Qua^{2}SeDiMo: Quantifiable Quantization Sensitivity of Diffusion Models

Keith G. Mills[1,2], **Mohammad Salameh**[2], **Ruichen Chen**[1], **Negar Hassanpour**[2], **Wei Lu**[3], **Di Niu**[1]

[1]Department of Electrical and Computer Engineering, University of Alberta
[2]Huawei Technologies, Edmonton, Alberta, Canada
[3]Huawei Kirin Solution, Shanghai, China
{kgmills, ruichen1, dniu}@ualberta.ca
{mohammad.salameh, negar.hassanpour2}@huawei.com
robin.luwei@hisilicon.com

Abstract

Diffusion Models (DM) have democratized AI image generation through an iterative denoising process. Quantization is a major technique to alleviate the inference cost and reduce the size of DM denoiser networks. However, as denoisers evolve from variants of convolutional U-Nets toward newer Transformer architectures, it is of growing importance to understand the quantization sensitivity of different weight layers, operations and architecture types to performance. In this work, we address this challenge with Qua^{2}SeDiMo, a mixed-precision Post-Training Quantization framework that generates explainable insights on the cost-effectiveness of various model weight quantization methods for different denoiser operation types and block structures. We leverage these insights to make high-quality mixed-precision quantization decisions for a myriad of diffusion models ranging from foundational U-Nets to state-of-the-art Transformers. As a result, Qua^{2}SeDiMo can construct 3.4-bit, 3.9-bit, 3.65-bit and 3.7-bit weight quantization on PixArt-α, PixArt-Σ, Hunyuan-DiT and SDXL, respectively. We further pair our weight-quantization configurations with 6-bit activation quantization and outperform existing approaches in terms of quantitative metrics and generative image quality.

Project — https://kgmills.github.io/projects/qua2sedimo/

Introduction

Diffusion Models (DM) (Sauer et al. 2024) have become the state of the art in image synthesis. However, at the core of every DM is a large denoiser network, e.g., a U-Net or Diffusion Transformer. The denoiser performs multiple rounds of inference, thus imposing a significant computational burden on the generative process.

One effective method for reducing this burden is quantization (Du, Gong, and Chu 2024) which reduces the bit precision of weights and activations. TFMQ-DM (Huang et al. 2024), a state-of-the-art DM Post-Training Quantization (PTQ) approach, carefully quantizes weight layers associated with time-step inputs to ensure accurate image generation. Q-Diffusion (Li et al. 2023) split the weight layers associated with long residual connections to compensate for bimodal activation distributions and has been integrated into Nvidia's TensorRT framework (NVIDIA 2024). Additionally, ViDiT-Q adopt Large Language Model (LLM) quantization techniques (Xiao et al. 2023) to compress newer Diffusion Transformers (DiT) (Peebles and Xie 2023) like the PixArt models (Chen et al. 2024, 2025). In order to preserve generation quality, each of these techniques employs a calibration set to perform gradient-based calibration for weight quantization. However, till now existing methods still struggle to quantize weight precision below 4-bits (W4) in diffusion models without severely degrading the image generation quality.

To achieve low-bit quantization, mixed-precision quantization has recently been explored for LLMs, although not for DMs yet, which aims to differentiate the bit precision applied to different weights. Talaria (Hohman et al. 2024) is a tool developed by Apple to visualize the impact of different compression techniques applied to different model layers on hardware metrics and latency, which however cannot assess the same impact on task performance. OWQ (Lee et al. 2024) attempts to identify weight column vectors that can generate outlier activations in LLMs, while PB-LLM (Yuan, Shang, and Dong 2024) measures the salience of individual weights. Such outlier or salience information is then used to apply different quantization configurations and bit precisions across weights in an LLM. Although these techniques rely on the Hessian of model weights to identify sensitive weights, the weight saliency is computed by heuristics and is not directly derived to associate with task performance. Another limitation is that as originally designed for LLMs, the granularity adopted is on a fine-grained weight (or weight column) level, rather than on an operator (layer) level or block level. However, such generalizable per-operator or per-model insights are valuable for DMs, which involve a diverse range of model types, e.g., various types of U-Nets and DiTs, as well as a wider range of operation types than in LLMs. These insights, if available, will not only help differentiating quantization method and configuration selection per operation, but also help identify specific operation types, e.g., time-step embeddings or skip-connections that can greatly affect end-to-end performance when improperly quantized.

In this paper, we propose Qua^{2}SeDiMo (pronounced kwa-see-dee-mo), short for **Qua**ntifiable **Qua**ntization **Se**nsitivity of **Di**ffusion **Mo**dels, a framework for discover-

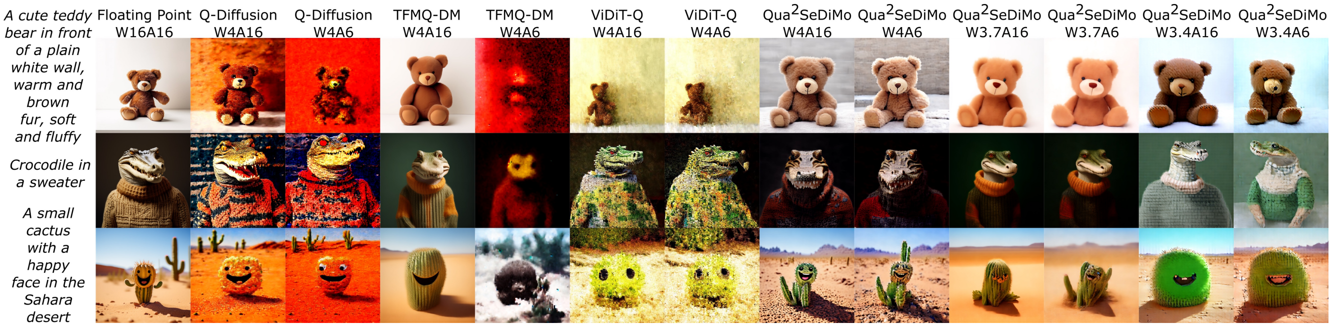

Figure 1: Example 512^2 images generated using PixArt-α. We compare images from the full precision model to those generated by a quantized denoiser using different PTQ techniques. Specifically, we compare Q-Diffusion, TFMQ-DM and ViDiT-Q at W4 precision to three configurations built by Qua^{2}SeDiMo - W4, W3.7 and W3.4 - with and without 6-bit activation quantization.

ing PTQ sensitivity of components in various types of DMs to user-defined end-to-end objectives, including task performance and model complexity. Qua^{2}SeDiMo can identify the individual weights, operation types and block structures that disproportionately impact end-to-end image generation performance when improperly quantized as well as higher-level insights regarding the preference of model and operation types for different quantization schemes and configurations. Furthermore, we combine the algorithm-discovered insights to construct mixed precision, sub 4-bit weight quantization configurations that facilitate high-quality image synthesis, as illustrated by Figure 1 for PTQ performed over a contemporary DiT model, PixArt-α. Our contributions are as follows:

First, unlike previous approaches that use Hessian and other proxies to identify sensitive weights, we propose a method to correlate the quantization method and bit precision of every layer (operation) directly to end-to-end network metrics such as model size or task performance. This is challenging because denoisers in DMs contain hundreds of layers, resulting in exponentially many quantization configuration combinations in the whole network. Moreover, DMs require costly computation to evaluate even with PTQ. However, our method can learn to assign the optimal configuration to each layer by evaluating less than 500 sampled quantization configurations. Qua^{2}SeDiMo achieves this by representing denoiser architectures as graphs, then leveraging an optimization-based GNN explanation method to attribute graph-level performance to individual layers as well as larger block structures like self-attention and temporal embedding layers.

Second, our insights reveal which specific model layers, blocks and quantization methods make sub 4-bit PTQ difficult. Specifically, we find that while U-Nets have a preference for uniform, scale-based quantization (Nahshan et al. 2021), DiT models prefer cluster-based (Han, Mao, and Dally 2016) methods. Additionally, we show that the ResNet blocks in U-Nets are more sensitive than DiT Transformer blocks to quantization, requiring higher bit precision to maintain end-to-end performance and image quality. We also find that the final output layer of DiT models are more sensitive to quantization than their U-Net counterparts.

Third, we construct efficient, mixed-precision weight quantization configurations that generate high-fidelity images. Specifically, we achieve 3.4, 3.9, 3.65, 3.7 and 3.5-bit PTQ on PixArt-α, PixArt-Σ, Hunyuan-DiT (Li et al. 2024), SDXL and DiT-XL/2, respectively, without requiring a calibration dataset. Finally, we pair our weight-quantization with activation quantization, outperforming existing techniques like Q-Diffusion, TFMQ-DM (Huang et al. 2024) and ViDiT-Q (Zhao et al. 2024) in terms of visual quality, FID and CLIP scores.

Related Work

Diffusion models (Sohl-Dickstein et al. 2015; Jiang et al. 2024) are a class of generative models that have been successfully adopted to generate high-fidelity visual content (Sauer et al. 2024). DMs utilize a progressive denoising process to achieve state-of-the-art image generation. Mainstream approaches for high-resolution image generation leverage the latent space of a Variational Auto-Encoder (VAE) (Kingma and Welling 2013) by placing a large denoiser network between the VAE encoder and decoder. Foundational text-to-image (T2I) DMs like SDv1.5 and SDXL (Podell et al. 2024) adopt a hierarchical U-Net-based denoiser architecture that blends Convolutional and Transformer block structures. However, more recent DMs like DiT and SD3 (Esser et al. 2024) use non-hierarchical patch-based architectures based on Vision Transformers (Frumkin, Gope, and Marculescu 2023). Our proposed method, Qua^{2}SeDiMo, is architecture agnostic, so we consider both architecture styles in this work.

The iterative denoising process makes DMs slow. Such a limitation is addressed through model optimization techniques, such as quantization (Gholami et al. 2022). Quantization reduces the bit precision of neural network weights and activation from $\geqslant$16-bit Floating Point (FP) formats to $\leqslant$8-bit Integer (INT)/FP (Shen et al. 2024) formats. There are two broad approaches: Post-Training Quantization (PTQ) (Lin et al. 2024; Lee et al. 2024) can be applied to pretrained model weights, while Quantization-Aware Training (QAT) (Sui et al. 2025) trains or fine-tunes weights in an end-to-end manner using Straight-Through Estimators (Huh et al. 2023) to preserve gradient flow. PTQ is generally computationally inexpensive relative to QAT, though some approaches (Nagel et al. 2020; Li et al. 2021) rely on a calibration dataset of unlabeled sample data. PTQ tends to encounter issues below 4-bit precision (Frumkin, Gope, and Marculescu 2023; Krishnamoorthi 2018) while QAT can quantize Large Language Models (LLM) (Touvron et al. 2023) weights to a very low precision of 1.58-bits (Ma et al. 2024). By contrast, this work achieves sub 4-bit mixed precision PTQ for DM weights without requiring a calibration set, after which activation quantization can be applied with minimal performance loss.

Several PTQ (He et al. 2024; Zhao et al. 2025) and QAT (Wang et al. 2024) approaches exist for DMs. The earliest publications, PTQ4DM (Shang et al. 2023) and Q-Diffusion (Li et al. 2023) emphasize the importance of carefully sampling a calibration dataset to quantize denoiser activations properly. TDQ (So et al. 2024) uses an auxiliary model to generate activation quantization parameters for different denoising steps while QDiffBench (Tang et al. 2025) relaxes activation bit precision at the start and end of the denoising process. Most approaches study the impact of quantization on the denoising process, while a few study the quantization sensitivity of denoiser weight types. For example, Q-Diffusion and TFMQ-DM proposed novel techniques to quantize the long residual connections and time-step embedding layers, respectively. However, these insights are specific to U-Net-based denoisers. By contrast, our work extends this discussion by studying the quantization sensitivity of all weight types and positions while introducing a generalizable method applicable to any denoiser architecture.

Background

We provide a briefing on several integer-based weight quantization methods, including how they are performed and impact on DM generative performance.

Given a tensor W_{FP} with precision N_{FP}, we *quantize* it into W_Q with precision N_Q, thus reducing the tensor size by a factor of ${}^{N_Q}/_{N_{FP}}$. At inference time, we *dequantize* W_Q into W_{DQ} with precision N_{FP}. Although W_{DQ} has the same precision as W_{FP}, quantization introduces an error $\epsilon = \|W_{FP} - W_{DQ}\|_p$, where $p \geqslant 2$; we refer interested readers to Nahshan et al. (2021) for further discussion on p.

One method to perform quantization is by applying K-Means clustering (Han, Mao, and Dally 2016) to W_{FP}. Specifically, we can cluster across the entire tensor or each output channel c_{out} of W_{FP} separately. In either case, W_Q is a matrix of indices corresponding to $K = 2^N$ cluster centroids of precision N_{FP}. However, as W_{DQ} is created by substituting the indices with their corresponding centroids, dequantization is slower and not as hardware-friendly as other methods (Jacob et al. 2018). Additionally, this form of quantization incurs a high FP overhead as centroids are kept in N_{FP}-bit precision. We refer readers to the supplementary for computation of the FP overhead.

In contrast to the costly K-Means, another popular PTQ method is Uniform Affine Quantization (UAQ) (Krishnamoorthi 2018), which involves computing a scale Δ,

$$\Delta = \frac{\texttt{max}(|W_{FP}|)}{2^{N-1}-1}. \qquad (1)$$

Then, tensor quantization is performed by rescaling and clamping W_{FP} as follows,

$$W_Q = \texttt{clamp}(\lfloor \frac{W_{FP}}{\Delta} \rceil, -2^{N-1}+1, 2^{N-1}-1), \qquad (2)$$

where $\lfloor \cdot \rceil$ is the rounding operation. Note that for simplicity, we assume W_{FP} is symmetric at 0 (Xiao et al. 2023). The computation is similar when factoring in a zero-point z for asymmetric quantization (Jacob et al. 2018). UAQ places the tensor values into 2^N evenly-spaced bins of width Δ. UAQ is performed per output channel c_{out} when performing weight quantization. UAQ has two key advantages over K-Means: First, the FP overhead is smaller as we only need to save Δ (and z, if applicable) as FP scalars. Second, UAQ dequantization is simple multiplication $W_{DQ} = \Delta W_Q$ which is very efficient on modern hardware using kernel fusion (Lin et al. 2024), making it the preferred method for deployment on edge devices.

However, note that Eq. 1 is deterministic and may not be optimal. One way to address this problem is to reduce Δ,

$$\Delta_\alpha = \frac{\texttt{max}(|W_{FP}|) \times (1 - (0.01\alpha))}{2^{N-1}-1}. \qquad (3)$$

where $\alpha \in [0, 100)$ is selected to minimize the L_p loss:

$$\min_\alpha \|W_{FP} - \Delta_\alpha W_Q\|_p. \qquad (4)$$

While it is straightforward to apply Eqs. 3 and 4 to individual operators, advanced PTQ methods like

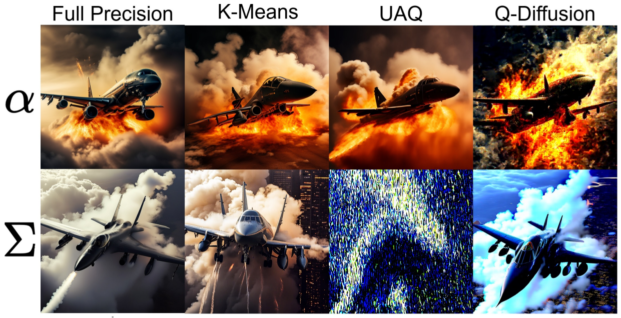

Figure 2: PixArt-α/Σ images at FP16 precision and quantized to W4A16 by K-Means, UAQ and Q-Diffusion. COCO prompt: 'A jet with smoke pouring from its wings'.

Quant. Method	α-W4	α-W3	Σ-W4	Σ-W3
K-Means	32.99	80.65	38.13	133.87
UAQ w/Eq. 3	32.74	77.06	143.25	447.11
Q-Diffusion	40.04	264.51	43.14	366.45

Table 1: FID score for PixArt-α/Σ generating 10k images using MS-COCO prompts under different weight quantization (W{3, 4}A16) configurations. Lower FID is better; FID of the FP model is **34.05** and **36.94** for α and Σ, respectively.

AdaRound (Nagel et al. 2020) and BRECQ (Li et al. 2021) use higher-order loss information to refine Δ. These methods are the basis of advanced DM PTQ schemes like Q-Diffusion. However, they require a calibration set to function, which is not required by K-Means or UAQ when quantizing weights.

As Fig. 2 shows, using any of these methods to quantize denoiser weights down to $N_Q = 4$ produces images that are similar in detail and/or structure to ones generated by the FP model. To quantify the performance, we generate 10k images using MS-COCO (Lin et al. 2014) prompts and measure the Fréchet Inception Distance (FID) (Heusel et al. 2017) using the validation set. As Table 1 shows, all three methods achieve comparable or even lower FID relative to the FP16 model (not unheard of for DM PTQ (Shang et al. 2023; Li et al. 2023; He et al. 2024; Huang et al. 2024)) at 4-bit precision. However, further weight quantization to $N_Q = 3$ yields a sharp rise in FID. We hypothesize that this occurs because PTQ methods quantize every weight to the same precision. Thus, we strike a balance by generating 4 and 3-bit mixed precision weight PTQ configurations.

Methodology

In this section, we elaborate on our search space and describe how to cast a DM denoiser as a graph. We then measure the quantization sensitivity of weight layers and block structures by a GNN explanation method that correlated end-to-end performance with operations and blocks.

We form a search space for each denoiser by varying the bit precision and quantization method applied to each weight layer. The yellow box in Fig. 3 enumerates the available choices. Specifically, we consider two bit-precisions $N_Q = \{3, 4\}$ and three quantization methods: K-Means C, K-Means A and UAQ. K-Means C quantizes each applies output channel c_{out} separately while K-Means A applies clustering to the entire tensor for smaller FP overhead. UAQ utilizes an optimal α value, predetermined using a simple grid search of 10 choices $\alpha \in [0, 10, ..., 80, 90]$ per layer.

In sum, this provides us with 6 quantization choices per weight layer and a total search space size of $6^{\#W}$ where $\#W$ is the number of quantizable weight layers across the entire denoiser architecture. We refer to a denoiser architecture where all weight layer nodes have been assigned a specific bit precision and quantization method as a *quantization configuration*. We can sample various configurations from the search space, apply them to the original FP DM denoiser network, generate images, and then measure end-to-end statistics like FID and average bit precision. Next, we describe how to exploit the properties of graph structures to extract meaningful insights about the denoiser search space.

Operation-Level Sensitivity via Graphs

We represent denoiser architectures as Directed Acyclic Graphs (DAG) (Mills et al. 2023) where nodes represent weight layers, e.g., `nn.Linear` or `nn.Conv2d`, and other operations like 'Add', while the edges model the forward-pass information flow. We provide an example illustration in Figure 3 where red nodes correspond to quantizable weights.

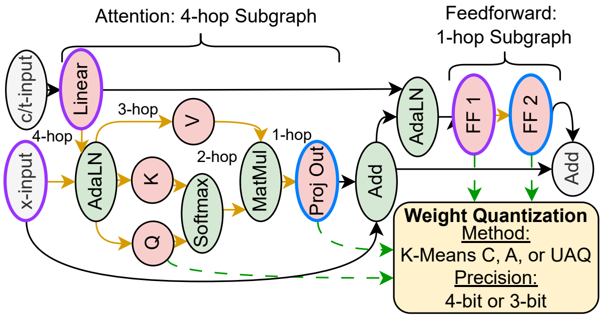

Figure 3: Induced DiT subgraphs. Attention weights (red) are captured in a 4-hop subgraph rooted at 'Proj Out'. The feedforward module is a 1-hop subgraph rooted at 'FF 2'. Yellow box: Each weight layer can be quantized using three methods and two bit-precision levels.

We encode the quantization method, bit precision, operation type (e.g., attention 'query' linear layer) and positional information like Transformer block index as node features. This encoding allows us to extract quantifiable insights on the sensitivity of denoiser architectures by identifying the operation types, block structures, positions and quantization methods that contribute to high end-to-end performance and low average bit precision. To achieve this, we introduce the following explanation method for Graph Neural Network (GNN) (Brody, Alon, and Yahav 2022; Fey and Lenssen 2019) regressors:

Let $\mathcal{G}$ be a denoiser graph with specific quantization configuration, annotated with ground-truth label $y_{\mathcal{G}}$, e.g., negative FID $y_{\mathcal{G}} = -FID_{\mathcal{G}}$. $\mathcal{G}$ contains a node set $\mathcal{V}_{\mathcal{G}}$, whose features describe the quantization settings for each weight layer, and edge set $\mathcal{E}_{\mathcal{G}}$. We can then use a GNN to learn to predict $y_{\mathcal{G}}$ given $\mathcal{G}$. A GNN contains $m \in [0, M]$ layers: an initial embedding layer followed by M message passing layers, each of which produces an embedding h_v^m for every node $v_i \in \mathcal{V}_{\mathcal{G}}$. Node embeddings from a given GNN layer can be aggregated to form a vector embedding for the graph, e.g., by averaging them,

$$h_{\mathcal{G}}^m = \frac{1}{|\mathcal{V}_{\mathcal{G}}|} \sum_{v \in \mathcal{V}} h_v^m. \tag{5}$$

We can then apply a simple MLP to the graph embedding to make a prediction $y'_{\mathcal{G}} = \texttt{MLP}(h_{\mathcal{G}}^M)$. A simple GNN can learn by minimizing a loss, e.g., mean-squared error $\|y_{\mathcal{G}} - y'_{\mathcal{G}}\|_2$, which we denote as $\mathcal{L}_{orig}$. This formulation is a typical black box model and while it can estimate $y_{\mathcal{G}}$, it cannot extract quantifiable insights. Instead, this is accomplished by incorporating an additional loss term:

$$\mathcal{L} = \mathcal{L}_{orig}(y_{\mathcal{G}}, y'_{\mathcal{G}}) + \frac{1}{M+1} \sum_{m=0}^{M} \mathcal{L}_{rank}(y_{\mathcal{G}}, \|h_{\mathcal{G}}^m\|_1), \tag{6}$$

where $\mathcal{L}_{rank}$ is a ranking loss that directly interfaces with the graph embeddings $h_{\mathcal{G}}^m$ from each GNN layer. The exact choice of $\mathcal{L}_{rank}$ is important. A straight-forward idea is to choose the differentiable spearman ρ loss that Blondel et al. (2020) provide, in order to maximize the Spearman Rank Correlation Coefficient (SRCC). However, SRCC assigns equal importance to the predicted rank of every entry considered, weighing entries that *minimize* and *maximize* the ground-truth equally. In contrast, depending on how we compute $y_{\mathcal{G}}$, our goal is to extract insights from the graphs that explicitly maximize $y_{\mathcal{G}}$. Therefore, one alternative is to maximize the Normalized Discounted Cumulative Gain (NDCG), an Information Retrieval metric that prioritizes the correct ranking of high-relevance (i.e., high $y_{\mathcal{G}}$) samples, by implementing the LambdaRank (Burges 2010) loss.

Regardless of the choice of $\mathcal{L}_{rank}$, the intuition behind our approach is to compress $h_{\mathcal{G}}^m$ into its scalar L1 norm and associate it with the ground-truth $y_{\mathcal{G}}$. Then, because $h_{\mathcal{G}}^m$ is computed by averaging all node embeddings per Eq. 5, the GNN is forced to learn which nodes contribute or detract from $y_{\mathcal{G}}$. As such, we are able to treat the scalar norm of the node embedding $\left\|h_v^m\right\|_1$ as a numerical score where high $\left\|h_v^m\right\|_1$ correspond to higher $y_{\mathcal{G}}$.

Finally, using this setup we can construct highly desirable quantization configurations. Assume we train a predictor using Eq. 6 where $y_{\mathcal{G}} = -FID_{\mathcal{G}}$. We can select the optimal bit precision and quantization method for every weight layer node simply by iterating across all possible combinations (i.e., 6 per node according to Fig. 3) and selecting the configuration that produces the highest score $\left\|h_v^0\right\|_1$.

Block-level Quantization Sensitivity

While we have shown how Eq. 6 produces sensitivity scores for individual weight nodes, it is non-trivial to extend this idea to larger denoiser components, e.g., ResNet blocks or time-step embedding modules. To do this, we model these structures as subgraphs contained within the overall denoiser graph. Each subgraph contains a root node corresponding to a single operation. The root only aggregates information (e.g., quantization method and precision features) from the other nodes in its subgraph, allowing us to interpret its score as representative of the entire subgraph block structure.

As a practical example, Figure 3 provides an illustration where a DiT-XL/2 Transformer block is split into attention and feedforward subgraphs, rooted at the 'Proj Out' and 'FF 2' weight nodes, respectively. Therefore, we cast $\left\|h^4_{ProjOut}\right\|_1$ and $\left\|h^1_{FF2}\right\|_1$ as the sensitivity scores for the attention and feedforward modules, respectively. Further, we can then construct high-quality quantization configuration by looping over all quantization method and bit precision choices for each weight layer node in the subgraph and selecting the option that yields the greatest score.

Note that this schema contains several design choices and details about the block structures we cast as subgraphs and which weights should be chosen as roots. Generally, we root our subgraphs at the last weighted layer of the block structure, though there are some exceptions and we provide extensive details in the supplementary.

Further, it should be noted that we are able to quantify the sensitivity of large block structures by exploiting the message passing properties of GNNs. Formally, given an arbitrary node $v_i \in \mathcal{V}_{\mathcal{G}}$, a single GNN layer will propagate latent embeddings h_v from all nodes in the immediate 1-hop neighborhood $\mathcal{N}(v_i) = \{v_j \in \mathcal{V}_{\mathcal{G}} | (v_j, v_i) \in \mathcal{E}_{\mathcal{G}}\}$ into the embedding of v_i, h_v. Applying another GNN layer will further propagate information from all nodes in the 2-hop neighborhood of v_i into h_v.

Denoiser	**FID**	$\#W$	**#Configs**	**FID Range**
PixArt-α	99.67	287	372	[92.29, 507.87]
PixArt-Σ	102.67	287	402	[98.08, 662.62]
Hunyuan	93.31	487	340	[99.68, 416.45]
SDXL	112.44	803	447	[101.57, 704.89]
SDv1.5	88.17	294	378	[92.90, 584.56]
DiT-XL/2	85.02	201	356	[64.70, 498.65]

Table 2: Denoiser search space statistics: number of sampled configurations, number of quantizable layers $\#W$ and FID range. FID is the performance of the W16A16 model.

We define the m-hop neighborhood $\mathcal{N}^m(v_i) \subseteq \mathcal{V}_{\mathcal{G}}$ as $\mathcal{N}^m(v_i) = \{v_j \in \mathcal{V}_{\mathcal{G}} | \langle v_j, v_i \rangle \leqslant m\}$, where $\langle v_j, v_i \rangle$ is the length of the shortest path between v_j and v_i. By induction, applying $m > 0$ GNN layers will aggregate information from all nodes in $\mathcal{N}^m(v_i)$ into h_v^m. As such, we extend the meaning of h_v^m from not simply the embedding of node v_i, but as the embedding of the subgraph containing all nodes in $\mathcal{N}^m(v_i)$ that is rooted at v_i. Likewise, we can now interpret $\left\|h_v^m\right\|_1$ as the quantifiable score of this subgraph.

Experimental Results and Discussion

In this section we evaluate Qua^{2}SeDiMo on several T2I DMs: PixArt-α, PixArt-Σ, Hunyuan and SDXL. Due to space constraints, additional results on SDv1.5 and DiT-XL/2 can be found in the supplementary. We apply our scheme to find cost-effective quantization configurations that minimize both FID and model size while providing some visual examples. We then compare our found quantization configurations to existing DM PTQ literature. Finally, we share some insights on the quantization sensitivity of denoiser architectures.

Pareto Optimal Mixed-Precision Denoisers

To train Qua^{2}SeDiMo predictors, we sample and evaluate hundreds of randomly selected quantization configurations per denoiser architecture. To evaluate a configuration, we generate 1000 images and compute the FID score relative to a ground-truth image set. Specifically, for all T2I DMs, we use prompts and images from the COCO 2017 validation set to generate images and compute FID, respectively. For DiT-XL/2, we generate one image per ImageNet class and measure FID against the ImageNet validation set. We generate 1024^2 images using PixArt-Σ and Hunyuan and set a resolution of 512^2 for all other DMs. We report additional details, e.g., number of steps, in the supplementary. Table 2 lists statistics for each denoiser search space.

We focus on maximizing visual quality while minimizing the average bit precision $\overline{Bits}$ of the denoiser. To achieve this, we train Qua^{2}SeDiMo to predict $y = -FID - \lambda\overline{Bits}$.

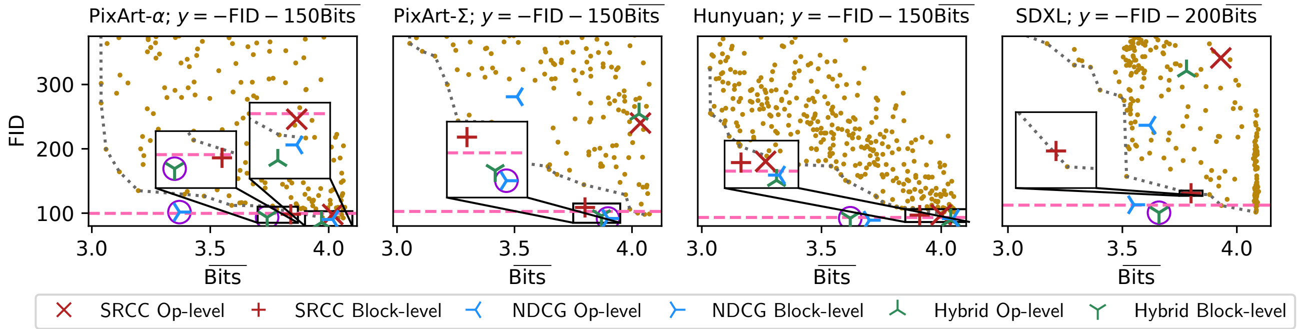

Figure 4: Results on PixArt-α, PixArt-Σ, Hunyuan and SDXL under constrained optimization to minimize FID and $\overline{Bits}$. Dashed horizonal line denotes the FID of the W16A16 model. Dotted grey line denotes the Pareto frontier constructed from our corpus of randomly sampled configurations (yellow dots). For each predictor ensemble, we generate two quantization configurations: 'Op-level' for individual weight layers and 'Block-level' for subgraph structures. Purple circles denote configurations we later investigate to generate images and draw insights from. Best viewed in color.

Specifically, λ re-scales $\overline{Bits}$ to determine how we weigh model size against performance (FID). As such, λ is a denoiser-dependent coefficient. Further, we consider three ranking losses $\mathcal{L}_{rank}$: the differentiable spearman ρ from Blondel et al. (2020) that maximizes SRCC, LambdaRank which maximizes NDCG and a 'Hybrid' loss that sums both of them to maximize SRCC and NDCG.

To leverage our limited training data per Table 2, we follow Mills et al. (2024) and train a predictor ensemble to generate subgraph scores using different data splits. Specifically, we split the corpus of quantization configurations into $K = 5$ folds, each containing an 80%/20% training/validation data split with disjoint validation partitions. We measure validation set performance for each predictor in the ensemble and use it as a weight to re-scale predictor scores. Detailed predictor hyperparameters and other details can be found in the supplementary.

Finally, we construct two quantization configurations: Operation and Block-level. Operation-level optimization enumerates each weight layer nodes v and selects the quantization method and bit precision that produces the highest score $\left\|h_v^0\right\|_1$. Block-level optimization enumerates settings for all nodes in block subgraphs to maximize the score of the subgraph root node.

Figure 4 reports our findings on PixArt-α, PixArt-Σ, Hunyuan and SDXL for $y = -FID - \lambda\overline{Bits}$. Additional results for for $y = -FID$ can be found in the supplementary. We observe that quantization configurations generated using the subgraph 'Block-level' approach with the NDCG and Hybrid losses are consistently superior to those found using the baseline SRCC loss and the Pareto frontier of randomly sampled training configurations. Generally, 'Op-level' optimization fails outright or fixates on the low-FID, high $\overline{Bits}$ region in the bottom right corner, but in either case, fails to produce configurations that optimize $y = -FID - \lambda\overline{Bits}$.

In terms of specific quantization configurations, on PixArt-α, we are able to find a remarkable quantization configuration that achieves 3.4-bit precision with comparable FID to the W16A16 model. Impressively, we also find 3.7-bit configurations that outperform the W16A16 model FID on PixArt-α and SDXL as well as a 3.65-bit Hunyuan configuration. Finally, PixArt-Σ proves to be the hardest denoiser to optimize as FID of random configurations rises sharply when quantizing below 4-bits, yet Qua^{2}SeDiMo is still able to construct several low-FID, 3.9-bit quantization configurations. Next, we compare our mixed-precision configurations to several prior 4-bit methods.

Comparison with Related Literature

We quantitatively and qualitatively compare Qua^{2}SeDiMo to several existing DM PTQ methods: Q-Diffusion, TFMQ-DM and ViDiT-Q. Specifically, we quantize weights down to 4-bits (W4) or lower, while considering three activation precision levels: A16, A8 and A6. Q-Diffusion and TFMQ-DM compute activation scales using a calibration set, while ViDiT-Q and Qua^{2}SeDiMo employ the online, patch-based technique from Microsoft's ZeroQuant (Yao et al. 2022).

For each method, we sample 10k unique `(caption, image)` pairs from the COCO 2014 validation set and generate one image per caption and compute FID using the selected validation set images. We also compute the CLIP score (Hessel et al. 2021) using the ViT-B/32 backbone and COCO validation captions.

Table 3 reports our findings on PixArt-α. We note that how that at every activation bit precision level, the W4 configuration built by Qua^{2}SeDiMo achieves the best FID and CLIP metrics while the W3.7 and W3.4 variants are not far behind, especially in terms of CLIP score. The most competitive method is ViDiT-Q, followed by TFMQ-DM. In contrast, we deliberately re-ran Q-Diffusion using the online ZeroQuant activation quantization (Q-Diffusion OAQ) as its original mechanism catastrophically fails in the W4A8 and W4A6 settings for DiTs.

Next, Table 4 provides an analogous comparison for PixArt-Σ. This denoiser is harder to quantize than its predecessor, yet despite this we are still able to find a W3.9-bit precision quantization configuration that outperforms competing methods across all activation precision levels. Cu-

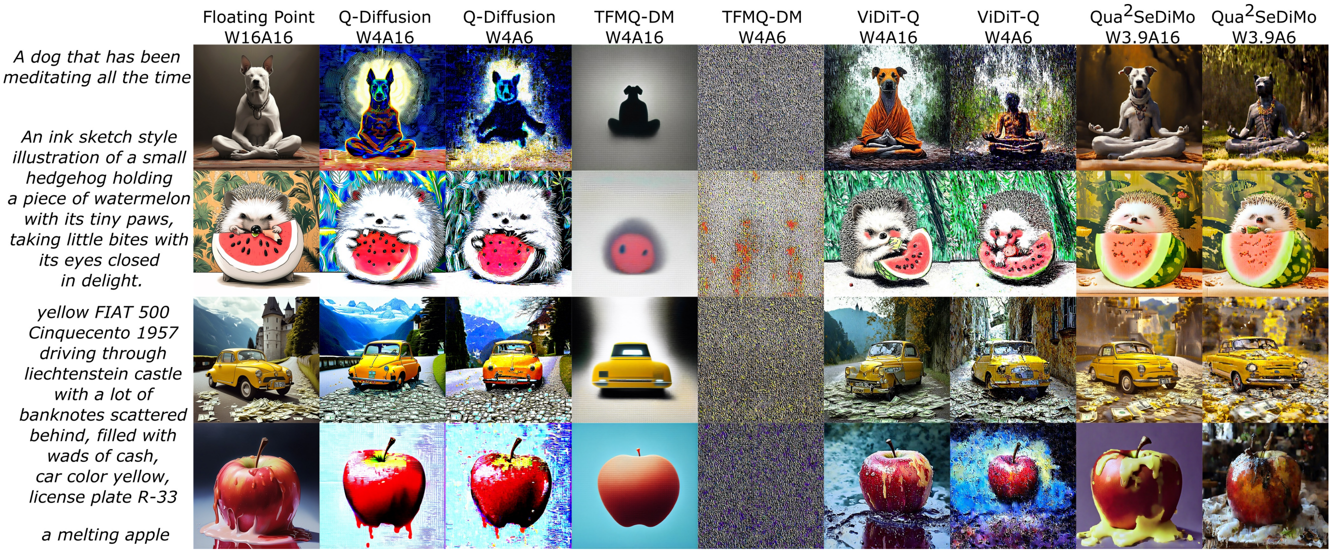
Floating Point W16A16
Q-Diffusion W4A16
Q-Diffusion W4A6
TFMQ-DM W4A16
TFMQ-DM W4A6
ViDiT-Q W4A16
ViDiT-Q W4A6
Qua^{2}SeDiMo W3.9A16
Qua^{2}SeDiMo W3.9A6
A dog that has been meditating all the time
An ink sketch style illustration of a small hedgehog holding a piece of watermelon with its tiny paws, taking little bites with its eyes closed in delight.
yellow FIAT 500 Cinquecento 1957 driving through liechtenstein castle with a lot of banknotes scattered behind, filled with wads of cash, car color yellow, license plate R-33
a melting apple

Figure 5: PixArt-Σ example images and comparison with related work. Resolution: 1024^2.

Method	Precision	FID ↓	CLIP ↑
Full Precision	W16A16	34.05	0.3102
Q-Diffusion	W4A16	41.93	*0.2992*
TFMQ-DM	W4A16	38.67	0.2905
ViDiT-Q	W4A16	33.98	0.2924
Qua^{2}SeDiMo	W4A16	**29.02**	**0.3056**
Qua^{2}SeDiMo	W3.7A16	*33.06*	0.2931
Qua^{2}SeDiMo	W3.4A16	35.51	0.2919
Q-Diffusion	W4A8	373.62	0.2126
Q-Diffusion OAQ	W4A8	52.20	0.2924
TFMQ-DM	W4A8	64.73	0.2594
ViDiT-Q	W4A8	*39.11*	0.2925
Qua^{2}SeDiMo	W4A8	**38.39**	**0.3015**
Qua^{2}SeDiMo	W3.7A8	51.48	*0.2928*
Qua^{2}SeDiMo	W3.4A8	53.61	0.2921
Q-Diffusion	W4A6	382.59	0.2121
Q-Diffusion OAQ	W4A8	70.96	0.2865
TFMQ-DM	W4A6	90.90	0.2545
ViDiT-Q	W4A6	*56.54*	0.2852
Qua^{2}SeDiMo	W4A6	**54.59**	**0.2950**
Qua^{2}SeDiMo	W3.7A6	58.05	*0.2906*
Qua^{2}SeDiMo	W3.4A6	59.75	0.2898

Table 3: Quantization comparison on PixArt-α generating 10k 512^2 images using COCO 2014 prompts. Q-Diffusion OAQ pairs the original method with online activation quantization. Best/second best results in bold/italics.

riously, more traditional PTQ approaches for U-Nets like Q-Diffusion are more competitive at this level for weight quantization, but must still discard calibration-based activation quantization in favour of the online approach.

In terms of qualitative comparison, recall Fig. 1 which shows generated images on PixArt-α, while Figure 5 provides images for PixArt-Σ. We note the robustness of Qua^{2}SeDiMo, as even the sub 4-bit configurations can generate acceptable images with low-bit activation quantization.

Method	Precision	FID ↓	CLIP ↑
Full Precision	W16A16	36.94	0.3154
Q-Diffusion	W4A16	38.27	0.3115
TFMQ-DM	W4A16	39.15	0.3096
ViDiT-Q	W4A16	*31.21*	*0.3132*
Qua^{2}SeDiMo	W3.9A16	**30.34**	**0.3154**
Q-Diffusion	W4A8	533.49	0.2219
Q-Diffusion OAQ	W4A8	*36.89*	*0.3127*
TFMQ-DM	W4A8	62.98	0.2987
ViDiT-Q	W4A8	37.17	0.3072
Qua^{2}SeDiMo	W3.9A8	**35.61**	**0.3136**
Q-Diffusion	W4A6	533.49	0.2117
Q-Diffusion OAQ	W4A6	*74.83*	*0.2981*
TFMQ-DM	W4A6	154.13	0.2600
ViDiT-Q	W4A6	87.47	0.2837
Qua^{2}SeDiMo	W3.9A6	**74.29**	**0.2999**

Table 4: Quantization comparison on PixArt-Σ generating 10k 1024^2 images using COCO 2014 prompts. Same experimental setup as Table 3.

Finally, Table 5 provides the results of a human preference study qualitatively comparing images produced by Qua^2 SeDiMo with other methods. These studies consisted of 20 human participants and 118 images. Each participant was given a prompt and the corresponding generated images for four W4A8 models quantized by different methods and asked to choose which image was best in terms of visual quality and prompt adherence. Users were given a 'Cannot Decide' option but asked to invoke it sparingly (13 times for α & 15 for Σ). The results of this survey show a significant preference for the images produced by Qua^{2}SeDiMo compared to other approaches for both PixArt models.

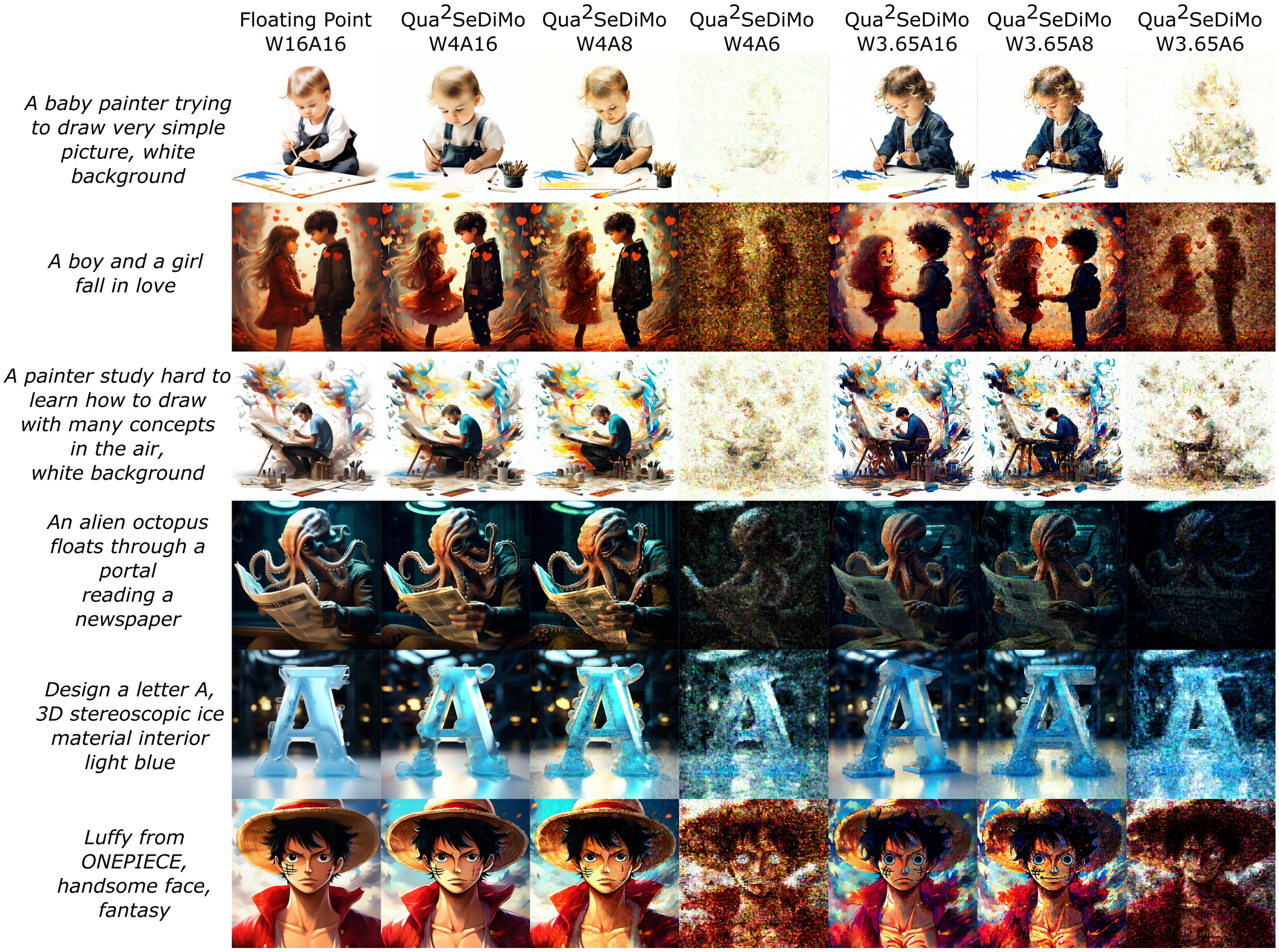
Floating Point
W16A16
Qua^{2}SeDiMo
W4A16
Qua^{2}SeDiMo
W4A8
Qua^{2}SeDiMo
W4A6
Qua^{2}SeDiMo
W3.65A16
Qua^{2}SeDiMo
W3.65A8
Qua^{2}SeDiMo
W3.65A6
A baby painter trying to draw very simple picture, white background
A boy and a girl fall in love
A painter study hard to learn how to draw with many concepts in the air, white background
An alien octopus floats through a portal reading a newspaper
Design a letter A, 3D stereoscopic ice material interior light blue
Luffy from ONEPIECE, handsome face, fantasy

Figure 6: Hunyuan-DiT example images. Resolution: 1024^2.

Model	Method	Precision	#Votes (%) ↑
PixArt-α	Q-Diffusion	W4A8	*34 (28.81%)*
	TFMQ-DM	W4A8	10 (8.47%)
	ViDiT-Q	W4A8	10 (8.47%)
	Qua^{2}SeDiMo	W4A8	**51 (43.22%)**
PixArt-Σ	Q-Diffusion	W4A8	*28 (23.73%)*
	TFMQ-DM	W4A8	3 (2.54%)
	ViDiT-Q	W4A8	*28 (23.73%)*
	Qua^{2}SeDiMo	W3.9A8	**44 (37.29%)**

Table 5: User preference study between Qua^{2}SeDiMo and baseline methods on PixArt-α/Σ. $N = 118$. Best/second best results in bold/italics.

Results on Hunyuan-DiT

Table 6 compares Qua^{2}SeDiMo to other methods on Hunyuan. We observe better performance in terms of lower FID and higher CLIP at W4 and W3.65-bit precision. However, compared to PixArt DiTs, Hunyuan is more difficult to adequately quantize to A6-bit precision, as all methods experi-

Method	Precision	FID ↓	CLIP ↑
Full Precision	W16A16	41.92	0.3089
Q-Diffusion	W4A16	42.09	0.3095
TFMQ-DM	W4A16	42.50	0.3066
Qua^{2}SeDiMo	W4A16	**40.25**	**0.3162**
Qua^{2}SeDiMo	W3.65A16	41.97	0.3158
Q-Diffusion OAQ	W4A8	71.99	0.2974
TFMQ-DM	W4A8	72.15	0.2929
Qua^{2}SeDiMo	W4A8	**51.91**	**0.3158**
Qua^{2}SeDiMo	W3.65A8	71.32	0.3078
Q-Diffusion OAQ	W4A6	171.40	0.2395
TFMQ-DM	W4A6	175.32	0.2375
Qua^{2}SeDiMo	W4A6	**155.19**	**0.2438**
Qua^{2}SeDiMo	W3.65A6	167.67	0.2322

Table 6: Quantization comparison on Hunyuan-DiT generating 10k 1024^2 images using COCO 2014 prompts. Same experimental setup as Table 4. Best result in bold.

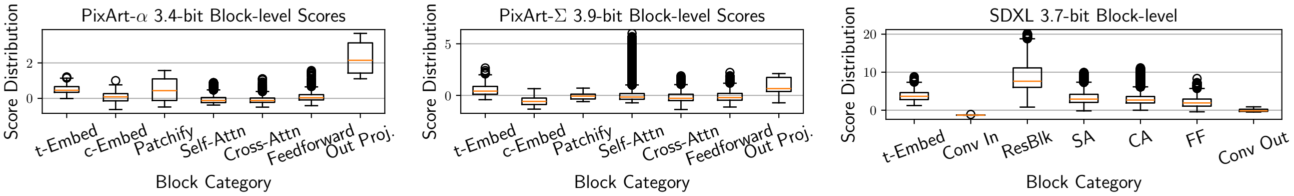

Figure 7: Block-level box-plots for sub 4-bit PixArt-α, PixArt-Σ and SDXL configurations.

ence a substantial performance degradation at this level.

This degradation is visualized in Figure 6, which provides images generated by Hunyuan when quantized by Qua^{2}SeDiMo. Specifically, we examine images at W{4, 3.65}A{16, 8, 6}-bit precision levels. This comparison visually contrasts the effect of weight and activation quantization. Specifically, weight quantization controls higher-level aspects of an image, e.g., the child's hair and clothing, artstyle of the boy and girl, shape of the octopus' head and Luffy's facial expression. In contrast, there is an inverse relationship between the activation bit precision and the amount of undesirable noise present.

Extracted Insights

We examine some of the quantization sensitivity insights Qua^{2}SeDiMo provides. Figure 7 plots the sensitivity score distributions for different subgraph block types, e.g., Self-Attention (SA) or Cross-Attention (CA). We interpret these scores as follows: If the score distribution for a block type has a large range with high outliers, it means there are quantization block settings which are crucial to maintaining efficient performance. If the distribution mean and range are low, the block is not very important.

Corroborating Huang et al. (2024), we find that the time embedding module (t-Embed) is an important block as the score distribution for each denoiser has a large mean, wide range, and a number of high-scoring outliers. In the SDXL U-Net, the time parameter interfaces with each convolutional ResNet Block (ResBlk), which carries the highest score distribution for that denoiser. In contrast, the condition embedding (c-Embed) in PixArt-α/Σ is quite low, indicating that adequate quantization of prompt embedding layers is less crucial. Also, note the moderate variance in the input 'Patchify' and 'Out Proj.' layers of PixArt DMs, indicating great importance, especially in contrast to the analogous 'Conv In' and 'Conv Out' in SDXL.

Finally, Figure 8 shows stacked bar plots illustrating the distribution of quantization methods and bit precisions selected to form the optimal sub 4-bit configurations. That is, PixArt-α contains 4 t-Embed linear layers, all kept at 4-bit precision: 3 using UAQ, and one using K-Means C. The model also contains 28 self-attention key (SA-K; one for each transformer block) layers quantized primarily using K-Means C/A at 3 and 4-bit precisions. It also has a single output (Out) layer quantized to 3-bits using UAQ. In general, these findings show that DiT blocks have a slight preference for K-Means-based quantization, whereas by contrast, the SDXL U-Net strongly prefers UAQ quantization.

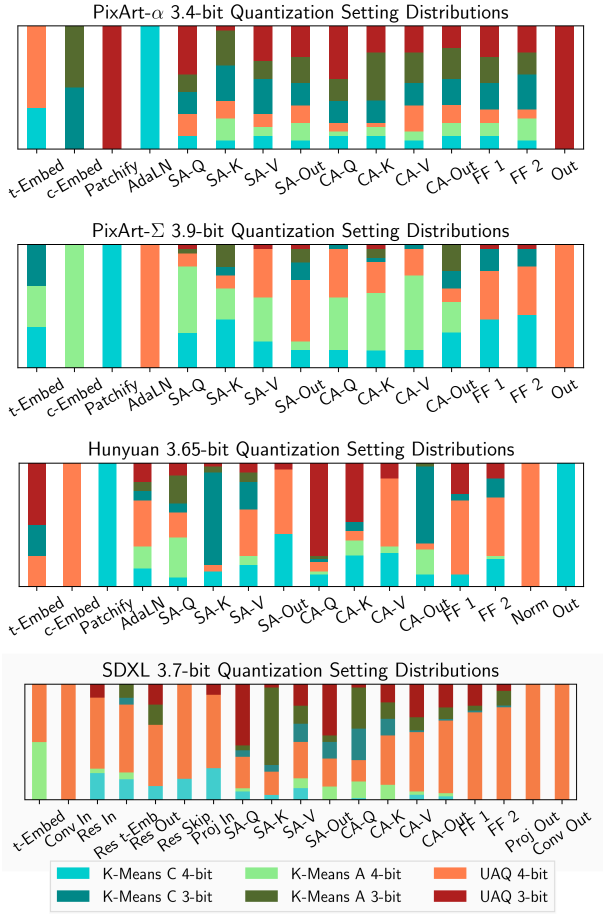

Figure 8: Stacked quantization method bar plots for several sub 4-bit quantization configurations. Best viewed in color.

Conclusion

We propose Qua^{2}SeDiMo, a mixed-precision DM weight PTQ framework. We cast denoisers as large search spaces characterized by choice of bit precision and quantization method per weight layer. It extracts quantifiable insights about how these choices correlate to end-to-end metrics such as FID and average bit precision. We use these insights to construct high-quality sub 4-bit weight quantization configurations for several popular T2I denoisers such as PixArt-α/Σ, Hunyuan and SDXL. We pair this method with low-bit activation quantization to outperform existing methods and generate convincing visual content.

Acknowledgements

This work is partially funded by an Alberta Innovates Graduate Student Scholarship (AIGSS). Alberta Innovates is an organization that expands the horizon of possibilites to solve today's challenges and create a healthier and more prosperous future for Alberta and the world.

References

Blondel, M.; Teboul, O.; Berthet, Q.; and Djolonga, J. 2020. Fast differentiable sorting and ranking. In *International Conference on Machine Learning*, 950–959. PMLR.

Brody, S.; Alon, U.; and Yahav, E. 2022. How Attentive are Graph Attention Networks? In *International Conference on Learning Representations*.

Burges, C. J. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581): 81.

Chen, J.; Ge, C.; Xie, E.; Wu, Y.; Yao, L.; Ren, X.; Wang, Z.; Luo, P.; Lu, H.; and Li, Z. 2025. PixArt-Σ: Weak-to-Strong Training of Diffusion Transformer for 4K Text-to-Image Generation. In *Computer Vision – ECCV 2024*, 74–91. Springer Nature Switzerland. ISBN 978-3-031-73411-3.

Chen, J.; Yu, J.; Ge, C.; Yao, L.; Xie, E.; Wang, Z.; Kwok, J. T.; Luo, P.; Lu, H.; and Li, Z. 2024. PixArt-α: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Du, D.; Gong, G.; and Chu, X. 2024. Model Quantization and Hardware Acceleration for Vision Transformers: A Comprehensive Survey. *arXiv preprint arXiv:2405.00314*.

Esser, P.; Kulal, S.; Blattmann, A.; Entezari, R.; Müller, J.; Saini, H.; Levi, Y.; Lorenz, D.; Sauer, A.; Boesel, F.; et al. 2024. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*.

Fey, M.; and Lenssen, J. E. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Frumkin, N.; Gope, D.; and Marculescu, D. 2023. Jumping through local minima: Quantization in the loss landscape of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16978–16988.

Gholami, A.; Kim, S.; Dong, Z.; Yao, Z.; Mahoney, M. W.; and Keutzer, K. 2022. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*, 291–326. Chapman and Hall/CRC.

Han, S.; Mao, H.; and Dally, W. J. 2016. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. In Bengio, Y.; and LeCun, Y., eds., *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

He, Y.; Liu, L.; Liu, J.; Wu, W.; Zhou, H.; and Zhuang, B. 2024. Ptqd: Accurate post-training quantization for diffusion models. *Advances in Neural Information Processing Systems*, 36.

Hessel, J.; Holtzman, A.; Forbes, M.; Bras, R. L.; and Choi, Y. 2021. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30.

Hohman, F.; Wang, C.; Lee, J.; Görtler, J.; Moritz, D.; Bigham, J. P.; Ren, Z.; Foret, C.; Shan, Q.; and Zhang, X. 2024. Talaria: Interactively optimizing machine learning models for efficient inference. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 1–19.

Huang, Y.; Gong, R.; Liu, J.; Chen, T.; and Liu, X. 2024. Tfmq-dm: Temporal feature maintenance quantization for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7362–7371.

Huh, M.; Cheung, B.; Agrawal, P.; and Isola, P. 2023. Straightening out the straight-through estimator: Overcoming optimization challenges in vector quantized networks. In *International Conference on Machine Learning*, 14096–14113. PMLR.

Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; and Kalenichenko, D. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2704–2713.

Jiang, L.; Hassanpour, N.; Salameh, M.; Singamsetti, M. S.; Sun, F.; Lu, W.; and Niu, D. 2024. FRAP: Faithful and Realistic Text-to-Image Generation with Adaptive Prompt Weighting. *arXiv preprint arXiv:2408.11706*.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Krishnamoorthi, R. 2018. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*.

Lee, C.; Jin, J.; Kim, T.; Kim, H.; and Park, E. 2024. OWQ: Outlier-Aware Weight Quantization for Efficient Fine-Tuning and Inference of Large Language Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 13355–13364.

Li, X.; Liu, Y.; Lian, L.; Yang, H.; Dong, Z.; Kang, D.; Zhang, S.; and Keutzer, K. 2023. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 17535–17545.

Li, Y.; Gong, R.; Tan, X.; Yang, Y.; Hu, P.; Zhang, Q.; Yu, F.; Wang, W.; and Gu, S. 2021. BRECQ: Pushing the Limit of Post-Training Quantization by Block Reconstruction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Li, Z.; Zhang, J.; Lin, Q.; Xiong, J.; Long, Y.; Deng, X.; Zhang, Y.; Liu, X.; Huang, M.; Xiao, Z.; Chen, D.; He, J.; Li, J.; Li, W.; Zhang, C.; Quan, R.; Lu, J.; Huang, J.; Yuan, X.; Zheng, X.; Li, Y.; Zhang, J.; Zhang, C.; Chen, M.; Liu, J.; Fang, Z.; Wang, W.; Xue, J.; Tao, Y.; Zhu, J.; Liu, K.; Lin, S.;

Sun, Y.; Li, Y.; Wang, D.; Chen, M.; Hu, Z.; Xiao, X.; Chen, Y.; Liu, Y.; Liu, W.; Wang, D.; Yang, Y.; Jiang, J.; and Lu, Q. 2024. Hunyuan-DiT: A Powerful Multi-Resolution Diffusion Transformer with Fine-Grained Chinese Understanding. arXiv:2405.08748.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Chen, W.-M.; Wang, W.-C.; Xiao, G.; Dang, X.; Gan, C.; and Han, S. 2024. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. *Proceedings of Machine Learning and Systems*, 6: 87–100.

Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C. L.; and Dollár, P. 2014. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, 740–755. Cham: Springer International Publishing.

Ma, S.; Wang, H.; Ma, L.; Wang, L.; Wang, W.; Huang, S.; Dong, L.; Wang, R.; Xue, J.; and Wei, F. 2024. The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits. *arXiv preprint arXiv:2402.17764*.

Mills, K. G.; Han, F. X.; Salameh, M.; Lu, S.; Zhou, C.; He, J.; Sun, F.; and Niu, D. 2024. Building Optimal Neural Architectures using Interpretable Knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5726–5735.

Mills, K. G.; Han, F. X.; Zhang, J.; Chudak, F.; Safari Mamaghani, A.; Salameh, M.; Lu, W.; Jui, S.; and Niu, D. 2023. GENNAPE: Towards Generalized Neural Architecture Performance Estimators. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(8): 9190–9199.

Nagel, M.; Amjad, R. A.; Van Baalen, M.; Louizos, C.; and Blankevoort, T. 2020. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, 7197–7206. PMLR.

Nahshan, Y.; Chmiel, B.; Baskin, C.; Zheltonozhskii, E.; Banner, R.; Bronstein, A. M.; and Mendelson, A. 2021. Loss aware post-training quantization. *Machine Learning*, 110(11): 3245–3262.

NVIDIA. 2024. NVIDIA TensorRT Accelerates Stable Diffusion Nearly 2x Faster with 8-bit Post-Training Quantization. https://developer.nvidia.com/blog/tensorrt-accelerates-stable-diffusion-nearly-2x-faster-with-8-bit-post-training-quantization/. Accessed: 2024-08-15.

Peebles, W.; and Xie, S. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4195–4205.

Podell, D.; English, Z.; Lacey, K.; Blattmann, A.; Dockhorn, T.; Müller, J.; Penna, J.; and Rombach, R. 2024. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Sauer, A.; Boesel, F.; Dockhorn, T.; Blattmann, A.; Esser, P.; and Rombach, R. 2024. Fast high-resolution image synthesis with latent adversarial diffusion distillation. In *SIGGRAPH Asia 2024 Conference Papers*, 1–11.

Shang, Y.; Yuan, Z.; Xie, B.; Wu, B.; and Yan, Y. 2023. Post-training quantization on diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1972–1981.

Shen, H.; Mellempudi, N.; He, X.; Gao, Q.; Wang, C.; and Wang, M. 2024. Efficient post-training quantization with fp8 formats. *Proceedings of Machine Learning and Systems*, 6: 483–498.

So, J.; Lee, J.; Ahn, D.; Kim, H.; and Park, E. 2024. Temporal dynamic quantization for diffusion models. *Advances in Neural Information Processing Systems*, 36.

Sohl-Dickstein, J.; Weiss, E.; Maheswaranathan, N.; and Ganguli, S. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, 2256–2265. PMLR.

Sui, Y.; Li, Y.; Kag, A.; Idelbayev, Y.; Cao, J.; Hu, J.; Sagar, D.; Yuan, B.; Tulyakov, S.; and Ren, J. 2025. BitsFusion: 1.99 bits Weight Quantization of Diffusion Model. *Advances in Neural Information Processing Systems*, 37.

Tang, S.; Wang, X.; Chen, H.; Guan, C.; Wu, Z.; Tang, Y.; and Zhu, W. 2025. Post-training quantization with progressive calibration and activation relaxing for text-to-image diffusion models. In *European Conference on Computer Vision*, 404–420. Springer.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, H.; Shang, Y.; Yuan, Z.; Wu, J.; and Yan, Y. 2024. QuEST: Low-bit Diffusion Model Quantization via Efficient Selective Finetuning. *arXiv preprint arXiv:2402.03666*.

Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; and Han, S. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, 38087–38099. PMLR.

Yao, Z.; Yazdani Aminabadi, R.; Zhang, M.; Wu, X.; Li, C.; and He, Y. 2022. ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 27168–27183. Curran Associates, Inc.

Yuan, Z.; Shang, Y.; and Dong, Z. 2024. PB-LLM: Partially Binarized Large Language Models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zhao, T.; Fang, T.; Liu, E.; Rui, W.; Soedarmadji, W.; Li, S.; Lin, Z.; Dai, G.; Yan, S.; Yang, H.; et al. 2024. ViDiT-Q: Efficient and Accurate Quantization of Diffusion Transformers for Image and Video Generation. *arXiv preprint arXiv:2406.02540*.

Zhao, T.; Ning, X.; Fang, T.; Liu, E.; Huang, G.; Lin, Z.; Yan, S.; Dai, G.; and Wang, Y. 2025. Mixdq: Memory-efficient few-step text-to-image diffusion models with metric-decoupled mixed precision quantization. In *European Conference on Computer Vision*, 285–302. Springer.