HQ-DM: Single Hadamard Transformation-Based Quantization-Aware Training for Low-Bit Diffusion Models

Shizhuo Mao*, Hongtao Zou*, Qihu Xie, Song Chen, and Yi Kang†

University of Science and Technology of China, Hefei, China
{msz123,zouht49,xieqihu}@mail.ustc.edu.cn and {songch,ykang}@ustc.edu.cn

Abstract. Diffusion models have demonstrated significant applications in the field of image generation. However, their high computational and memory costs pose challenges for deployment. Model quantization has emerged as a promising solution to reduce storage overhead and accelerate inference. Nevertheless, existing quantization methods for diffusion models struggle to mitigate outliers in activation matrices during inference, leading to substantial performance degradation under low-bit quantization scenarios. To address this, we propose **HQ-DM**, a novel Quantization-Aware Training framework that applies Single Hadamard Transformation to activation matrices. This approach effectively reduces activation outliers while preserving model performance under quantization. Compared to traditional Double Hadamard Transformation, our proposed scheme offers distinct advantages by seamlessly supporting INT convolution operations while preventing the amplification of weight outliers. For conditional generation on the ImageNet 256×256 dataset using the LDM-4 model, our W4A4 and W4A3 quantization schemes improve the Inception Score by 12.8% and 467.73%, respectively, over the existing state-of-the-art method.

Keywords: Model Quantization · Single Hadamard Transformation · Diffusion Models

1 Introduction

In recent years, deep generative models have exhibited remarkable capabilities across multiple modalities, including image [15, 26, 27], speech [20, 24, 31], and text [2,34]. Among them, diffusion models [12,27] have emerged as a mainstream generative framework due to their superior generative quality and stable training dynamics. However, the generation process of diffusion models relies on an iterative, multi-step denoising sampling procedure [12], resulting in prolonged inference latency and high memory consumption that hinder their deployment in real-time or resource-constrained scenarios.

*These authors contributed equally.
†Corresponding author.

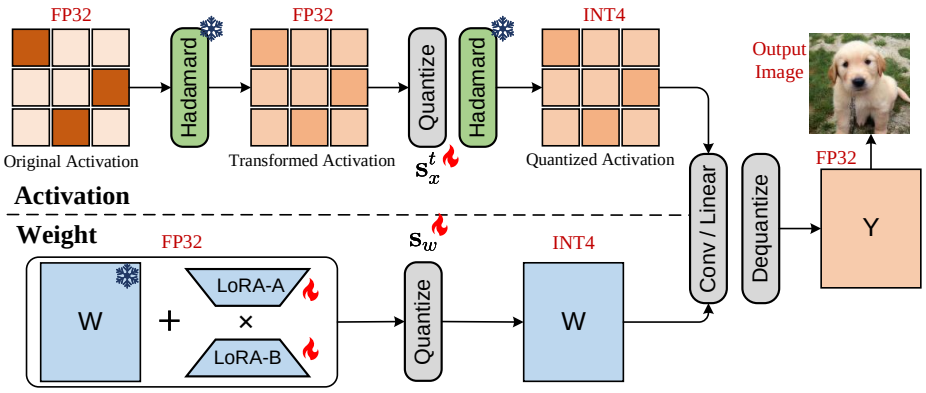


Fig. 1: Overview diagram of the HQ-DM distillation framework, illustrating the application of Single Hadamard transformation before activation quantization to reduce outliers, followed by hardware-efficient implementation of matrix multiplication and convolution operations using quantized integer matrices.

Model quantization [8, 18, 28] is an effective model compression method that maps high-precision numerical values to low-bit representations, thereby achieving significant memory savings and computational acceleration. Existing quantization approaches are typically categorized into two types: Post-Training Quantization (PTQ) and Quantization-Aware Training (QAT) [44]. PTQ [19, 30, 37] performs offline discretization of weights and activations after training, offering ease of deployment and low implementation cost, but it may suffer from notable accuracy degradation in complex generative tasks. In contrast, QAT [6, 9] introduces quantization noise during training, enabling the model to adapt to low-precision arithmetic through gradient-based optimization, thereby achieving a more favorable balance between model accuracy and quantization efficiency.

Nevertheless, applying quantization techniques directly to diffusion models presents significant and unique challenges. Unlike traditional classification or generative tasks, the activations of the noise prediction network in diffusion models exhibit strong non-stationarity during the multi-step denoising process: the scale of input noise, feature structures, and statistical properties vary substantially across timesteps, leading to continuously shifting activation distributions throughout sampling [18, 23]. Moreover, the prevalent long-tailed and outlier characteristics in diffusion models further enlarge the activation dynamic range, making them susceptible to significant clipping and rounding errors under low-bit quantization [17]. More critically, these errors accumulate over multiple denoising steps, resulting in cumulative and amplified degradation of generation quality. Consequently, the non-stationary and outlier behaviors of activations constitute the key bottlenecks that hinder accurate quantization of diffusion models, calling for systematic solutions to mitigate their impact.

Several recent studies have begun to explore the challenges of temporal quantization in diffusion models. For instance, Li et al. [18] identified the issues of tem-

poral variation in activation distributions and the accumulation of quantization errors, and proposed a timestep-aware calibration mechanism to mitigate performance degradation in PTQ. Similarly, So et al. [32] further analyzed how temporal activation dynamics influence quantization interval selection. EfficientDM [9] introduced a LoRA-based distillation scheme to effectively enhance quantized model performance, thereby bringing efficient quantization-aware training into diffusion model quantization. However, due to the lack of further methods to handle activation outliers, existing approaches still suffer from significant performance degradation under low-bit quantization.

Hadamard transformation is an orthogonal transformation capable of mapping outliers into another linear space, a property that can be leveraged to address the issue of activation outliers that hinder quantization [1, 39]. Specifically, QuaRot [1] introduces the Hadamard transformation into PTQ for LLMs, yet the adjustment of quantization factors following this transformation remains relatively under-explored. Xi et al. [39] incorporate the Hadamard transformation into quantization-aware training but do not account for the optimization of time-step-dependent quantization factors. Moreover, due to its adoption of the Double Hadamard Transformation, it lacks support for integer convolution computation modules, and the offline-generated Hadamard-transformed weight matrices amplify quantization errors.

To tackle these challenges, this paper introduces a Single Hadamard Transformation-Based Quantization-Aware Training (QAT) method for diffusion models, which mitigates outlier effects via the Hadamard transformation and integrates LoRA-based weight fine-tuning with timestep-wise learnable quantization factors. This design significantly alleviates the accumulation of quantization errors under data-free fine-tuning conditions. Experimental results show that our method substantially improves generation quality at the W4A4 / W4A3 quantization level, demonstrating its practicality and strong generalization potential for diffusion model quantization. Overall, the main contributions of this paper are summarized as follows:

(1) We introduce a Single Hadamard Transformation-based distillation framework that reduces activation outliers prior to quantization through Hadamard transformation, thereby enhancing diffusion model performance under low-bit quantization settings.

(2) We propose the Single Hadamard Transformation, which is more implementation-friendly than conventional approach. This transformation addresses two critical limitations of the conventional Hadamard transformation: its incompatibility with integer (INT) convolution operations and its tendency to generate additional outliers in weight matrices during computation.

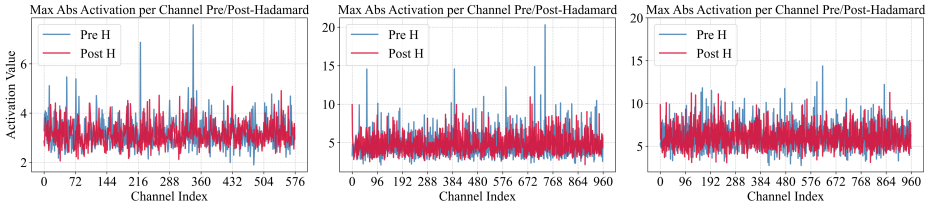
(3) HQ-DM demonstrates exceptional performance across multiple datasets and models, surpassing existing SOTA methods, particularly in low-bit activation quantization scenarios. Under W4A3 quantization with the LDM-4 model on ImageNet 256×256, HQ-DM achieves a 467.73% improvement on IS score, while under W4A4 quantization it delivers a 12.8% enhancement on IS score compared to EfficientDM.

2 Related Work

Quantization of Diffusion Models. Diffusion models represent a class of generative models that learn data distributions through a progressive denoising process, typically formulated as a Markov chain that gradually transforms random noise into structured samples. Recently, quantization has been extended to diffusion models to reduce their prohibitive computational and memory overhead. Early PTQ-based approaches such as PTQ4DM [30] and Q-Diffusion [18] leverage data-free calibration by extracting intermediate activations from the denoising process. More advanced methods, including PTQD [10], TFMQ-DM [14] and APQ-DM [35], enhance quantization precision through improved calibration and temporal feature modeling. However, PTQ-based strategies still exhibit severe performance drops at 4-bit or lower precision due to activation outliers and the non-stationary feature dynamics across timesteps. To address this issue, QAT-based methods like EfficientDM [9] and QuEST [36] incorporate fine-tuning to recover generation quality under low-bit settings. Despite the advancements achieved by these works in diffusion model quantization, these quantization schemes lead to severe performance degradation under lower-bit quantization scenarios.

Activation Outliers and Smoothing Techniques for Model Quantization. Post-training quantization (PTQ) is an effective approach to reduce model size and inference cost without retraining. However, neural activations often follow heavy-tailed distributions, where a few large-magnitude outliers dominate the tensor range and severely degrade quantization accuracy [38]. Early works mitigate this by clipping or architectural transformations such as Outlier Channel Splitting (OCS) [43], while recent methods for large models introduce algebraically equivalent scaling to smooth activation distributions without retraining. Notably, SmoothQuant [40] migrates quantization difficulty from activations to weights via scaling, proving that explicit outlier handling is the key to achieving low-bit quantization. Diffusion models exhibit more complex outlier behaviors: activation statistics vary significantly across timesteps, and quantization errors can accumulate during iterative denoising. To address this, diffusion-specific PTQ methods introduce timestep- or layer-aware smoothing mechanisms. DMQ [17] analyzes outlier sources in diffusion networks and applies diffusion-aware transformations to suppress extreme activations, while MPQ-DM [7] leverages mixed-precision allocation guided by outlier statistics to maintain quality under ultra-low-bit settings. Overall, these studies demonstrate that timestep-adaptive outlier smoothing—through equivalent scaling or precision allocation—is essential for stable and accurate quantization of diffusion models.

QuaRot [1] introduces Hadamard transformation into Post-Training Quantization (PTQ) for large language models to mitigate outliers and enhance model performance under low-bit quantization. Similarly, Xi et al. [39] incorporate the Hadamard transformation into the training framework of large models to reduce outliers and improve performance. However, these works do not optimize time-step-dependent quantization factors as training objectives. Moreover, the proposed Double Hadamard Transformation suffers from two critical limitations:



(a) Input_block_proj_out (b) Middle_block_proj_out (c) Output_block_proj_out

Fig. 2: Channel-wise analysis of outlier distribution in activations from the input_block, middle_block and output_block layer at timestep 0, comparing before and after Hadamard Transformation states. Given that the minimum abs value of each channel in the matrix is zero, the maximum value of each channel can be considered as the outlier for that channel.

incompatibility with integer-based convolution operations and the introduction of additional outliers in weight matrices, thereby hindering its effective application in diffusion models.

3 Background

Diffusion Model Diffusion Models (DMs) learn to reverse a gradual noising process, transforming a simple Gaussian prior into complex data distributions through iterative denoising [12,27]. Formally, the forward diffusion process progressively adds Gaussian noise to a clean data sample x_0 over T timesteps:

$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I) \tag{1}$$

where β_t controls the noise variance at each timestep. During training, a denoising network $\epsilon_\theta(x_t, t)$ is optimized to predict the added noise by minimizing the simplified objective:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{x_0,t,\epsilon}\left[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon, t)\|^2\right] \tag{2}$$

where $\bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$. To accelerate sampling, several variants have been proposed. DDIM [33] introduces deterministic sampling to reduce inference steps, while Latent Diffusion Models (LDMs) [27] perform diffusion in a compressed latent space, greatly improving efficiency.

Model Quantization Quantization refers to the process of mapping high-precision numbers to low-bit representations. A typical quantization method is linear mapping quantization, which maintains a stable quantization step size and uniformly maps values within the dynamic range to integer numbers. The formula for symmetric quantization from floating-point numbers to N-bit integers is as follows:

$$\mathbf{X}_{\text{int}} = \text{clamp}\left(\left\lfloor \frac{\mathbf{X}_{\text{fp}}}{\mathbf{S}} \right\rceil, min, max\right), \quad \hat{\mathbf{X}} = \mathbf{S}\cdot\mathbf{X}_{\text{int}} \tag{3}$$

where $\lfloor \cdot \rceil$ represents the rounding function, and clamp denotes truncation within the integer range representable by N-bit precision. $\mathbf{S}$ represents the quantization scale factor, which is a trainable parameter in this paper. Asymmetric quantization involves adding an offset (zero-point) after dividing $\mathbf{X}$ by the scale factor $\mathbf{S}$, which alters the dynamic range of the value distribution. To overcome the non-differentiability of the rounding and clamping functions during the QAT process, we adopt the Straight-Through Estimator (STE) convention:

$$\frac{\partial \text{Round}(x)}{\partial x} = 1, \quad \frac{\partial \text{Clamp}(x)}{\partial x} = \mathbb{I}_{[\min,\max]}(x) \tag{4}$$

Outliers hold significant implications in model quantization, referring to large values that substantially exceed the magnitude of other values. Due to the presence of these outliers, it becomes challenging to simultaneously preserve information from both normal values and outliers after quantization, consequently leading to substantial performance degradation in low-bit quantization scenarios. Compared to weight values, activations typically exhibit more frequent occurrences of outliers [4, 38, 41]. Therefore, to maintain model performance under low-bit quantization, it is crucial to mitigate the impact caused by these outlier values.

Hadamard Transformation An orthogonal matrix is defined as a real square matrix $\mathbf{Q}$ satisfying $\mathbf{Q}\mathbf{Q}^T = \mathbf{I}$. The linear transformation corresponding to an orthogonal matrix preserves both vector lengths and angles between vectors, making it particularly suitable for performing linear transformations on matrices. A Hadamard matrix represents a special class of orthogonal matrices, satisfying the following recurrence relation:

$$\mathbf{H}_0 = [1], \quad \mathbf{H}_k = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{H}_{k-1} & \mathbf{H}_{k-1} \\ \mathbf{H}_{k-1} & -\mathbf{H}_{k-1} \end{bmatrix} \tag{5}$$

The matrix $\mathbf{H}_k$ is of size $2^k \times 2^k$ and possesses both orthogonality and rotational symmetry. Consequently, it satisfies

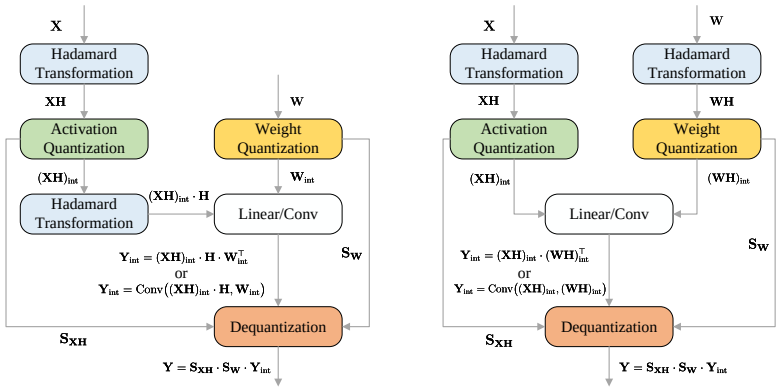
$$\mathbf{H}_k \mathbf{H}_k^\top = \mathbf{I} \quad \text{and} \quad \mathbf{H}_k = \mathbf{H}_k^\top \tag{6}$$

It can be observed that the orthonormal Hadamard matrix is defined as

$$\mathbf{H}_k = 2^{-k/2} \cdot \mathbf{H}_k^{\text{raw}} \tag{7}$$

where $\mathbf{H}_k^{\text{raw}}$ is the unnormalized Hadamard matrix with all entries in $\{+1, -1\}$. This decomposition is crucial for integer-only quantized inference, ensuring that the matrix multiplication after transformation can be performed entirely within integer arithmetic. For any matrix-vector multiplication of the form $\mathbf{H}_k\mathbf{x}$, the computational complexity is $\mathcal{O}(k \cdot 2^k)$ operations. The Hadamard transformation is a linear transformation that uses a Hadamard matrix as its transform kernel.

The order k of the base Hadamard matrix $\mathbf{H}_k$ is directly related to its outlier-diffusion capability: larger k leads to stronger diffusion of dominant outliers across all coordinates. Fundamentally, the Hadamard transformation maps input vectors into a linear subspace with fewer extreme values, thereby significantly improving model performance under low-bit quantization.



(a) Single Hadamard Transformation (b) Double Hadamard Transformation

Fig. 3: Schematic diagrams of Single Hadamard Transformation (a) and Double Hadamard Transformation (b). The key distinction lies in whether the offline Hadamard transformation is applied to the weights.

4 Method

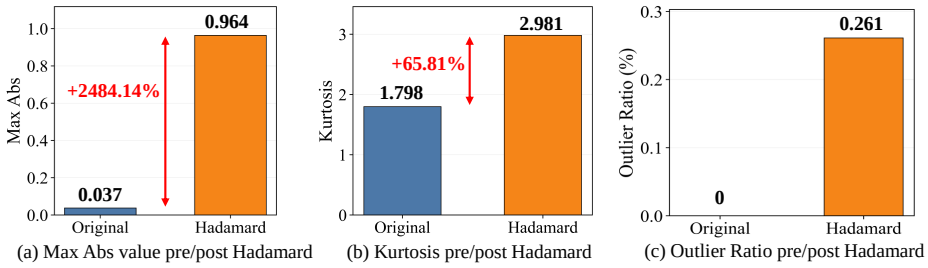
In the context of low-bit quantization for diffusion models, we employ the Hadamard transformation to suppress outliers and thereby enhance model performance under aggressive quantization. In this section, we present our Hadamard-based LoRA fine-tuning framework—termed **HQ-DM** (Single Hadamard Transformation-Based Quantization Aware Training on Diffusion Models)—which applies quantization to both activations and weights, as illustrated in Figure 1.

4.1 LoRA-based Distillation

Inspired by EfficientDM [9], we retain the LoRA-based distillation framework. The core idea of Low-Rank Adaptation (LoRA) is to freeze the original pre-trained weight and only train a pair of low-rank update matrices that are adapted to it [5,13]. This strategy drastically reduces the number of trainable parameters while preserving a sufficiently expressive parameter space, as the adaptation is concentrated on the dominant principal components of the weight updates. Formally, consider a linear transformation $\mathbf{Y} = \mathbf{X} \cdot \mathbf{W}$, where $\mathbf{X} \in \mathbb{R}^{T \times C_i}$ and $\mathbf{W} \in \mathbb{R}^{C_i \times C_o}$. Under LoRA, the adapted weight matrix becomes

$$\mathbf{W}' = \mathbf{W} + \mathbf{B} \cdot \mathbf{A} \tag{8}$$

where $\mathbf{B} \in \mathbb{R}^{C_i \times r}$ and $\mathbf{A} \in \mathbb{R}^{r \times C_o}$ are trainable low-rank matrices with $r \ll \min(C_i, C_o)$. In the distillation process, at denoising timestep t, the same noisy input $\mathbf{x}_t$ is fed into both the full-precision (FP) teacher model and the quantized student model, and the mean squared error (MSE) between their outputs is



(a) Max Abs value pre/post Hadamard (b) Kurtosis pre/post Hadamard (c) Outlier Ratio pre/post Hadamard

Fig. 4: Comparison of max absolute value, kurtosis, and outlier ratio (values exceeding 3σ) before and after Hadamard transformation on output_blocks0.attn2.k.weight in LDM-4. "Original" corresponds to the Single Hadamard Transformation (no weight transformation), while "Hadamard" denotes the Double Hadamard Transformation.

minimized:

$$\mathcal{L}_{\text{distill}} = \mathbb{E}_{\mathbf{x}_t,t}\left[\|\epsilon_{\text{FP}}(\mathbf{x}_t,t) - \epsilon_{\text{quant}}(\mathbf{x}_t,t)\|_2^2\right] \tag{9}$$

Prior work has shown that in diffusion models, the activation distributions vary significantly across timesteps. To address this, LSQ [6, 9] introduces timestep-aware quantization scales and optimizes them independently for each denoising step. We adopt this strategy unchanged for both weight and activation quantization in our framework.

4.2 Hadamard Transformation to smooth outliers

We utilize Single Hadamard Transformation to reduce outliers in the activations, while avoiding the introduction of additional outliers in the weights that typically arises from the conventional Double Hadamard Transformation. As shown in Figure 2, the outliers in the activation values are significantly reduced after the transformation.

Matrix Multiplication The Hadamard transformation is applied to a 2D input matrix by multiplying the matrix to be quantized with a Hadamard matrix. Since the input channel dimension C_i is not necessarily a power of two, we construct a block-diagonal Hadamard matrix $\mathbf{H} \in \mathbb{R}^{C_i \times C_i}$ composed of m identical submatrices $\mathbf{H}_k$, i.e.,

$$\mathbf{H} = \text{BlockDiag}(\underbrace{\mathbf{H}_k, \mathbf{H}_k, \ldots, \mathbf{H}_k}_{m \text{ blocks}}), \quad \text{where} \quad C_i = m \cdot 2^k. \tag{10}$$

This block-diagonal structure preserves orthogonality, satisfying $\mathbf{H}\mathbf{H}^\top = \mathbf{H}^2 = \mathbf{I}$. In cases where the number of input channels is smaller than the base order k (e.g., during the initial projection of image features), we set $\mathbf{H} = \mathbf{I}$, effectively bypassing the transformation. Given the orthogonality of the Hadamard matrix ($\mathbf{H}^\top = \mathbf{H}^{-1} = \mathbf{H}$), the original activation matrix $\mathbf{X}$ can be exactly recovered

from its transformed and quantized form via inverse transformation followed by dequantization. Specifically, let $(\mathbf{XH})_{\text{int}}$ denote the integer-valued quantized representation of the transformed activations $\mathbf{XH}$, and let $\mathbf{S_{XH}}$ be the corresponding floating-point quantization scale. Then the dequantized output is:

$$\hat{\mathbf{X}} = \mathbf{S_{XH}} \cdot (\mathbf{XH})_{\text{int}} \cdot \mathbf{H} \tag{11}$$

Consequently, for a linear layer $\mathbf{Y} = \mathbf{XW}$, we insert the identity $\mathbf{HH}^\top = \mathbf{I}$ to obtain:

$$\mathbf{Y} = \mathbf{XH} \cdot \mathbf{H}^\top \mathbf{W} = (\mathbf{XH}) \cdot \mathbf{H} \cdot \mathbf{W} \approx \mathbf{S_{XH}}\, \mathbf{S_W} \cdot \big((\mathbf{XH})_{\text{int}} \cdot \mathbf{H} \cdot \mathbf{W}_{\text{int}}\big) \tag{12}$$

In practice, the normalization factor $2^{-k/2}$ (arising from the orthonormal definition $\mathbf{H}_k = 2^{-k/2}\mathbf{H}_k^{\text{raw}}$) is absorbed into the floating-point quantization scales. As a result, the core computation $(\mathbf{XH})_{\text{int}} \cdot \mathbf{H} \cdot \mathbf{W}_{\text{int}}$ involves only integer matrix multiplication, enabling efficient inference on hardware accelerators.

Convolution For a convolutional layer, let the input activation tensor be $\mathbf{X}_c \in \mathbb{R}^{B \times C_{\text{in}} \times h \times w}$ and the convolutional kernel be $\mathbf{W}_c \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times L \times L}$, where C_{in} and C_{out} denote the input and output channel dimensions, and $h \times w$ and $L \times L$ are the spatial resolutions of the activations and the kernel. To apply the Hadamard transformation on convolution, we first reshape $\mathbf{X}_c$ into a 2D matrix $\widetilde{\mathbf{X}}_c \in \mathbb{R}^{(B \cdot C_{\text{in}} \cdot h) \times w}$ by merging the batch, input-channel, and height dimensions. We then compute the transformed activations as the matrix product $(\widetilde{\mathbf{X}}_c \mathbf{H}_c)$, where $\mathbf{H}_c \in \mathbb{R}^{w \times w}$ is a block-diagonal Hadamard matrix. This design ensures that the Hadamard transformation operates only along the last (width) dimension and that $\mathbf{H}_c$ is square—allowing compatibility with standardized Hadamard orders. Due to the orthogonality of $\mathbf{H}_c$ ($\mathbf{H}_c^\top = \mathbf{H}_c^{-1} = \mathbf{H}_c$), the original activations can be recovered from the quantized transformed representation via inverse transformation and dequantization. Specifically, let $(\widetilde{\mathbf{X}}_c \mathbf{H}_c)_{\text{int}}$ denote the integer-valued quantized form of the product $\widetilde{\mathbf{X}}_c \mathbf{H}_c$, and let $\mathbf{S}_{\widetilde{\mathbf{X}}_c \mathbf{H}_c}$ be its associated floating-point quantization scale. The dequantized result is:

$$\hat{\widetilde{\mathbf{X}}}_c = \mathbf{S}_{\widetilde{\mathbf{X}}_c \mathbf{H}_c} \cdot (\widetilde{\mathbf{X}}_c \mathbf{H}_c)_{\text{int}} \ \cdot \mathbf{H}_c \tag{13}$$

Consequently, the convolution operation can be rewritten using the identity $\mathbf{H}_c \mathbf{H}_c^\top = \mathbf{I}$ as:

$$\mathbf{Y} = \text{Conv}\big((\widetilde{\mathbf{X}}_c \mathbf{H}_c) \cdot \mathbf{H}_c,\, \mathbf{W}_c\big) \approx \mathbf{S}_{\widetilde{\mathbf{X}}_c \mathbf{H}_c}\, \mathbf{S}_{\mathbf{W}_c} \cdot \text{Conv}\Big((\widetilde{\mathbf{X}}_c \mathbf{H}_c)_{\text{int}} \cdot \mathbf{H}_c,\, (\mathbf{W}_c)_{\text{int}}\Big) \tag{14}$$

Similarly, in practical implementation, the normalization factor of the Hadamard matrix $2^{-k/2}$ can be merged with the preceding floating-point quantization factors, so that the convolution operation is performed only on integer matrices.

Single Hadamard Transformation For any standard basis vector $\mathbf{e}_i \in \mathbb{R}^{2^k}$ (i.e., a vector with a single entry equal to 1 and all others 0), the product $\mathbf{e}_i^\top \mathbf{H}_k$ yields a constant vector:

$$\mathbf{e}_i^\top \mathbf{H}_k = 2^{-k/2} \cdot \mathbf{1}_{2^k}, \quad \text{where} \quad \mathbf{1}_{2^k} = (1, 1, \ldots, 1)^\top \in \mathbb{R}^{2^k}. \tag{15}$$

This property implies that when the input vector contains a dominant outlier—i.e., one coordinate significantly larger than the rest—the Hadamard transformation effectively maps it to a uniform (flat) vector. Such a structure is highly favorable for quantization, as the resulting equal-valued entries simplify the quantization process and improve its efficiency. However, similarly, applying a Hadamard transformation to a vector with uniformly distributed elements will increase the outliers in its distribution. For example, a vector with all elements equal to 1 (i.e. $\mathbf{1}_{2^k}$) yields $(2^{k/2}, 0, 0, \ldots, 0)$ after transformation, which significantly amplifies the outliers in the resulting vector.

$$\mathbf{1}_{2^k}\mathbf{H}_k = 2^{k/2}\,\mathbf{e}_1^\top \tag{16}$$

While previous research has employed Hadamard transformation to enhance model performance, the Hadamard transformation utilized in HQ-DM differs significantly from conventional approaches: we designate our method as the Single Hadamard Transformation, in contrast to the conventional Double Hadamard Transformation (see Figure 3 (a) (b)). The primary distinction lies in the computational methodology. Conventional Double Hadamard Transformation attempts to insert Hadamard transformation between the operations of $\mathbf{X}$ and $\mathbf{W}$, enabling online quantization of $\mathbf{XH}$ while performing $\mathbf{H}^\top\mathbf{W}$ quantization offline. This approach introduces two critical limitations: 1) inadequate support for convolution operations in diffusion models (due to the non-applicability of matrix multiplication distributive property to convolutions, especially for multi strides), and 2) increased outliers in the weight matrix $\mathbf{W}$ when computing $\mathbf{H}^\top\mathbf{W}$ with originally low-outlier $\mathbf{W}$ matrices, consequently amplifying quantization errors and degrading model performance (see Figure 4). Our proposed Single Hadamard Transformation not only minimizes outlier introduction in weight matrices but also provides excellent convolution support. More importantly, by extracting the floating-point factor $2^{-k/2}$, the $\mathbf{H}_k^{\text{raw}}$ matrix becomes a sparse integer matrix, enabling efficient hardware implementation of intermediate results through integer matrix multiplication.

Implementation-Friendly Scheme Single Hadamard Transformation possesses the following characteristics: its transformation of activation matrices renders activations more amenable to quantization, thereby achieving high efficiency for low-bit integer arithmetic. In contrast to several existing approaches that employ asymmetric quantization for activations (like EfficientDM), HQ-DM adopts symmetric quantization for activations, ensuring compatibility with hardware acceleration for integer matrix multiplication. While this approach effectively reduces the available quantization bits by one compared to asymmetric quantization in extreme cases, experimental results still demonstrate the substantial advantages of HQ-DM. Furthermore, as previously discussed, the Single Hadamard Transformation enables direct execution of integer matrix multiplication and convolution operations without requiring subsequent dequantization, establishing it as a hardware-friendly scheme.

Method	Bit (W/A)	IS ↑	FID ↓	sFID ↓	Precision(%)↑
FP	32/32	365.35	11.22	7.78	93.68
Q-Diffusion	8/8	350.93	10.60	9.29	92.46
PTQD	8/8	359.78	**10.05**	9.01	93.00
EfficientDM	8/8	362.34	11.38	8.04	**93.77**
HQ-DM	8/8	**363.22**	11.01	**7.76**	**93.46**
Q-Diffusion	4/8	336.80	9.29	9.29	91.06
PTQD	4/8	344.72	**8.74**	7.98	91.69
EfficientDM	4/8	351.97	9.96	7.80	92.48
HQ-DM	4/8	**355.59**	10.07	**7.14**	**93.07**
PTQD	8/4	1.93	262.01	111.10	0.54
EfficientDM	8/4	252.00	**6.75**	**8.48**	84.90
HQ-DM	8/4	**292.76**	7.68	9.93	**88.68**
PTQD	4/4	2.57	258.11	124.33	0.69
QuEST	4/4	202.45	**5.98**	–	–
EfficientDM	4/4	220.20	6.63	9.80	80.97
HQ-DM	4/4	**248.30**	6.58	**9.60**	**84.44**
PTQD	4/3	1.62	297.88	150.46	0.31
EfficientDM	4/3	8.74	99.78	65.79	27.48
HQ-DM	4/3	**49.62**	**33.47**	**12.09**	**54.87**

Table 1: Performance comparisons of fully-quantized LDM-4 models on ImageNet 256×256 dataset. The sampling process employs 20 denoising steps. We evaluate IS (higher is better), FID (lower is better), sFID (lower is better), Precision (higher is better) on these quantized models.

5 Experiments

5.1 Experiments Settings

We conduct experiments on conditional generation using ImageNet 256×256 [3] and unconditional generation on both LSUN-Bedrooms 256×256 and LSUN-Churches 256×256 [42]. For our backbone architectures, we employ both U-Net-based Latent Diffusion Models (LDM) [27] and Transformer-based Diffusion Models (DiT) [25]. To adhere to the standard configurations of these models, the denoising process is executed using the DDIM sampler [33] for LDM and the default Improved DDPM sampler [23] for DiT. Trainable parameters are optimized using the AdamW optimizer [16, 21], with distinct initial learning rates assigned to activation quantization factors, weight quantization factors, and LoRA low-rank matrices. For evaluation, we utilize generated sample batches of 50,000 images from each model and compute metrics against reference batches. The primary evaluation metrics include Inception Score (IS) [29], Fréchet Inception Distance (FID) [11], sFID [22], Precision and Recall. We compare our results with several prominent works in the field, including Q-Diffusion [18], PTQD [10], QuEST [36], TFMQ-DM [14], and EfficientDM [9]. Our experiments cover multiple quantization bit configurations, where WxAy denotes x-bit weight quan-

tization and y-bit activation quantization respectively. More comparisons and details can be found in Appendix.

5.2 Experiment Results

Conditional generation on LDM We evaluate the performance of HQ-DM on ImageNet 256 × 256 using LDM-4. The results in Table 1 demonstrate that HQ-DM outperforms existing methods across these bit configurations, with this advantage becoming increasingly pronounced at lower activation quantization bit-widths. For the W4A4 configuration, HQ-DM surpasses the current SOTA method EfficientDM by 12.76% in Inception Score. Compared to EfficientDM, HQ-DM achieves performance improvements exceeding 100% across all four metrics under the W4A3 configuration, with approximately 99.7% enhancement in precision, thereby highlighting HQ-DM's potential for ultra-low-bit quantization.

Method	Bit (W/A)	FID ↓	sFID ↓	Precision(%) ↑	Recall(%) ↑
FP	32/32	7.56	14.06	51.95	41.18
TFMQ-DM	8/4	226.08	105.91	0.04	0.00
EfficientDM	8/4	17.91	19.44	40.40	**37.86**
HQ-DM	8/4	**12.35**	**16.27**	**47.31**	37.44
TFMQ-DM	4/4	234.29	108.42	0.04	0.00
EfficientDM	4/4	16.81	20.81	41.42	29.66
HQ-DM	4/4	**13.15**	**17.53**	**45.32**	**36.34**
TFMQ-DM	3/4	227.64	107.44	0.06	7.32
QuEST	3/4	**19.08**	32.75	–	–
EfficientDM	3/4	20.92	22.78	**38.57**	29.38
HQ-DM	3/4	**19.22**	**19.29**	**38.32**	**31.92**
TFMQ-DM	2/4	245.85	130.20	0.06	0.00
QuEST	2/4	**24.92**	36.33	–	–
EfficientDM	2/4	33.68	**27.19**	28.76	**20.34**
HQ-DM	2/4	28.50	30.17	**32.64**	18.76
EfficientDM	4/3	26.80	24.01	33.43	29.02
HQ-DM	4/3	**21.88**	**22.15**	**34.52**	**29.10**

Table 2: Performance comparisons of fully-quantized LDM-4 models on LSUN-Bedrooms 256 × 256 dataset. The sampling process employs 100 denoising steps.

Unconditional generation on LDM Following conventional practice, we employ the LDM-4 model for unconditional image generation on LSUN-Bedrooms 256 × 256 in Table 2, and the LDM-8 model for unconditional generation on LSUN-Churches 256 × 256 in Table 3. On LSUN-Bedrooms, HQ-DM surpasses existing quantization schemes across multiple bit configurations, achieving FID reductions of 5.56, 3.66 and 4.92 for W8A4, W4A4 and W4A3 settings, respectively. It is noteworthy that since HQ-DM exclusively processes activations, the accuracy of the quantized model depends on weight stability. In other words,

Method	Bit (W/A)	FID ↓	sFID ↓	Precision(%) ↑	Recall(%) ↑
FP	32/32	6.17	20.98	59.96	48.54
EfficientDM	4/4	12.83	30.04	52.61	33.69
HQ-DM	4/4	**9.64**	**28.46**	**57.10**	**35.91**
EfficientDM	4/3	20.97	36.56	46.25	26.06
HQ-DM	4/3	**14.67**	**31.94**	**53.22**	**27.08**

Table 3: Performance comparisons of fully-quantized LDM-8 models on LSUN-Churches 256×256 dataset. The sampling process employs 100 denoising steps.

the closer the quantized weights remain to the original floating-point model, the lower the activation quantization loss achieved by HQ-DM. On LSUN-Churches, HQ-DM also outperforms EfficientDM at low quantization bit-widths. Compared to EfficientDM, HQ-DM reduces the FID by 3.19 under the W4A4 configuration and by 6.3 under W4A3. It is noteworthy that these improvements were achieved while employing hardware-friendly symmetric quantization, which typically incurs an adverse impact on performance. Across both LSUN datasets, we consistently observe that the performance gains from HQ-DM become increasingly pronounced as activation quantization bit-width decreases.

Conditional generation on DiT We implement the conditional generation of HQ-DM on the DiT-XL model and compare it against the EfficientDM method in Table 4. Under the W4A4 quantization configuration, HQ-DM consistently outperforms EfficientDM across all metrics. For instance, on DiT-XL (ImageNet 256×256), HQ-DM achieves a relative improvement of 60.92% in IS and 13.79% in precision. Meanwhile, in terms of FID and sFID, HQ-DM reduces the scores by 0.79 and 3.72, respectively, compared to EfficientDM. These results demonstrate the robust scalability and effectiveness of our method on mainstream Transformer-based architectures beyond U-Net.

Model (Dataset)	Method	Bit (W/A)	IS ↑	FID ↓	sFID ↓	Precision(%) ↑
DiT-XL ImageNet 256	FP	32/32	477.51	16.86	11.08	93.25
	EfficientDM	4/4	221.07	10.41	13.90	77.45
	HQ-DM	4/4	**355.74**	**9.62**	**10.18**	**88.13**
DiT-XL ImageNet 512	FP	32/32	451.68	17.71	8.40	83.60
	EfficientDM	4/4	193.00	9.91	14.72	82.03
	HQ-DM	4/4	**278.90**	**9.04**	**11.84**	**84.34**

Table 4: Performance comparisons using **DiT-XL** models on ImageNet dataset using 50 denoising steps.

Distillation Training Performance Figure 5 (a) (b) illustrates the convergence characteristics of HQ-DM under W4A4 and W4A3 distillation training settings. The results indicate that HQ-DM achieves more stable convergence

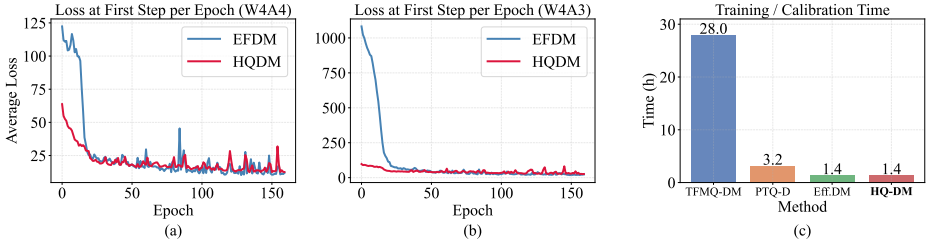


Fig. 5: (a-b) Training loss curves for W4A4 and W4A3 configurations on the ImageNet 256×256 dataset. We observe the loss change at step 0, as the first step of the training process corresponds to the final step of the inference denoising output. (c) Comparison of time costs for training or calibration across different quantization methods.

compared to EfficientDM, with accelerated convergence rates. As the activation quantization bit-width decreases, the acceleration and stability of training become increasingly pronounced. Figure 5 (c) compares the calibration and training time costs of various quantization schemes on a single A100 GPU. The results demonstrate that HQ-DM achieves a training efficiency comparable to EfficientDM, with both requiring significantly lower time costs. This further indicates that the computational overhead introduced by the sparse Hadamard transformation (i.e. $\mathbf{H}_k^{\text{raw}}$) is negligible. Consequently, this underscores that HQ-DM is a highly efficient quantization solution.

5.3 Ablation Study

Method/Variant	LSQ	LoRA	Hada.	IS↑	FID↓	sFID↓	Prec.(%)↑
FP32 Baseline				365.35	11.22	7.78	93.68
EfficientDM	✓	✓		220.20	6.63	9.80	80.97
Naive PTQ + LSQ	✓			5.09	193.59	67.24	3.38
Naive PTQ + LSQ + Hada.	✓		✓	**33.29**	**68.16**	**48.86**	**26.50**
Naive QAT + LSQ	✓			159.24	7.69	8.38	76.85
Naive QAT + LSQ + Hada.	✓		✓	**203.69**	**6.47**	10.98	**82.53**
LoRA only		✓		185.51	8.55	10.46	73.25
LoRA + Hada.		✓	✓	**229.75**	**6.52**	10.72	**79.55**
HQ-DM (Full)	✓	✓	✓	248.30	6.58	9.60	84.44

Table 5: W4A4 Ablation study of fully-quantized LDM-4 models on ImageNet 256×256. We evaluate the impact of LSQ (timestep-aware learning quantization scales), LoRA (LoRA-based distillation), and our Hadamard Transformation (Hada.).

Component Study Table 5 presents a component-wise ablation study of HQ-DM on the LDM-4 model (ImageNet 256×256) under W4A4 quantization. The evaluation demonstrates that the Hadamard transformation operates independently of distillation adjustments. For example, it significantly enhances the

Method	Bit (W/A)	IS ↑	FID ↓	sFID ↓	Precision(%) ↑	Latency (s/sample) ↓	Speedup ↑
FP	32/32	365.35	11.22	7.78	93.68	-	-
DHQ-DM	8/8	5.72	178.07	36.17	24.92	1.0688	1.00×
HQ-DM	8/8	**363.22**	**11.01**	**7.76**	**93.46**	**0.9419**	**1.13**×
DHQ-DM	4/8	6.13	199.00	35.48	14.41	1.1510	1.00×
HQ-DM	4/8	**355.59**	**10.07**	**7.14**	**93.07**	**1.0257**	**1.12**×
DHQ-DM	8/4	4.74	167.97	42.42	18.75	1.0677	1.00×
HQ-DM	8/4	**292.76**	**7.68**	**9.93**	**88.68**	**0.9391**	**1.14**×
DHQ-DM	4/4	4.37	179.40	49.20	14.63	1.1585	1.00×
HQ-DM	4/4	**248.30**	**6.58**	**9.60**	**84.44**	**1.0581**	**1.10**×

Table 6: Performance and efficiency comparisons between DHQ-DM (Double Hadamard Transformation) and HQ-DM on ImageNet 256×256. Latency is measured per sample. Speedup is calculated relative to DHQ-DM within each bit-width.

Naive PTQ scheme (where LSQ here denotes the application of timestep-aware static scales) by elevating the IS from 5.09 to 33.29 and lowering the FID by 125.43. Furthermore, the experiments without learnable LSQ confirm that HQ-DM's gains are not reliant on LSQ, as it maintains robust performance (e.g., 6.52 FID in the LoRA+Hada. variant) even with a fixed scale.

Transformation Scheme Table 6 provides a detailed comparison between the traditional Double Hadamard Transformation and the Single Hadamard Transformation using the LDM-4 model on ImageNet 256×256 dataset. Since Double Hadamard Transformation is inherently incompatible with convolutions, these layers are regressed to the Single Hadamard Transformation for a fair comparison. It is observed that the Double Hadamard Transformation consistently lags in performance across all quantization settings, with an inference latency overhead exceeding 10%. This performance degradation aligns with our earlier assertion: imposing Hadamard transformation on weights containing minimal outliers can exacerbate quantization noise, which then propagates through the iterative sampling process. Additional ablation results are provided in the Appendix.

6 Conclusion

We propose HQ-DM, a novel framework that enables low-bit quantization for diffusion models. To address the outlier issue in activation values during diffusion model computation, HQ-DM employs a Single Hadamard Transformation to reduce activation outliers, thereby enhancing quantized model performance. Furthermore, this hardware-friendly framework is capable of supporting integer convolution operations without amplifying outliers in the weights during computation, thereby enhancing performance.

References

1. Ashkboos, S., Mohtashami, A., Croci, M.L., Li, B., Cameron, P., Jaggi, M., Alistarh, D., Hoefler, T., Hensman, J.: Quarot: Outlier-free 4-bit inference in rotated llms. Advances in Neural Information Processing Systems **37**, 100213–100240 (2024)
2. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255 (2009)
4. Dettmers, T., Lewis, M., Belkada, Y., Zettlemoyer, L.: Gpt3.int8 (): 8-bit matrix multiplication for transformers at scale. Advances in neural information processing systems (2022)
5. Dettmers, T., Pagnoni, A., Holtzman, A., Zettlemoyer, L.: Qlora: Efficient finetuning of quantized llms. Advances in neural information processing systems **36**, 10088–10115 (2023)
6. Esser, S.K., McKinstry, J.L., Bablani, D., Appuswamy, R., Modha, D.S.: Learned step size quantization. In: International Conference on Learning Representations (ICLR) (2020)
7. Feng, W., Qin, H., Yang, C., An, Z., Huang, L., Diao, B., Wang, F., Tao, R., Xu, Y., Magno, M.: Mpq-dm: Mixed precision quantization for extremely low bit diffusion models. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 16595–16603 (2025)
8. Gao, W., Hou, Z., Yin, J., Wang, F., Peng, L., Liu, X.: Modulated diffusion: Accelerating generative modeling with modulated quantization. In: Proceedings of the 42nd International Conference on Machine Learning (ICML) (2025)
9. He, Y., Liu, J., Wu, W., Zhou, H., Zhuang, B.: Efficientdm: Efficient quantization-aware fine-tuning of low-bit diffusion models. In: International Conference on Learning Representations (ICLR) (2024)
10. He, Y., Liu, L., Liu, J., Wu, W., Zhou, H., Zhuang, B.: PTQD: accurate post-training quantization for diffusion models. In: Advances in Neural Information Processing Systems (NeurIPS) (2023)
11. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30**, 6626–6637 (2017)
12. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Advances in Neural Information Processing Systems (NeurIPS). vol. 33, pp. 6840–6851 (2020)
13. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-rank adaptation of large language models. In: International Conference on Learning Representations (2022)
14. Huang, Y., Gong, R., Liu, J., Chen, T., Liu, X.: TFMQ-DM: temporal feature maintenance quantization for diffusion models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7362–7371 (2024)

15. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4401–4410 (2019)
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (2015)
17. Lee, D., Hur, J., Shon, H., Lee, J.Y., Kim, J.: Dmq: Dissecting outliers of diffusion models for post-training quantization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18510–18520 (2025)
18. Li, X., Liu, Y., Lian, L., Yang, H., Dong, Z., Kang, D., Zhang, S., Keutzer, K.: Q-diffusion: Quantizing diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2023)
19. Li, Y., Gong, R., Tan, X., Yang, Y., Hu, P., Zhang, Q., Yu, F., Wang, W., Gu, S.: BRECQ: pushing the limit of post-training quantization by block reconstruction. In: International Conference on Learning Representations (2021)
20. Liu, H., Yuan, Y., Liu, X., Mei, X., Kong, Q., Tian, Q., Wang, Y., Wang, W., Wang, Y., Plumbley, M.D.: Audioldm 2: Learning holistic audio generation with self-supervised pretraining. IEEE/ACM Transactions on Audio, Speech, and Language Processing **32**, 2871–2883 (2024)
21. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2019)
22. Nash, C., Menick, J., Dieleman, S., Battaglia, P.: Generating images with sparse representations. In: International Conference on Machine Learning. pp. 7958–7968. PMLR (2021)
23. Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: International conference on machine learning. pp. 8162–8171. PMLR (2021)
24. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: Wavenet: A generative model for raw audio. In: Speech Synthesis Workshop (SSW) (2016)
25. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF international conference on computer vision (2023)
26. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: SDXL: improving latent diffusion models for high-resolution image synthesis. In: International Conference on Learning Representations (2024)
27. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
28. Ryu, H., Park, N., Shim, H.: DGQ: distribution-aware group quantization for text-to-image diffusion models. In: International Conference on Learning Representations (ICLR) (2025)
29. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. Advances in neural information processing systems **29**, 2226–2234 (2016)
30. Shang, Y., Yuan, Z., Xie, B., Wu, B., Yan, Y.: Post-training quantization on diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1972–1981 (2023)
31. Shen, K., Ju, Z., Tan, X., Liu, E., Leng, Y., He, L., Qin, T., Zhao, S., Bian, J.: Naturalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers. In: International conference on learning representations (2024)
32. So, J., Lee, J., Ahn, D., Kim, H., Park, E.: Temporal dynamic quantization for diffusion models. In: Advances in Neural Information Processing Systems (NeurIPS) (2023)

33. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: International Conference on Learning Representations (ICLR) (2021)
34. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
35. Wang, C., Wang, Z., Xu, X., Tang, Y., Zhou, J., Lu, J.: Towards accurate post-training quantization for diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16026–16035 (2024)
36. Wang, H., Shang, Y., Yuan, Z., Wu, J., Yan, J., Yan, Y.: Quest: Low-bit diffusion model quantization via efficient selective finetuning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15542–15551 (2025)
37. Wei, X., Gong, R., Li, Y., Liu, X., Yu, F.: Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. In: International Conference on Learning Representations (2022)
38. Wei, X., Zhang, Y., Zhang, X., Gong, R., Zhang, S., Zhang, Q., Yu, F., Liu, X.: Outlier suppression: Pushing the limit of low-bit transformer language models. Advances in Neural Information Processing Systems **35**, 17402–17414 (2022)
39. Xi, H., Li, C., Chen, J., Zhu, J.: Training transformers with 4-bit integers. Advances in Neural Information Processing Systems **36**, 49146–49168 (2023)
40. Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., Han, S.: SmoothQuant: Accurate and efficient post-training quantization for large language models. In: International conference on machine learning. pp. 38087–38099. PMLR (2023)
41. Yao, Z., Yazdani Aminabadi, R., Zhang, M., Wu, X., Li, C., He, Y.: ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. Advances in neural information processing systems **35**, 27168–27183 (2022)
42. Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., Xiao, J.: LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365 (2015)
43. Zhao, R., Hu, Y., Dotzel, J., De Sa, C., Zhang, Z.: Improving neural network quantization without retraining using outlier channel splitting. In: International conference on machine learning. pp. 7543–7552. PMLR (2019)
44. Zhu, X., Li, J., Liu, Y., Ma, C., Wang, W.: A survey on model compression for large language models. Transactions of the Association for Computational Linguistics **12**, 1556–1577 (2024)