}_\theta(\tilde{x}_t, t)}{\partial \tilde{x}_t}. \tag{14}$$

We have empirically observed in Fig. 3 that the first term in Eq. (14) dominates the entire gradient, whereas the second one is negligible (see red and blue lines). Similar findings are reported in [40, 58]. Based on this observation, we approximate the gradient in Eq. (14):

$$\frac{\partial \tilde{x}_{t-1}}{\partial \tilde{x}_t} \approx \frac{\sqrt{\alpha_{t-1}}}{\sqrt{\alpha_t}} I. \tag{15}$$

Using the approximation in Eq. (15), we can represent g_m in Eq. (13) as:

$$g_m \approx \frac{\sqrt{\alpha_{t-M}}}{\sqrt{\alpha_{t-m}}} I, \quad \text{for } m = 1, 2, \ldots, M. \tag{16}$$

That is, we can simplify the gradient, computed by backpropagating through multiple denoising steps, into a scalar ratio of noise scheduling coefficients, reducing the memory complexity from $\mathcal{O}(M)$ to $\mathcal{O}(1)$. Using the approximation in Eq. (16), the gradient of the objective in Eq. (12) can then be represented as follows:

$$\frac{\partial \mathcal{L}_{\text{MSE}}}{\partial s_l} \approx \frac{\partial \mathcal{L}_{\text{MSE}}}{\partial \tilde{x}_{t-M}} \left[\sum_{m=1}^{M} \frac{\sqrt{\alpha_{t-M}}}{\sqrt{\alpha_{t-m}}} \frac{\partial \tilde{x}_{t-m}}{\partial s_l} \right]. \tag{17}$$

To fully exploit the approximated gradient of Eq. (17), we introduce a new loss function:

$$\mathcal{L}_{\text{AccuQuant}} = \sum_{m=1}^{M} \frac{\sqrt{\alpha_{t-M}}}{\sqrt{\alpha_{t-m}}} \| \texttt{sg}(x_{t-M} - \tilde{x}_{t-M}) + \tilde{x}_{t-m} - \texttt{sg}(\tilde{x}_{t-m}) \|_2^2, \tag{18}$$

such that its gradient becomes Eq. (17). Here, sg is a stop-gradient operator preventing computing gradients. Accordingly, we calibrate the quantization parameters of quantized diffusion models by optimizing the objective of Eq. (18), while considering the accumulated errors with a memory complexity of $\mathcal{O}(1)$. Detailed derivations for Eqs. (16) and (18) and the algorithm of AccuQuant can be found in the section A.

Table 1: Quantization results for unconditional image generation on CIFAR-10 (32×32) [23], LSUN-Churches (256×256) and LSUN-Bedrooms (256×256) [56]. †: The official implementation of TFMQ-DM uses a 32-bit for certain layers, such as attention modules. We modified these components, such that all layers are quantized with the same bit-width, for fair comparisons with other methods. TAC [54] does not provide source codes, and thus its performance cannot be measured.

Model	Method	Bits(W/A)	IS↑	FID↓	FID2FP32↓
DDIM CIFAR-10 (steps=100)	Full-Precision	32/32	9.09	4.26	0.00
	Q-Diffusion [25]	6/6	8.63	30.46	35.24
	TFMQ-DM† [19]	6/6	8.94	7.84	3.64
	Ours	6/6	**9.18**	**5.79**	**3.30**
	Q-Diffusion [25]	4/8	9.12	4.93	3.26
	TFMQ-DM† [19]	4/8	8.89	5.52	1.52
	TAC [54]	4/8	**9.15**	4.89	–
	Ours	4/8	9.06	**4.75**	**1.15**
	Q-Diffusion [25]	4/6	**9.47**	25.35	30.66
	TFMQ-DM† [19]	4/6	9.15	11.04	7.39
	Ours	4/6	8.99	**7.07**	**3.94**
	Q-Diffusion [25]	3/8	8.53	17.31	13.94
	TFMQ-DM† [19]	3/8	8.40	28.47	26.96
	TAC [54]	3/8	**8.86**	9.55	–
	Ours	3/8	8.76	**9.03**	**5.15**
	Q-Diffusion [25]	3/6	7.51	40.94	42.74
	TFMQ-DM† [19]	3/6	8.34	29.29	27.11
	TAC [54]	3/6	8.27	31.88	–
	Ours	3/6	**8.65**	**9.89**	**6.69**

Model	Method	Bits(W/A)	FID↓	sFID↓	FID2FP32↓
LDM-8 LSUN-Churches (steps=500)	Full-Precision	32/32	3.91	10.10	0.00
	Q-Diffusion [25]	4/8	4.49	**10.36**	0.79
	TFMQ-DM† [19]	4/8	8.43	19.83	10.14
	TAC [54]	4/8	**3.81**	–	–
	Ours	4/8	3.99	10.67	**0.66**
	Q-Diffusion [25]	4/6	45.24	25.96	40.58
	TFMQ-DM† [19]	4/6	8.60	20.77	10.07
	Ours	4/6	**7.47**	**11.07**	**4.84**
	Q-Diffusion [25]	3/8	5.68	11.25	1.49
	TFMQ-DM† [19]	3/8	9.03	21.77	10.48
	TAC [54]	3/8	7.78	–	–
	Ours	3/8	**5.04**	**10.95**	**1.26**
	Q-Diffusion [25]	3/6	47.49	27.97	42.27
	TFMQ-DM† [19]	3/6	9.84	14.20	10.19
	Ours	3/6	**8.91**	**12.58**	**5.79**
LDM-4 LSUN-Bedrooms (steps=200)	Full-Precision	32/32	3.04	7.07	0.00
	Q-Diffusion [25]	4/8	5.46	7.92	1.92
	TFMQ-DM† [19]	4/8	7.33	12.68	4.59
	TAC [54]	4/8	4.94	–	–
	Ours	4/8	**4.03**	**7.81**	**1.07**
	Q-Diffusion [25]	3/8	11.98	14.02	7.54
	TFMQ-DM† [19]	3/8	11.32	11.62	7.64
	TAC [54]	3/8	5.14	–	–
	Ours	3/8	**4.18**	**10.45**	**1.14**

4 Experiments

We describe in this section implementation details (Sec.4.1), and compare our approach with other quantization methods quantitatively and qualitatively (Sec.4.2). We then present a detailed analysis of our method (Sec.4.2). More quantitative and qualitative results can be found in the appendix.

4.1 Implementation details

Datasets and models. We apply AccuQuant to various diffusion models and perform extensive experiments on standard benchmarks for unconditional, class-conditional, and text-to-image generation tasks. For the unconditional generation task, we exploit DDIM [49] on CIFAR-10 [23], and Latent Diffusion Model (LDM) [41] on LSUN-Bedrooms and LSUN-Churches [56]. For class-conditional generation, we perform experiments using LDM [41] on ImageNet [6]. We use Stable Diffusion (SD) v1.4 [41] on MS-COCO [28] for text-to-image generation.

Quantization. We employ adaptive rounding [37, 26] for weight quantizers, following prior approaches [45, 25, 19]. For activation quantization, we split a denoising process into 20 groups for unconditional image generation, 10 groups for class-conditional image generation and 25 groups for text-to-image generation. We perform the calibration process for 50, 20, and 10 epochs on DDIM [49], LDM, and SD [41], respectively, using the Adam optimizer [22]. Following the work of [25], we generate 256 calibration samples for each group with full-precision models, maintaining the total number of the samples consistent with that of [25] across all experiments. More detailed settings can be found in the section B.

Evaluation metrics. Following the previous approaches [25, 19, 54], we evaluate 50K images for unconditional/class-conditional generation with Inception Score (IS) [44], Frechet Inception Distance (FID) [16], and sFID [38]. We also measure PSNR, SSIM, and LPIPS [59] between images generated by full-precision and quantized models for class-conditional image generation. For text-to-image generation, we generate 5K images, and evaluate them with FID [16] and CLIP scores [15], following the work of [50]. The ViT-L/14 is used to compute CLIP scores [15]. We also

Table 2: Quantization results for class-conditional image generation on ImageNet (256×256) [6].

Model	Method	Bits(W/A)	FID↓	sFID↓	IS↑	FID2FP32↓	LPIPS↓	PSNR↑	SSIM↑
LDM-4 ImageNet (steps=20)	Full-Precision	32/32	11.13	7.85	368.19	0.00	0.00	0.00	0.00
	Q-Diffusion [25]	4/8	9.55	13.50	339.44	1.71	0.1912	23.6703	0.8391
	PTQD [14]	4/8	9.99	8.43	**361.95**	0.84	0.1598	24.0672	0.8614
	TFMQ-DM [19]	4/8	9.80	**7.19**	358.38	1.25	0.1702	23.6309	0.8567
	Ours	4/8	**9.39**	7.41	356.48	**0.65**	**0.1585**	**24.4338**	**0.8713**
	Q-Diffusion [25]	3/8	7.57	12.43	266.65	11.26	0.3741	18.4588	0.6532
	PTQD [14]	3/8	7.75	9.51	**308.47**	6.30	0.3446	19.2485	0.6728
	TFMQ-DM [19]	3/8	7.96	8.77	295.46	4.47	0.3942	16.8180	0.7200
	Ours	3/8	**6.61**	**8.65**	301.06	**3.86**	**0.2937**	**19.9392**	**0.7649**
	Q-Diffusion [25]	3/6	28.83	39.30	99.55	41.25	0.4666	17.9855	0.6358
	PTQD [14]	3/6	9.15	11.42	244.43	13.85	0.3852	18.1508	0.6662
	TFMQ-DM [19]	3/6	8.17	10.71	237.32	10.06	0.3928	17.4886	0.7183
	Ours	3/6	**5.95**	**8.61**	**282.87**	**6.31**	**0.3296**	**19.0135**	**0.7296**

Table 3: Quantization results for text-to-image generation on MS-COCO (512×512) [28]. MP: Mixed-precision quantization.

Model	Method	Bits(W/A)	FID↓	FID2FP32↓	CLIP Score↑
Stable Diffusion v1.4 MS-COCO (steps=50)	Full-Precision	32/32	27.50	0.00	26.46
	Q-Diffusion [25]	4/8	27.87	20.42	26.15
	PTQ4DM [45]	4/8	25.64	17.73	26.25
	PCR [50]	4/8.1 (MP)	23.86	14.39	26.35
	PCR [50]	4/8.4 (MP)	**22.04**	14.25	26.48
	Ours	4/8	22.73	**10.99**	**26.85**

use FID2FP32 [50] computing FID of generated images by quantized models, but with the images from full-precision models as a reference, measuring accumulated errors from quantized models for all experiments.

4.2 Results

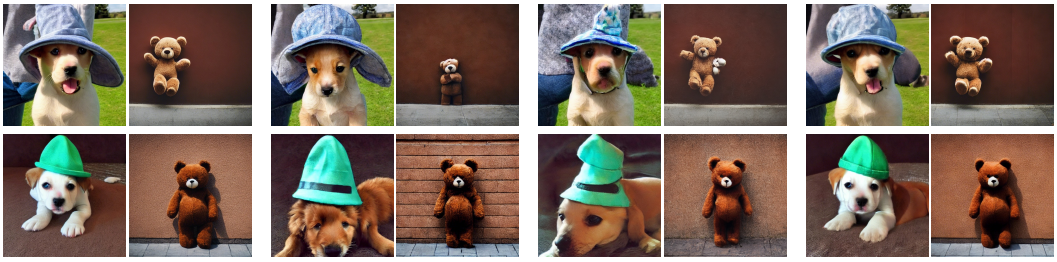
Quantitative results. We show in Tables 1-3 quantitative comparisons of our method and the state of the art [25, 19, 54] for unconditional image generation (*i.e.*, CIFAR-10 [23], LSUN-Bedrooms, and LSUN-Churches [56]), class-conditional image generation (*i.e.*, ImageNet [6]) and text-to-image generation (*i.e.* MS-COCO [28]), respectively. For fair comparison, we report the results of TFMQ-DM [19], with official source codes provided by authors, under the same settings as ours. We refer to the results from the papers for other methods that do not provide official implementations.

We summarize our findings as follows: (1) AccuQuant outperforms all previous approaches [45, 25, 14, 19, 54, 54], specially designed to quantize diffusion models, by significant margins in terms of FID2FP32 [50]. Specifically, it provides better results than PCR [50], even with lower bit-widths. This demonstrates that AccuQuant reduces accumulated errors effectively, and better maintains the behavior of full-precision models compared to other methods. (2) AccuQuant achieves significant performance gains, especially in low-bit settings for activation quantization. Note that low-bit settings are more vulnerable to accumulated errors, due to the limited representational capacity. This demonstrates the robustness of AccuQuant and its ability to maintain high performance, even under constrained conditions. (3) AccuQuant outperforms the state-of-the-art methods [25, 19, 54] in various standard benchmarks, verifying that considering multiple denoising steps in a sampling process is effective to reduce accumulated errors for quantizing diffusion models.

Qualitative results. We provide in Fig. 4 visual comparisons of generated images by full-precision model, Q-Diffusion [25], TFMQ-DM [19] and AccuQuant on LSUN-Bedrooms and LSUN-Churches [56] for unconditional image generation. We can see that AccuQuant provides more realistic images, and they are more close to the results from full-precision models, verifying once more that AccuQuant minimizes the discrepancies between full-precision and quantized models effectively. We also show Fig. 5 generated images, conditioned on text prompts, by SD v1.4 [41] and

Full precision Q-Diffusion [25] TFMQ-DM [19] Ours

Figure 4: Visual comparisons of generated images on (**Top**) LSUN-Bedrooms [56] and (**Bottom**) LSUN-Churches [56] for unconditional image generation under a 3/8-bit setting.



Full-Precision Q-Diffusion [25] PCR ($\tau = 0.20$) [50] Ours

Figure 5: Visual comparisons of generated images conditioned on text prompts. We generate images by SD v1.4 [41] and its quantized versions using PCR [50] under a 4/8.4-bit setting and Q-Diffusion [25], AccuQuant under a 4/8-bit setting. $\tau = 0.20$ indicates setting 20% of entire denoising steps into 10 bits. The prompts corresponding to each column are: *"A puppy wearing a hat"* and *"A cute teddy bear in front of a plain wall, warm and brown fur."*

its quantized versions using Q-Diffusion [25], PCR [50] and AccuQuant on MS-COCO [28]. We can observe that AccuQuant generates high-quality images that closely resemble those obtained by the full-precision model, compared to Q-Diffusion [25] and PCR [50]. Note that PCR [50] exploits more bit-widths than ours, demonstrating the effectiveness of our approach to minimizing accumulated quantization errors. For example, the detailed shape of the hat, and the pose of the dog and teddy bear are preserved well for our method, while Q-Diffusion [25] and PCR [50] do not. More qualitative results can be found in the appendix.

4.3 Discussion

Analysis of a group size. We compare in Table 4 quantization results of AccuQuant, with different group sizes, for DDIM [49] on CIFAR-10 [23]. We can see that AccuQuant performs better accordingly with an increase of a group size from 1 to 5, confirming once again that it is effective for quantizing diffusion models to account for the behaviors of full-precision and quantized models within multiple denoising steps. This also suggests that considering accumulated errors is not enough with few denoising steps. On the other hand, the quantization performance degrades, when the number of denoising steps is too large. This indicates that it is hard to estimate appropriate quantization parameters reducing the accumulated error across many denoising steps. To this end, the group size is treated as a hyperparameter that balances two objectives: (1) sufficiently capturing accumulated errors across multiple timesteps, and (2) ensuring stable optimization of quantization parameters. As shown in Table 4, if the group size is too small, it may fail to capture the long term error accumulation; if it is too large, the optimization becomes unstable as a single set of parameters must account for the diverse behaviors of many timesteps. The optimal group size can vary depending on the model and dataset, but we empirically found that dividing the timesteps into 10 to 20 groups yields consistently strong results across our experiments.

Table 4: Quantitative comparisons of AccuQuant under a 6/6-bit setting with varying group size.

Group size	IS ↑	FID ↓	FID2FP32 ↓
1	9.17	6.96	5.62
2	9.02	6.82	4.48
5	**9.18**	**5.79**	**3.30**
10	9.13	6.18	4.30
50	8.69	14.52	14.19
100	7.86	50.11	57.96

Table 5: Comparison of the full gradient ($\frac{\partial \tilde{x}_{t-1}}{\partial \tilde{x}_t}$), Jacobian term ($c_t \frac{\partial \tilde{\epsilon}_\theta(\tilde{x}_t, t)}{\partial \tilde{x}_t}$), and our approximation. We quantize DDIM [49] with CIFAR-10 [23] under W4A8 settings and report the mean and the entire range of the gradient at every 20 timestep.

Timestep	**100**	**80**	**60**	**40**	**20**
$\frac{\partial \tilde{x}_{t-1}}{\partial \tilde{x}_t}$	1.0221 **±0.112**	1.0586 **±0.059**	1.0273 **±0.034**	1.0082 **±0.034**	1.0008 **±0.071**
$c_t \frac{\partial \tilde{\epsilon}_\theta(\tilde{x}_t, t)}{\partial \tilde{x}_t}$	-0.1236 **±0.112**	-0.0097 **±0.059**	-0.0015 **±0.034**	-0.0006 **±0.034**	-0.0002 **±0.071**
$\sqrt{\frac{\alpha_{t-1}}{\alpha_t}}$	1.1457	1.0683	1.0288	1.0088	1.0010

Table 6: Quantitative comparisons of AccuQuant with and without gradient approximation. We quantize DDIM [49] on CIFAR-10 [23] with group size 5.

Bits (W/A)	**FID ↓ / FID2FP32 ↓**	
	w/o Approx.	w/ Approx.
6/6	6.13 / 4.02	**5.79 / 3.30**
4/8	5.26 / 2.32	**4.75 / 1.15**
4/6	7.21 / 5.18	**7.07 / 3.94**
3/8	9.72 / 5.34	**9.03 / 5.15**
3/6	10.26 / 7.80	**9.89 / 6.69**

Analysis of gradient approximation. We show in Table 6 quantitative comparisons of AccuQuant with and without the gradient approximation technique. It shows that 1) AccuQuant with our gradient approximation technique even provides better results quantitatively, with much less memory, compared with those obtained without using the approximation, and 2) the performance gains are more significant for FID2FP32. This suggests that gradients of Eq. (14) are corrupted by substantial noise before the quantization parameters s_l of Eq. (10) converge, which makes the calibration process unstable. Specifically, we show in Table 5 the same statistics as in Fig. 3 with the entire range to better illustrate the variability of the gradients. As shown in the Table 5, Jacobian component (i.e., Row 2 in Table 5) is significantly smaller in average magnitude compared to the dominant scalar coefficient (i.e., Row 3 in Table 5), but it exhibits a high dynamic range that introduce substantial noise into the full gradient (i.e., Row 1 in Table 5). This high-variance noise causes the quantization parameters to update inconsistently at each step, leading to an unstable and unreliable calibration process. In contrast, our gradient approximation exploit only the dominant scalar coefficient (i.e., Row 3 in Table 5) by omitting the highly dynamic Jacobian term. This leads to more stable convergence during calibration and ultimately yields more optimal results, as demonstrated in Table 6.

5 Limitation

AccuQuant requires multiple denoising steps during the calibration phase. While AccuQuant effectively mitigates accumulated errors and achieves strong performance even in few-step settings such as 20 steps on ImageNet [6], it may have limited efficacy when applied to diffusion models employing only 1–2 denoising steps [35, 55]. In addition, AccuQuant has a hyperparameter, the group size, which is currently fixed across all groups. Exploring adaptive strategies to dynamically determine the optimal group size for each group would be an interesting direction for future work.

6 Conclusion

We have shown a detailed analysis on quantization errors of diffusion models that the errors are accumulated over denoising steps. Based on this, we have introduced a novel PTQ method, dubbed AccuQuant, that alleviates the error accumulation problem by simulating multiple denoising steps in a sampling process of a diffusion model. We have also presented a gradient approximation technique to reduce the computational overhead of storing gradients for intermediate activations along the denoising steps. We have demonstrated that AccuQuant outperforms state-of-the-art PTQ methods across various bit-widths on standard benchmarks.

Acknowledgments and Disclosure of Funding

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No.RS-2022-00143524, Development of Fundamental Technology and Integrated Solution for Next-Generation Automatic Artificial Intelligence System, No.RS-2025-09942968, AI Semiconductor Innovation Lab(Yonsei University)), the National Research Foundation of Korea(NRF) grants funded by the Korea government(MSIT) (No. 2023R1A2C2004306, RS-2025-02216328), Samsung Electronics Co., Ltd (IO240520-10013-01), and the Yonsei Signature Research Cluster Program of 2025 (2025-22-0013).

References

[1] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[2] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *CVPR*, 2023.

[3] Zhaowei Cai, Xiaodong He, Jian Sun, and Nuno Vasconcelos. Deep learning with low precision by half-wave gaussian quantization. In *CVPR*, 2017.

[4] Thibault Castells, Hyoung-Kyu Song, Bo-Kyeong Kim, and Shinkook Choi. LD-Pruner: Efficient pruning of latent diffusion models using task-agnostic insights. In *CVPR*, 2024.

[5] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, I Pierce, Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. PACT: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *NeurIPS*, 2021.

[8] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[9] Jun Fang, Ali Shafiee, Hamzah Abdel-Aziz, David Thorsley, Georgios Georgiadis, and Joseph H Hassoun. Post-training piecewise linear quantization for deep neural networks. In *ECCV*, 2020.

[10] Tomer Garber and Tom Tirer. Image restoration by denoising diffusion models with iteratively preconditioned guidance. In *CVPR*, 2024.

[11] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016.

[12] Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. AnimateDiff: Animate your personalized text-to-image diffusion models without specific tuning. In *ICLR*, 2023.

[13] Yefei He, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Efficientdm: Efficient quantization-aware fine-tuning of low-bit diffusion models. In *ICLR*, 2024.

[14] Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Ptqd: Accurate post-training quantization for diffusion models. In *NeurIPS*, 2023.

[15] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *EMNLP*, 2021.

[16] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[18] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPSW*, 2021.

[19] Yushi Huang, Ruihao Gong, Jing Liu, Tianlong Chen, and Xianglong Liu. TFMQ-DM: Temporal feature maintenance quantization for diffusion models. In *CVPR*, 2024.

[20] Yongkweon Jeon, Chungman Lee, Eulrang Cho, and Yeonju Ro. Mr.BiQ: Post-training non-uniform quantization based on minimizing the reconstruction error. In *CVPR*, 2022.

[21] Dohyung Kim, Junghyup Lee, and Bumsub Ham. Distance-aware quantization. In *ICCV*, 2021.

[22] Diederik P Kingma. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[23] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical report*, 2009.

[24] Junghyup Lee, Dohyung Kim, and Bumsub Ham. Network quantization with element-wise gradient scaling. In *CVPR*, 2021.

[25] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-Diffusion: Quantizing diffusion models. In *ICCV*, 2023.

[26] Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: Pushing the limit of post-training quantization by block reconstruction. In *ICLR*, 2021.

[27] Zhikai Li, Junrui Xiao, Lianwei Yang, and Qingyi Gu. RepQ-ViT: Scale reparameterization for post-training quantization of vision transformers. In *ICCV*, 2023.

[28] 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 *ECCV*, 2014.

[29] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. AudioLDM: Text-to-audio generation with latent diffusion models. In *ICML*, 2023.

[30] Jiawei Liu, Lin Niu, Zhihang Yuan, Dawei Yang, Xinggang Wang, and Wenyu Liu. PD-Quant: Post-training quantization based on prediction difference metric. In *CVPR*, 2023.

[31] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *ICLR*, 2022.

[32] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015.

[33] Cheng Lu, Zhou Yuhao, 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 *NeurIPS*, 2022.

[34] Chengtao Lv, Hong Chen, Jinyang Guo, Yifu Ding, and Xianglong Liu. PTQ4SAM: Post-training quantization for segment anything. In *CVPR*, 2024.

[35] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *CVPR*, 2023.

[36] Jaehyeon Moon, Dohyung Kim, Junyong Cheon, and Bumsub Ham. Instance-aware group quantization for vision transformers. In *CVPR*, 2024.

[37] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *ICML*, 2020.

[38] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W Battaglia. Generating images with sparse representations. In *ICML*, 2021.

[39] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

[40] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*, 2023.

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[42] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[43] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. DreamBooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *CVPR*, 2023.

[44] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *NeurIPS*, 2016.

[45] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *CVPR*, 2023.

[46] Gil Shomron, Freddy Gabbay, Samer Kurzum, and Uri Weiser. Post-training sparsity-aware quantization. In *NeurIPS*, 2021.

[47] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[48] Junhyuk So, Jungwon Lee, Daehyun Ahn, Hyungjun Kim, and Eunhyeok Park. Temporal dynamic quantization for diffusion models. In *NeurIPS*, 2024.

[49] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2020.

[50] Siao Tang, Xin Wang, Hong Chen, Chaoyu Guan, Zewen Wu, Yansong Tang, and Wenwu Zhu. Post-training quantization with progressive calibration and activation relaxing for text-to-image diffusion models. In *ECCV*, 2024.

[51] Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. QDrop: Randomly dropping quantization for extremely low-bit post-training quantization. In *ICLR*, 2022.

[52] Junyi Wu, Haoxuan Wang, Yuzhang Shang, Mubarak Shah, and Yan Yan. Ptq4dit: Post-training quantization for diffusion transformers. In *NeurIPS*, 2024.

[53] Jiwei Yang, Xu Shen, Jun Xing, Xinmei Tian, Houqiang Li, Bing Deng, Jianqiang Huang, and Xian-sheng Hua. Quantization networks. In *CVPR*, 2019.

[54] Yuzhe Yao, Feng Tian, Jun Chen, Haonan Lin, Guang Dai, Yong Liu, and Jingdong Wang. Timestep-aware correction for quantized diffusion models. In *ECCV*, 2024.

[55] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, 2024.

[56] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[57] Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. PTQ4ViT: Post-training quantization framework for vision transformers with twin uniform quantization. In *ECCV*, 2020.

[58] Kexun Zhang, Xianjun Yang, William Yang Wang, and Lei Li. Redi: Efficient learning-free diffusion inference via trajectory retrieval. In *ICML*, 2023.

[59] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[60] Ritchie Zhao, Yuwei Hu, Jordan Dotzel, Chris De Sa, and Zhiru Zhang. Improving neural network quantization without retraining using outlier channel splitting. In *ICML*, 2019.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We introduce a quantization framework for diffusion models that reduces accumulated errors. We demonstrate the effectiveness of our framework on various bit-widths, model architectures, datasets, and baseline quantization methods.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We describe the limitations of AccuQuant in the Appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We conduct a theoretical analysis of the accumulated quantization error in diffusion model quantization. We visualize the accumulated quantization error in Fig. 2a and Fig. 2b to validate our assumptions, and we also include the detailed derivation of AccuQuant in the Appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the detailed AccuQuant algorithm in the Appendix. We also specify the various experimental settings such as group size, optimizer, epochs, batch size, learning rate, dataset, and models for each configuration in both the Sec.4 and the Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We do not release our code due to copyright restrictions. However, we include detailed pseudocode of our algorithm, dataset descriptions, and experimental guidelines in Sec. 4, enabling straightforward reproduction.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide detailed calibration and inference settings, including group size, batch size, optimizer, learning rate, number of epochs, dataset, and model in both Sec. 4 and Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide standard deviations in the Fig. 3 and appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We specify the GPUs used for our experiments and report the memory usage in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and our research conforms to it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our work presents a framework for quantizing diffusion models. We believe that it has no direct positive or negative societal impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our work does not involve any data or models that pose a high risk of misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly credited the owners of all assets used in this work (e.g., CIFAR-10 [23], LSUN [56], ImageNet [6], and MS-COCO [28]) and clearly stated each license and terms of use.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not introduce any new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not conduct any crowdsourcing experiments or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We do not conduct any crowdsourcing experiments or research with human subjects.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components. We use LLMs only for grammer checking.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Detailed algorithm for AccuQuant

A.1 Algorithm table for AccuQuant

Algorithm 1 Pseudo code of AccuQuant.

```
# M : group size
# x_T : random gaussian noise
# fp_model : full-precision model
# quant_model : quantized model
# gather_output() : one iteration of diffusion denoising step
# sg() : stop gradient operation

# Initialize
x_t = x_T
t = T
group_index = 0

for i in range(number_of_groups):
    # Gather x_(t-M) from Full Precision Model
    with torch.no_grad():
        fp_x_M = gather_output(fp_model,x_t,t)

    # Reconstruction stage
    quant_model.set_group(group_index)
    optimizer=torch.optim.Adam(quant_params,learning_rate)

    for epoch in range(epochs):
        # Gather \tilde x_(t-M) with stop gradient
        with torch.no_grad():
            sg_quant_x_M = gather_output(quant_model,x_m,m,quant_params)

        optimizer.zero_grad()

        # Accumulate the gradient
        for m in range(M):
            # Calculate gradient scaling factor
            g_m =sqrt_alpha_M / sqrt_alpha_m

            # Gather \tilde x_(t-m)
            quant_x_m = gather_output(quant_model,x_m,m,quant_params)

            # Compute \tilde x_(t-M) with stop gradient
            quant_x_M = sg_quant_x_M - sg(quant_x_m) + quant_x_m

            # Compute Loss for current step
            loss_accuquant = torch.mse(fp_x_M - quant_x_M) * g_m

            # Update quantization parameters with accumulated gradients
            loss_accuquant.backward()

        optimizer.step()

    # update indices
    x_t = fp_x_M
    t = t-M
    group_index += 1
```

A.2 Detailed derivations of Eq. (15)

We provide a detailed derivation of Eq. (15). We can represent the gradient between consecutive denoising steps for the quantized diffusion model as follows:

$$\frac{\partial \tilde{x}_{t-1}}{\partial \tilde{x}_t} = \frac{\sqrt{\alpha_{t-1}}}{\sqrt{\alpha_t}} I + \left(-\frac{\sqrt{\alpha_{t-1}}\sqrt{1-\alpha_t}}{\sqrt{\alpha_t}} + \sqrt{1-\alpha_{t-1}} \right) \frac{\partial \tilde{\epsilon}_\theta(\tilde{x}_t, t)}{\partial \tilde{x}_t}. \tag{19}$$

Here, I is the identity matrix, and $\tilde{\epsilon}_\theta(\tilde{x}_t, t)$ is the output of the quantized model. We have empirically observed in Fig. 3 that the first term in Eq. (14) dominates the entire gradient, whereas the second one is negligible. Based on this observation, we approximate the gradient in Eq. (14) by omitting the second term as follows:

$$\frac{\partial \tilde{x}_{t-1}}{\partial \tilde{x}_t} \approx \frac{\sqrt{\alpha_{t-1}}}{\sqrt{\alpha_t}} I. \tag{20}$$

Using Eq. (20) in Eq. (16), we obtain the approximated g_m as follows:

$$g_m = \begin{cases} 1 & m = M \\ \prod_{j=1}^{M-m} \frac{\partial \tilde{x}_{t-M-1+j}}{\partial \tilde{x}_{t-M+j}}, & m \neq M, \end{cases} \tag{21}$$

$$\approx \begin{cases} 1 & m = M \\ \prod_{j=1}^{M-m} \frac{\partial x_{t-M-1+j}}{\partial x_{t-M+j}}, & m \neq M, \end{cases} \tag{22}$$

$$\approx \begin{cases} 1 & m = M \\ \prod_{j=1}^{M-m} \frac{\sqrt{\alpha_{t-M-1+j}}}{\sqrt{\alpha_{t-M+j}}} I, & m \neq M, \end{cases} \tag{23}$$

$$= \frac{\sqrt{\alpha_{t-M}}}{\sqrt{\alpha_{t-m}}} I, \quad \text{for } m = 1, 2, \ldots, M. \tag{24}$$

A.3 Detailed derivation of the objective in Eq. (18) with the gradient approximation

We derive a loss function in Eq. (18) incorporating our gradient approximation of Eq. (24). First, by substituting Eq.(24) into Eq. (12), we obtain:

$$\frac{\partial \mathcal{L}_{\text{MSE}}}{\partial s_l} = \frac{\partial \mathcal{L}_{\text{MSE}}}{\partial \tilde{x}_{t-M}} \left[\sum_{m=1}^{M} g_m \frac{\partial \tilde{x}_{t-m}}{\partial s_l} \right] \tag{25}$$

$$\approx \frac{\partial \mathcal{L}_{\text{MSE}}}{\partial \tilde{x}_{t-M}} \left[\sum_{m=1}^{M} \frac{\sqrt{\alpha_{t-M}}}{\sqrt{\alpha_{t-m}}} \frac{\partial \tilde{x}_{t-m}}{\partial s_l} \right] \tag{26}$$

$$= 2(\tilde{x}_{t-M} - x_{t-M})^\top \left[\sum_{m=1}^{M} \frac{\sqrt{\alpha_{t-M}}}{\sqrt{\alpha_{t-m}}} \frac{\partial \tilde{x}_{t-m}}{\partial s_l} \right] \tag{27}$$

$$= \sum_{m=1}^{M} \frac{\sqrt{\alpha_{t-M}}}{\sqrt{\alpha_{t-m}}} \left[2(\tilde{x}_{t-M} - x_{t-M})^\top \frac{\partial \tilde{x}_{t-m}}{\partial s_l} \right]. \tag{28}$$

Note that we omit identity matrix in Eq. (24) for simple notation. From Eq. (28), we derive a loss function whose gradient matches the derived expression by adding x_{t-m} and subtracting a detached version of x_{t-m} within the squared term:

$$\mathcal{L}_{\text{AccuQuant}} = \sum_{m=1}^{M} \frac{\sqrt{\alpha_{t-M}}}{\sqrt{\alpha_{t-m}}} \| \texttt{sg}(x_{t-M} - \tilde{x}_{t-M}) + \tilde{x}_{t-m} - \texttt{sg}(\tilde{x}_{t-m}) \|_2^2. \tag{29}$$

In this formulation, we compute x_{t-M} and $\tilde{x}_{t-M}$ before updating gradients without saving intermediate feature maps and treat it as a constant during optimization. We then compute loss term inside the summation and accumulate its gradients progressively over multiple denoising steps. After completing M denoising steps, we update the step size using the cumulative gradient. The detailed pipeline of our method is provided in Algorithm 1.

B Implementation details

For weight quantization, we adopt a uniform quantizer identical for all denoising steps, following Q-Diffusion [25], since the weights remain invariant across different denoising steps. For the CIFAR-10 [23], we employ the DDIM [49] with 100 total denoising steps, grouping them into 20 sets of 5 denoising steps each and setting learning rate in $\{1 \times 10^{-3}, 4 \times 10^{-4}\}$. For the LSUN-Bedroom [56], we use LDM-4 [41] and the DDIM sampler [49] with 200 total denoising steps, grouping them into 20 sets of 10 denoising steps each and setting learning rate into 4×10^{-5}. For the LSUN-Church [56], we use LDM-8 [41] and the DDIM sampler [49] with 500 total denoising steps, grouping them into 20 sets of 25 denoising steps each and setting learning rate into 4×10^{-5}. For the ImageNet [6], we employ LDM-4 [41] and the DDIM sampler [49] with 20 total denoising steps and forming 10 groups and setting learning rate into 1×10^{-3}. For text-to-image generation, we employ Stable-Diffusion v1.4 [3] with the PNDM sampler [31], using 50 total denoising steps and forming 25 groups and setting learning rate into 1×10^{-5}. We set the calibration batch size to 8 for DDIM [49] and LDMs [41], and to 1 for Stable Diffusion [41]. All other experimental settings are identical to those of Q-Diffusion [25]. We also retain the default settings [41] during the sampling phase, ensuring that we can conduct all experiments on a single A100 (80 GB) GPU. Also, to ensure a fair comparison, we quantize every layer of TFMQ-DM [19], such as attention-modules and skip-connections for unconditional image generation. Finally, we evaluate the FID [16] and IS [44] scores, including the FID2FP32 metric [50], using the torch-fidelity library [4] and the official Guided Diffusion [7] codebase.

C Detailed analysis of quantitative results

In this section, we analyze the quantitative results presented in Tables 1–3. Firstly, Table 1 shows that AccuQuant outperforms other methods in most settings. In particular, AccuQuant achieves overwhelmingly superior performance on the FID2FP32 [50] metric across all bit-widths, and this effect becomes more pronounced at lower bit-widths (*e.g., 3/6 bits*). Furthermore, AccuQuant achieves higher scores on the FID [16], sFID [38], and IS [44] metrics compared to previous methods [25, 19, 54], which are not designed for reducing the accumulated quantization error. These findings indicate that AccuQuant effectively reduces accumulated quantization error allowing better performance and confirms that minimizing the accumulated quantization error constitutes a crucial factor in diffusion model quantization. Secondly, for the class-conditional image generation task reported in Table 2, we draw on previous work [54] showing that FID [16] does not perform reliably on ImageNet [6] and therefore we report LPIPS [59], PSNR, and SSIM metrics to quantify differences between the full precision and quantized models. Likewise to FID2FP32 [50], these metrics measure how closely the quantized model's outputs match those of the full precision model. We find that AccuQuant outperforms previous methods in terms of FID2FP32 [50], LPIPS [59], PSNR, and SSIM across all bit-widths, demonstrating that AccuQuant generates images that most closely resemble the full precision model's outputs while maintaining high visual quality. These results imply that accumulated quantization error remains critical even when employing few denoising steps, such as 20. Furthermore, even compared to PTQD [14], which is designed to reduce accumulating quantization error, AccuQuant outperforms it in FID2FP32 [50], LPIPS [59], PSNR, and SSIM across all bit widths, demonstrating the superiority of our framework in explicitly reducing accumulating quantization error. Finally, the text-to-image generation results in Table 3 reveal that, despite PCR [50] use of more bit-widths (*i.e.,* 8.4 bits for activations) and adopting independent quantizer at every denoising steps, AccuQuant achieves a better FID2FP32 [50] score. Additionally, AccuQuant attains the highest CLIP-Score [15], indicating that it not only reproduces images similar to the full precision model but also aligns them effectively with the text prompts. To this end, our findings demonstrate that the AccuQuant framework outperforms existing methods for reducing accumulated quantization error, and that AccuQuant not only achieves results comparable to those of the full-precision model but also confirms the fidelity term by resulting superior performance on FID [16], sFID [38], and CLIP score [15].

[3] https://huggingface.co/CompVis/stable-diffusion-v-1-4-original

[4] https://github.com/toshas/torch-fidelity

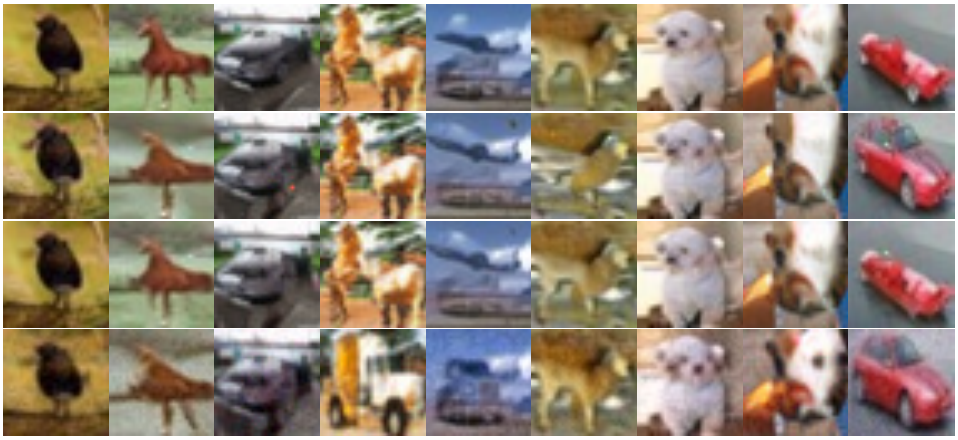


Figure 6: Visual comparisons of DDIM [49] for unconditional image generation on CIFAR-10 [23] under various group sizes. Both weights and activations were quantized to 6 bits for a visually distinct and clear comparison. From top to bottom, the presented sequences correspond to the full-precision model, our method with group sizes of 1, 5, and 100.

Table 7: Quantitative comparisons of AccuQuant with and without the gradient approximation via various group sizes. We quantize DDIM [49] on CIFAR-10 [23] under a 4/8-bit setting.

Group size	FID ↓ / FID2FP32 ↓	
	w/o Approximation	w/ Approximation
1	6.88 / 3.11	6.88 / 3.11
5	5.26 / 2.32	**4.75** / **1.15**
10	5.27 / **1.62**	**5.07** / 1.88
25	5.73 / 1.49	**5.24** / 1.49

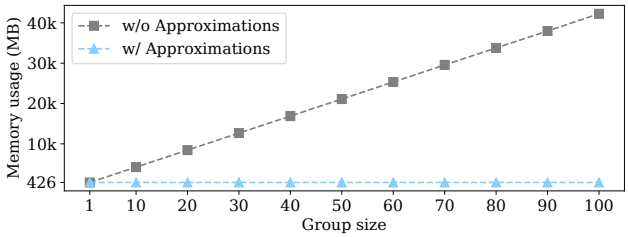
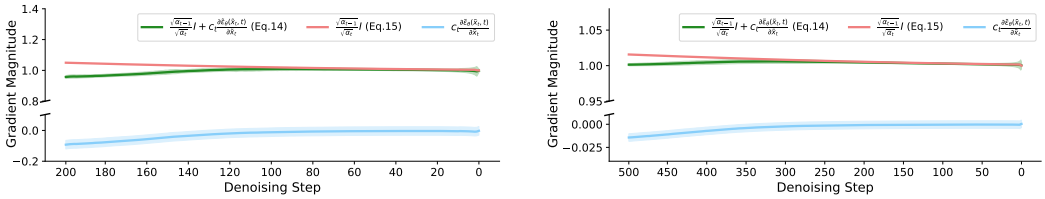


Figure 7: Comparisons of a total amount of memory of AccuQuant with and without the gradient approximation. We measure the memory usage during the calibration process of AccuQuant under a 6/6-bit setting for DDIM [49] on CIFAR-10 [23].

D Qualitative results for group sizes

In this section, we present the qualitative results for various group sizes, as reported in Table 4. We observe from Fig. 6 that selecting an appropriate group size (e.g., group size of 5) generate images that closely resembling those of the full-precision model, under low bit-widhts. For example, in the first row of generated horse images, we can see that when the group size is set to 1, the quantized model fails to adequately account for accumulated errors, leading to the generation of distorted and unrealistic shapes. Similarly, when the group size is excessively large (e.g., group size of 100), the quantized model is unable to effectively manage the substantial accumulation of errors, resulting in unrealistic image outputs. In contrast, with an appropriately chosen group size (e.g., group size of 5), the accumulated errors are effectively mitigated, allowing the generated images to maintain a structure and appearance comparable to those of the full-precision model. In the dog images of column 6, using a group size of 5 yields images that most closely match the full-precision model in both texture and shape, outperforming group sizes 1 and 100. This indicates that an appropriate group sizes is a crucial factor to mitigate the accumulating quantization error.



LSUN-Bedroom LSUN-Church

Figure 8: Plots of the gradient in Eq. (19) and its components over denoising steps. The gradients are calculated during a calibration process of AccuQuant for LSUN-Bedroom [56] on the left and LSUN-Church [56] on the right.

Table 8: Quantization results for unconditional image generation with DPM-Solver++ sampler [33] on CIFAR-10 (32×32) [23].

Model	Method	Bits(W/A)	IS↑	FID↓	FID2FP32↓
DPM-Solver++ CIFAR-10 (steps=50)	Full-Precision	32/32	9.42	3.59	0.00
	Q-Diffusion [25]	4/8	**9.40**	10.25	7.65
	Ours	4/8	9.31	**5.75**	**2.57**
	Q-Diffusion [25]	4/6	7.88	52.65	49.21
	Ours	4/6	**9.33**	**7.34**	**4.28**
	Q-Diffusion [25]	3/8	**9.57**	40.41	39.04
	Ours	3/8	9.11	**22.36**	**20.90**

E Quantitative results for group sizes and gradient approximation

In this section, we show in Table. 7 quantitative ablation results for varying group sizes, both with and without gradient approximation, for 4/8-bit quantized DDIM [49] on CIFAR-10 [23]. We observe that both FID [16] and FID2FP32 [50] degrade at group sizes of 1 and 25, since a group size of 1 fails to capture accumulated quantization error, whereas 25 is too large to find quantization parameters minimizing the error. This result highlights the importance of selecting appropriate group sizes and aligns with the findings discussed in the main paper. We also investigate the impact of gradient approximation. When the group size is 1, we calculate the gradient at every denoising steps and therefore the approximation has no effect on performance. We show that using gradient approximation usually yields better performance, since the gradients of Eq. (14) are corrupted before the quantization parameters s_l converge, leading unstable calibration process. We also show in Fig. 7 a total amount of memory of AccuQuant with and without the gradient approximation, w.r.t. the group size. The results show that the memory requirement grows linearly with group size in the absence of the gradient approximation, whereas it remains constant across all group sizes when the approximation is applied. Consequently, omitting the noisy gradient term effectively reduce memory requirement and ensure calibration process stable.

F Gradient approximation across different dataset

In this section, we visualize the gradient approximation on LDM-4 and LDM-8 [41] across diverse datasets, including LSUN-Bedroom and LSUN-Church [56], as illustrated in Fig. 8. We plot the full gradients of Eq. (19), the gradient approximation of Eq. (20), and the residual term (*i.e.*, $c_t \frac{\partial \tilde{\epsilon}_\theta(\tilde{x}_t, t)}{\partial \tilde{x}_t}$) at 200 and 500 denoising steps. The results show that, consistent with the behavior observed in Fig. 8, the gradient approximation remains closely aligned with the full gradient throughout multiple denoising steps on both datasets, thereby corroborating the validity of our approximation. Moreover, as discussed in Table. 6 and Table. 7, by omitting the residual term we avoid noisy updates during the calibration phase, resulting in a more stable calibration process.

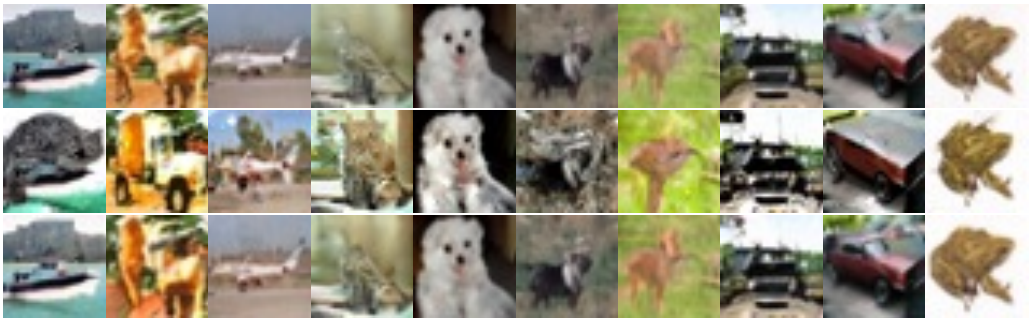


Figure 9: Visual comparisons of generated images on CIFAR-10 [23] (32×32) for unconditional image generation with 3rd order DPM-Solver++ [33] under a 4/8-bit setting. From top to bottom, the presented sequences correspond to the Full precision, Q-Diffusion [25], and Ours.

G Evaluation with advanced sampler

In this section, we report additional results for the advanced sampler (i.e., 3rd-order DPM-Solver++ [33]). We have already reported performance for Stable Diffusion [41] with the PNDM sampler [31] in Table. 3. Here, we show in Table. 8 a quantitative comparisons between our method and Q-Diffusion [25] for unconditional CIFAR-10 generation [23] over 50 denoising steps. For quantization, we devide the 50-step denoising process into 17 groups: the first 16 groups contain three consecutive steps each, and the final group contains two steps. We observe that our method consistently outperforms Q-Diffusion [25] in both FID [16] and FID2FP32 [50], demonstrating robustness across different samplers. Notably, in the 4/6-bit setting, our approach yields substantial gains, indicating enhanced tolerance to accumulated quantization error under low-bit activations. In Fig. 9, we present qualitative comparisons between our method and Q-Diffusion [25] under 4/8-bit quantization. It demonstrates that our approach recovers visual fidelity close to the full-precision model, whereas Q-Diffusion [25] produces visibly degraded outputs. This qualitative results further validates the ability of our method to mitigate accumulated quantization error.

H Additional visualization on various benchmarks

In this section, we show additional qualitative results for unconditional image generation, class-conditional image generation and text-to-image generation. For unconditional image generation, we conduct the LSUN-Bedroom [56] using LDM-4 [41] with 200 denoising steps, and LSUN-Church [56] using LDM-8 [41] with 500 denoising steps. The model weights and activations are quantized to 3/8 bits respectively. As illustrated in Fig. 10 and Fig. 11, AccuQuant successfully prevents the accumulation of quantization errors, ensuring that generated images closely match their full precision counterparts with images containing rich semantics even in low-bit configurations. For example, in Fig. 10, we observe that Q-Diffusion [25] and TFMQ-DM [19] present an overall monotonous coloration compared to the full precision results and generate different content of the bed due to the accumulated quantization error. In contrast, AccuQuant preserves a coloration and the content similar to that of the full precision model. For class conditional image generation, we conduct the ImageNet [6] using LDM-4 [41] with 20 denoising steps and quantized weights and activations to 3/8 bits respectively. As illustrated in Fig. 12, Q-Diffusion [25] and TFMQ-DM [19], which do not account for accumulating quantization error, respectively exhibit chroma noise or produce monotone coloration. PTQD [14], which does account for accumulating quantization error, generates vivid colors but alters content compared to full precision and producing unrealistic artifacts such as a chicken with two heads (*i.e., row 3, column 3 in Fig. 12*). In contrast, AccuQuant generates vibrant colors, realistic imagery, and outputs that closely match those of the full precision model. For text-to-image generation, we use COCO validation prompts [28] in Fig. 13 and user defined prompts in Fig. 14 with 50 denoising steps of Stable Diffusion v1.4 [3] quantized into 4/8 bit-widths. As illustrated in Fig. 13, PCR [50], which is designed to mitigate accumulating quantization error, allocates more bit-width to activations yet still produces images with altered style or distortion compared to the full precision model. In contrast, AccuQuant uses fewer quantizers and a lower bit-width than PCR [50] but effectively minimizes accumulated quantization error, generating realistic images that closely

Table 9: Computational cost of calibration process for DDIM [49] on CIFAR-10 (32×32) [23] with 100 timesteps.

Method	Bits (W/A)	Batch size	Calibration time (h)	Energy (Wh)	FID↓	FID2FP32↓
Full-Precision	32/32	8	–	–	4.26	0.00
Q-Diffusion [25]	6/6	8	5.97	**519.39**	30.46	35.24
Ours	6/6	8	**5.56**	561.56	**5.79**	**3.30**

Table 10: Computational cost of real-time CPU inference. Experiments are conducted on CIFAR-10 (32×32) [23] with DDIM [49] over 100 timesteps.

Method	Bits (W/A)	Batch size	GBops	Memory (MB)	CPU latency (s)	Model size (MB)	Speedup (×)
Full-Precision	32/32	64	6597	1726.0	94.589	143.08	1.00
Q-Diffusion [25]	8/8	64	798	1541.7	31.322	36.57	3.02
Ours	8/8	64	798	1541.7	31.380	36.58	3.01

Table 11: Comparison against lightweight QAT. Experiments are conducted on ImageNet (256×256) [6] with LDM-4 [2] over 20 timesteps. * denotes result under the same resource constraint.

Method	W/A	Calibration time (h)	# of calibration data	FID↓	sFID↓	IS↑	FID2FP32↓
Full-Precision	32/32	–	–	11.13	7.85	368.19	0.00
EfficientDM*	4/8	4.34	5120	12.43	25.07	197.86	14.55
EfficientDM-Full [13]	4/8	6.50	32 000	9.92	**7.40**	351.79	1.63
Ours	4/8	4.34	5120	**9.39**	7.41	**356.48**	**0.65**

match those of the full precision model. We also visualize generated images conditioned on user defined prompts. As illustrated in Fig. 14, Q-Diffusion [25] and PCR [50] generates output that differ from those of the full precision model and misalign with the text prompt (*e.g., No astronaut in row 2, column 2 and row 3, column 4 in Fig. 14a*), while AccuQuant align strongly with the prompts. In summary, AccuQuant not only achieves superior performance on quantitative evaluation metrics but also qualitatively generates images that most closely resemble full precision outputs among state-of-the-art methods, align strongly with the prompts, and exhibit naturalness and semantic richness, demonstrating the superiority of our framework.

I Computational cost and efficiency

In this section, we provide a detailed analysis of computational cost and efficiency in terms of calibration and inference. In Tab. 9, we compare AccuQuant with Q-Diffusion [25] in terms of computational cost for calibration and the generation performance. We can see that AccuQuant achieves superior performance with shorter calibration time compared to Q-Diffusion. We note that although AccuQuant may require more computational cost for one loss calculation, it only requires 50 epochs per group. In contrast, Q-diffusion calibrates each layer and residual block for 5000 epochs. Although AccuQuant incurs a marginal increase in energy consumption, it delivers a favorable trade-off for real-world deployment. In addition, we evaluate the inference efficiency of our quantized diffusion model against both Q-Diffusion and the full-precision baseline. Since the official PyTorch quantization API does not support bit-widths lower than 8, we quantize both weights and activations to 8 bits and measure the memory usage and runtime latency using ONNX Runtime with the Intel Xeon Gold 6226R CPU. As shown in Tab. 10, both quantized models achieve over a 3 times speedup compared to the FP model. Our method incurs a marginally higher latency (by less than 0.18%) and larger model size (by 0.006 MB) compared to Q-Diffusion, which we attribute to the separate quantization parameters per group. We also compare in Tab. 11 AccuQuant with a lightweight QAT approach, EfficientDM [13]. We report the results both under the same resource constraints

Table 12: Quantitative results for Gaussian deblurring on CelebA (256×256) [32].

Method	W/A	PSNR↑	LPIPS↓
Full precision	32/32	31.23	0.0584
Q-Diffusion [25]	4/8	27.63	0.2131
Ours	4/8	**30.57**	**0.0984**
Q-Diffusion [25]	3/8	24.70	0.4574
Ours	3/8	**26.15**	**0.3979**

Table 13: Quantitative results for style transfer on the COCO validation set (512×512) [28]. We report SSIM, PSNR, LPIPS, style loss, content loss, and FID2FP32. All scores are calculated using the output of the full-precision model, since ground truth images are not available for style transfer.

Method	W/A	SSIM↑	PSNR↑	LPIPS↓	Style loss↓	Content loss↓	FID2FP32↓
Q-Diffusion [25]	4/8	0.6645	18.50	0.3310	0.0012	13.2267	14.20
Ours	4/8	**0.7444**	**20.49**	**0.2594**	**0.0006**	**9.3687**	**10.81**

Table 14: Quantization results of DIT-XL/2 with 100 denoising steps on ImageNet (256×256) [23]. W/A denotes bit-widths of weights and activations. (a) without classifier-free guidance. (b) with classifier-free guidance scale of 1.5.

Method	W/A	FID↓	sFID↓
Full-precision	32/32	12.41	19.23
PTQ4DM [45]	4/8	213.66	85.11
RepQ-ViT [27]	4/8	224.14	81.24
TFMQ-DM [19]	4/8	143.47	61.09
PTQ4DiT [52]	4/8	28.90	34.56
Ours	4/8	**18.60**	**18.83**

(a) **without** classifier-free guidance

Method	W/A	FID↓	sFID↓
Full-precision	32/32	5.31	17.61
PTQ4DM [45]	4/8	215.68	86.63
RepQ-ViT [27]	4/8	226.60	77.93
TFMQ-DM [19]	4/8	141.90	56.01
PTQ4DiT [52]	4/8	7.75	22.01
Ours	4/8	**6.80**	**17.78**

(b) **with** classifier-free guidance

(denoted as EfficientDM*) and using the official training recipe [5] with larger training data and longer training time (denoted as EfficientDM-Full). We find that under an identical setting, AccuQuant achieves substantially better performance than EfficientDM*, suggesting that QAT frameworks cannot converge sufficiently within limited resource budgets. Furthermore, although EfficientDM-Full benefits from extended training time and larger datasets, AccuQuant still outperforms it in terms of FID2FP32, demonstrating the effectiveness of our method. In summary, although our method marginally increases latency and model size compared to Q-Diffusion, the benefits of reduced calibration time and improved generation quality offer a favorable trade-off for practical deployment.

J Expanding to various tasks and advanced models

In this section, we demonstrate the generalization ability of AccuQuant by evaluating image restoration (e.g., Gaussian deblurring), style transfer, and applying to transformer-based diffusion models [39]. For Gaussian deblurring, we leverage the pre-trained DDPG [10] model from the official GitHub repository [6] on the CelebA dataset (256×256 resolution, 1K images) [32], injecting noise with $\sigma_Y = 0.05$. Also, we use 100 total timesteps with a group size of 5 and calibrate with 64 samples obtained from a full-precision model. In Tab. 12, AccuQuant consistently outperforms

[5] https://github.com/ThisisBillhe/EfficientDM
[6] https://github.com/tirer-lab/DDPG

Q-Diffusion [25] across both PSNR and LPIPS metrics, demonstrating its effectiveness in the restoration setting. For style transfer (Fig. 13), we use Stable Diffusion v1.4 [7] with the prompt *'A cartoon style'* and evaluated on the COCO validation set [28], with the entire timestep of 35 using the DDIM sampler [49]. Since ground-truth images are unavailable for style transfer, we evaluate by comparing to the outputs obtained from the full-precision model. Note that we compute style and content losses following [11], using feature maps from a VGG network [47]. In Tab. 13, AccuQuant outperforms all evaluated metrics, highlighting its robustness for diverse generative tasks beyond standard image synthesis. For transformer-based diffusion model [39] adaptation, we compare AccuQuant to recent diffusion-focused quantization methods [45, 19] including diffusion-transformer based quantization [52] and advanced transformer quantization method [27]. We conduct experiments on the ImageNet dataset [6] using the DiT-XL/2 [39] with 100 timesteps, under both without and with classifier-free guidance settings [18]. As shown in Tab. 14, AccuQuant outperforms transformer-based, diffusion-focused, and even DiT-specific quantization methods, demonstrating that reducing the accumulating error across diffusion timesteps is the key factor for quantizing diffusion models. In summary, by designing the quantization methods at the diffusion framework level rather than for a specific diffusion architecture, AccuQuant generalizes to diverse tasks and diffusion models.

[7] https://huggingface.co/CompVis/stable-diffusion-v-1-4-original

Figure 10: Visual comparisons of generated images on LSUN-Bedrooms [56] (256×256) for unconditional image generation with LDM-4 [41] under a 3/8-bit setting. Each row corresponds to Full Precision, Q-Diffusion [25], TFMQ-DM [19], and Ours.

Figure 11: Visual comparisons of generated images on LSUN-Church [56] (256×256) for unconditional image generation with LDM-8 [41] under a 3/8-bit setting. Each row corresponds to Full Precision, Q-Diffusion [25], TFMQ-DM [19], and Ours.

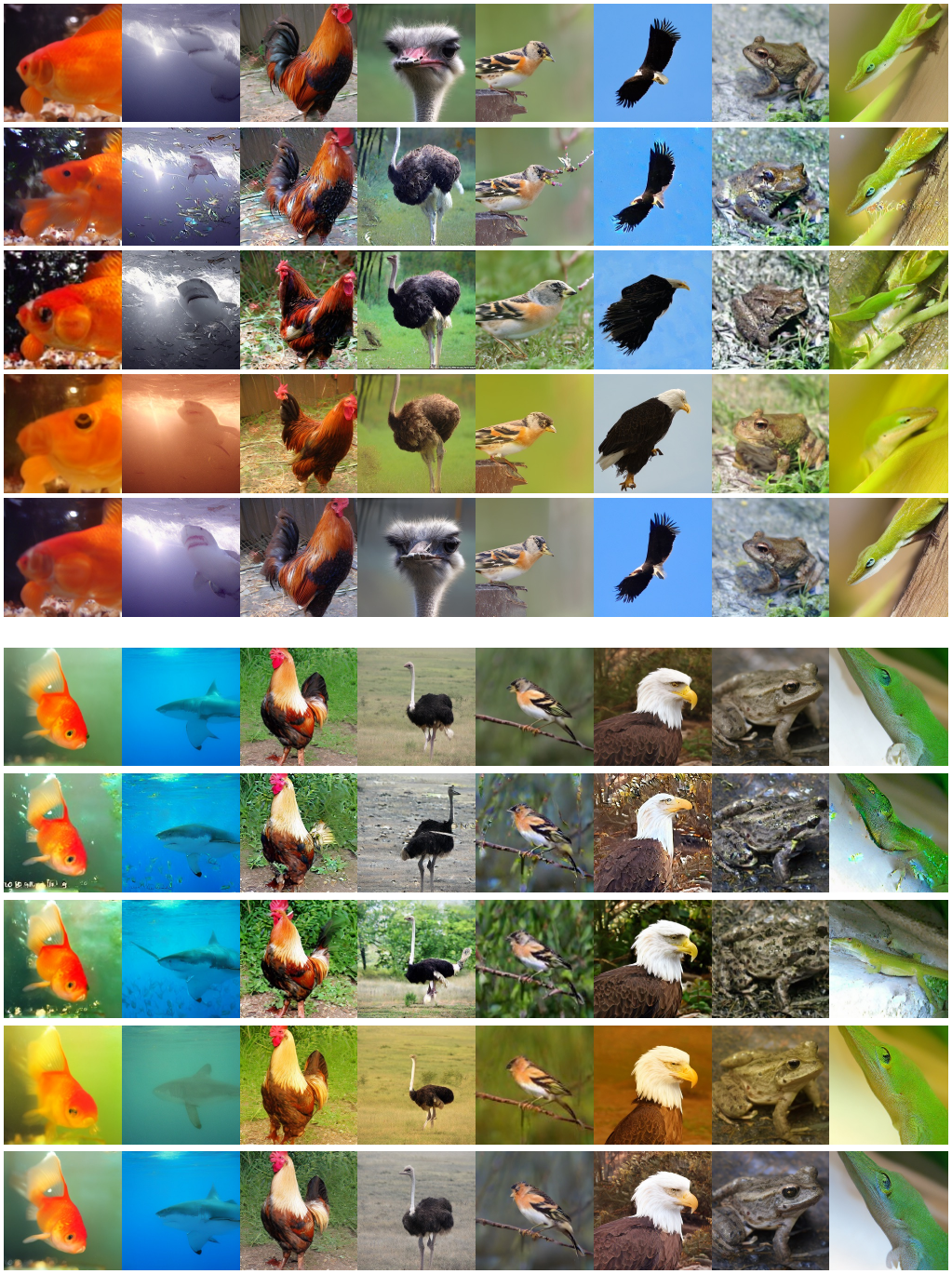
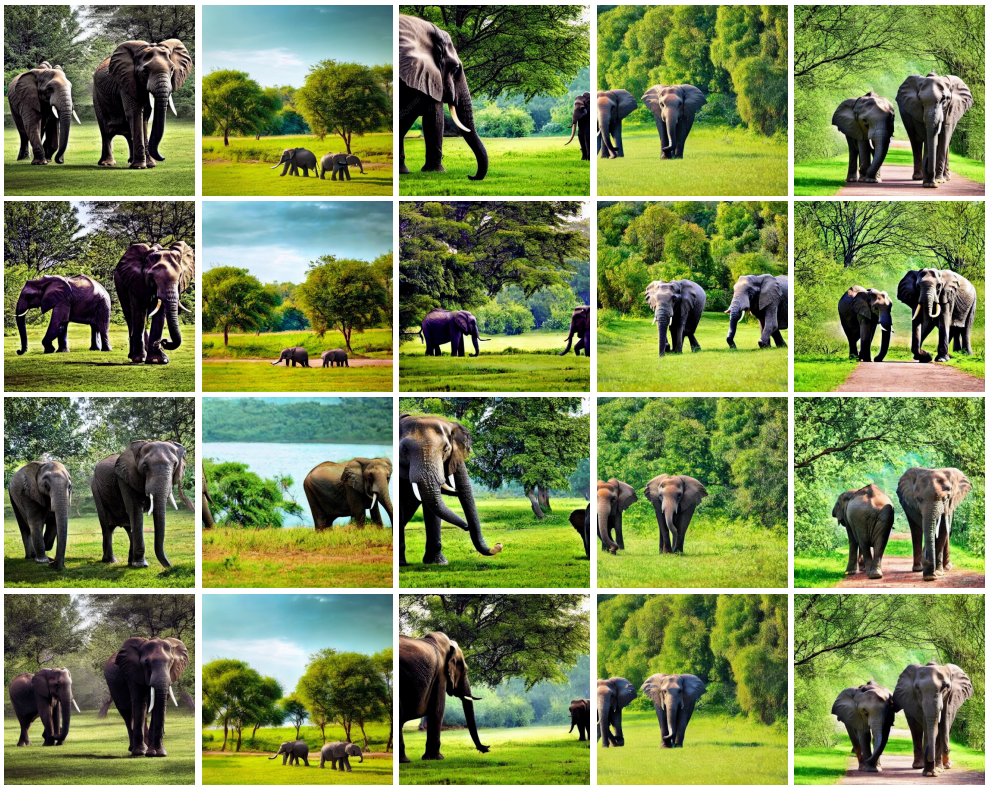


Figure 12: Visual comparisons of generated images on ImageNet [6] (256×256) for class-conditional image generation with LDM-4 [41] under a 3/8-bit setting. Each row corresponds to Full Precision, Q-Diffusion [25], PTQD [14], TFMQ-DM [19], and Ours.

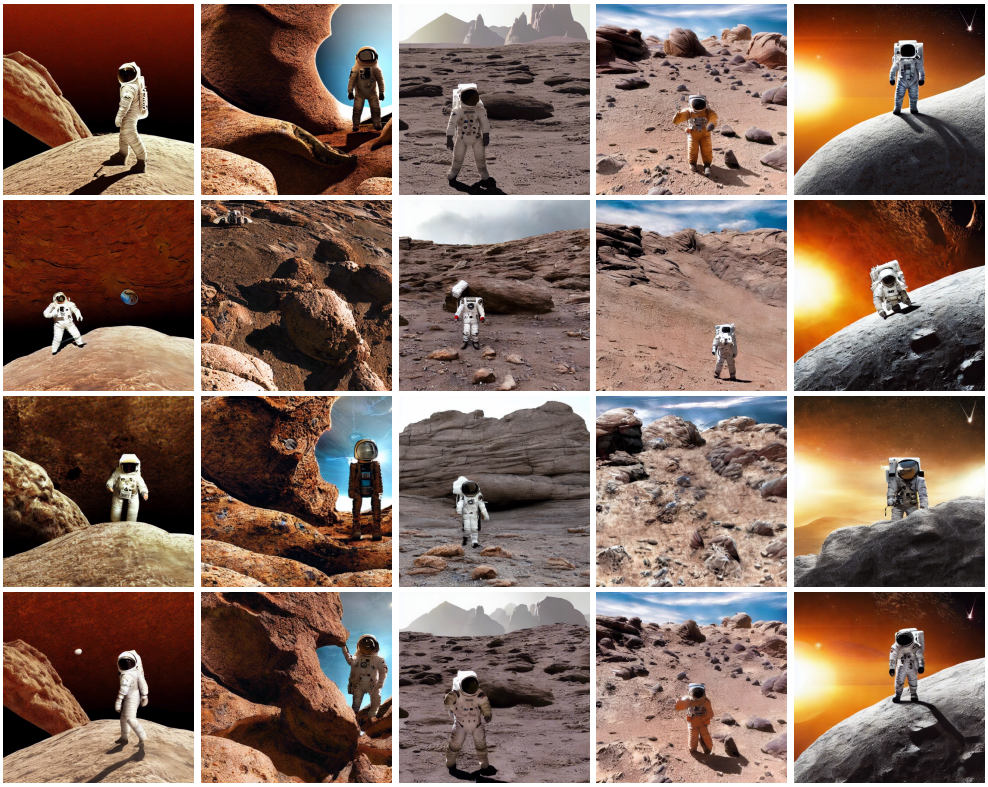
"A colorful hot air balloon floating in a blue sky above clouds."

"A table with some oranges and some cups."



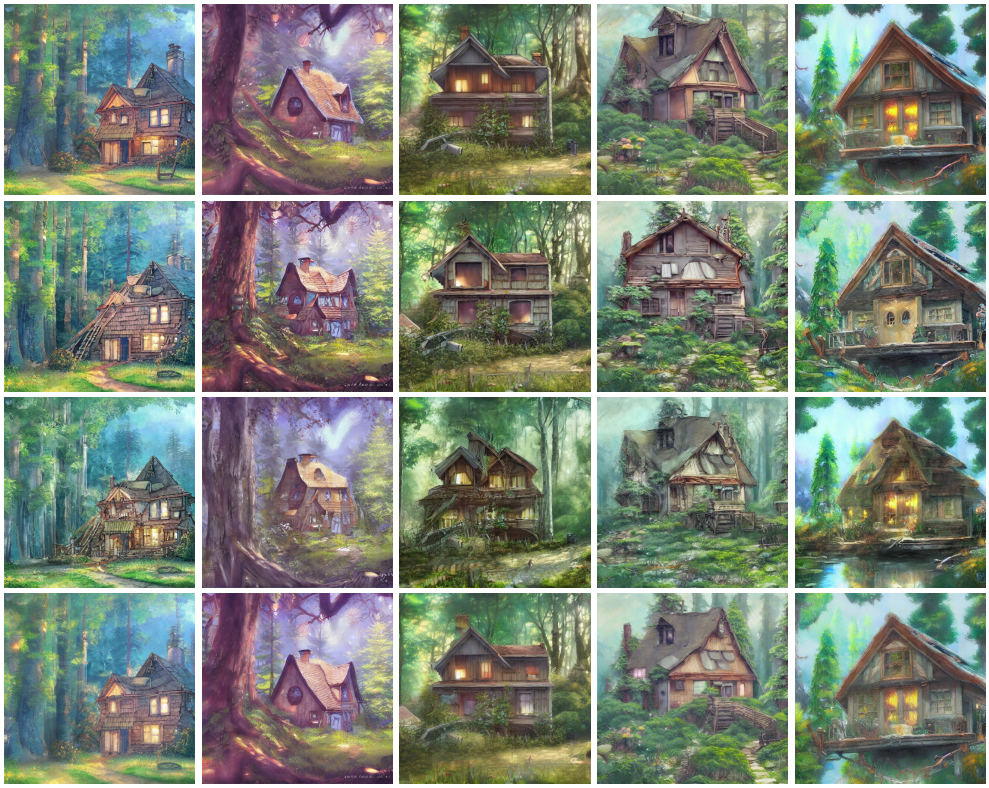
"Two elephants outdoors walking on green grass with trees nearby."

Figure 13: Visual comparisons of generated images (512×512) conditioned on COCO validation prompts [28]. We generate images by SD v1.4 [41] and its quantized versions using Q-Diffusion [25] and AccuQuant under a 4/8-bit setting and PCR [50] in 4/8.4-bit setting. Each row corresponds to Full Precision, Q-Diffusion [25], PCR($\tau = 0.2$) [50] and Ours.



“An astronaut exploring an alien planet with strange rock formations."

“A robot assistant helping in a modern kitchen."



"Cluttered house in the woods, anime, oil painting, high resolution, cottagecore, ghibli inspired"

Figure 14: Visual comparisons of generated images (512×512) conditioned on user defined prompts. We generate images by SD v1.4 [41] and its quantized versions using Q-Diffusion [25] and AccuQuant under a 4/8-bit setting and PCR [50] in 4/8.4-bit setting. Each row corresponds to Full Precision, Q-Diffusion [25], PCR($\tau = 0.2$) [50] and Ours.