Supplementary Material

A. Visualization of Post-GELU Activations

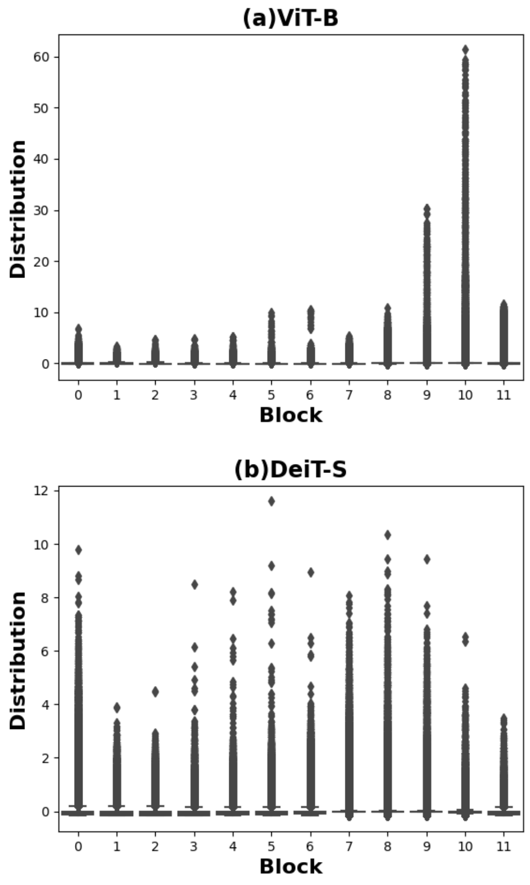

Figure 7. Box plots of block-wise post-GELU activations on (a) ViT-B and (b) DeiT-S.

In Fig. 7, we illustrate the block-wise post-GELU value distributions for ViT-B and DeiT-S through box plots. These visualizations reveal a highly variant distribution across different blocks, with the disparity between the maximum values of the 1^{st} and 10^{th} blocks in ViT-B reaching up to a factor of 10. Additionally, the distribution notably varies between models, as evidenced by the distinct characteristics of the box plots for ViT-B and DeiT-S. This variance underscores the need for data-driven quantization approaches, as static, hand-crafted designs may not sufficiently adapt to such diverse conditions. The limited adaptability of existing methods like those documented in [30,32] accounts for their varied performance on different models, as shown in Tab. 1. Unlike these methods, our proposed AutoScale is designed to dynamically adjust to various data distributions, effectively preserving performance across diverse model architectures.

B. Comparison for Post-GELU Quantization

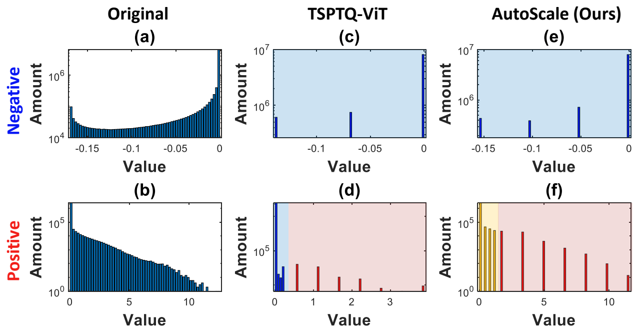

Figure 8. Post-GELU values of 9^{th} blocks of DeiT-S under 4-bit: (a)(b) original, (c)(d) TSPTQ-ViT [32], (e)(f) AutoScale. Please note that the histogram is displayed on a logarithmic scale.

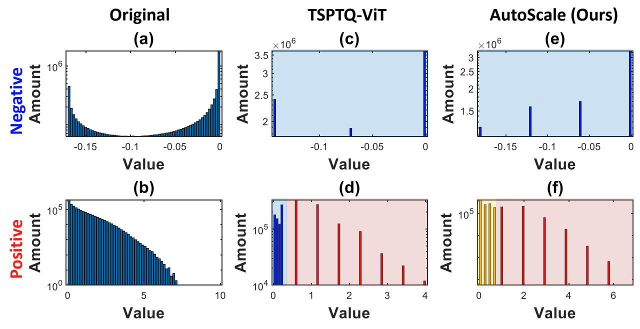

Figure 9. Post-GELU values of 11^{th} blocks of ViT-S under 4-bit: (a)(b) original, (c)(d) TSPTQ-ViT [32], (e)(f) AutoScale. Please note that the histogram is displayed on a logarithmic scale.

To compare with the prior specialized quantizer, TSPTQ-ViT [32], we visualize the distribution of quantized values of DeiT-S (Fig. 8) and ViT-S (Fig. 9). Values in each color region are represented by a specific scaling factor. While TSPTQ-ViT [32] divides values based on magnitude, AutoScale divides them according to distribution, as mentioned in Sec. 3.2. Comparing the negative post-GELU values in Fig. 8(c) and (e), TSPTQ-ViT [32] shows only 3 bins with a scaling factor of 0.0693. In contrast, our method exhibits 4 bins with $s_0 \approx 0.0515$, indicating a higher precision. For the positive post-GELU values shown in Fig. 8(d) and (f), while TSPTQ-ViT [32] allocates 4 bins to small positive values, the representative range is significantly narrower than the complete spectrum of positive values. Moreover, the large scaling factor in [32] is 0.5544, resulting in a maximum quantized value of 3.8808, much lower than the original maximum value of 12.4909. Conversely, AutoScale employs a data-driven strategy to determine the appropriate SF ratios among regions, selecting $m_0 = 3$ and $m_1 = 5$. This results in scaling factors of $s_1 = s_0 \cdot 2^3 \approx 0.4124$ for small positive values and $s_2 = s_0 \cdot 2^5 \approx 1.6494$ for large positive values. This approach expands the range of small positive values and extends the maximum quantized value to 11.5458, notably reducing clamping loss.

Furthermore, in terms of adaptability, our method effectively adjusts the scaling factors when applied to different networks, as shown in Fig. 8(f) and Fig. 9(f). Since the positive values are more concentrated in ViT-S, our AutoScale assigns smaller scaling factors for a more accurate representation. In contrast, TSPTQ-ViT [32] tends to apply similar scaling across different scenarios, as shown in Fig. 8(d) and Fig. 9(d).

To sum up, by assigning region-specific scaling factors tailored to the data distribution, AutoScale effectively manages the non-normal and variant distributions, thus enhancing the precision of the post-quantization values.

C. Ablation Study of Greedy MP Metric

Method	W/A	ViT-L	DeiT-B	Swin-B
FP	32/32	85.84	81.80	85.27
Single-precision	5/5	80.02	58.53	8.74
SQNR only (Eq. (11))	5/5	83.46	75.50	9.82
Greedy MP (Eq. (10))	5/5	**83.61**	**76.44**	**41.69**

Table 8. Ablation study of the selection metric of Greedy MP on ImageNet dataset.

In this section, we evaluate the effectiveness of the selection metric α_{X^l}, as introduced in Sec. 3.3. This metric is designed to balance model performance and compressibility, aiming for an optimal trade-off between these two critical factors. To validate its effectiveness, we conduct an ablation study by using SQNR only as the selection metric:

$$\alpha_{X^l} = SQNR_{b-1}(X_l). \tag{11}$$

The comparison results in Tab. 8 reveal that employing α_{X^l} solely based on SQNR results in diminished accuracy compared to our approach that also considers compressibility (Greedy MP). This outcome highlights the significance of accounting for both performance and compressibility when designing a selection metric. It is noteworthy that using a metric based solely on the number of elements per layer proves ineffective. Given the static nature of element counts, such an approach would bias the quantization process towards consistently targeting the same layer, disregarding the dynamic requirements of different layers. By integrating both performance and compressibility factors into our Greedy MP method, we facilitate a more effective and balanced layer-wise bit-width allocation for quantization.

D. Computation Time Analysis

Tab. 9 shows our calibration and inference times on the NVIDIA GeForce RTX 4090, using the 50,000 ImageNet validation images for inference. While training takes significantly longer, *e.g.*, approximately 90 minutes per epoch for ViT-B, our calibration time remains comparatively short and acceptable. Specifically, Eq. (8) remains manageable since m_1 is small (typically less than 6), and the gradients required for the Hessian matrix are computed once and reused throughout the search.

Method	State	ViT-S	ViT-B	DeiT-S	DeiT-B	Swin-S	Swin-B
TSPTQ-ViT [32]	Calib.	16	23	12	15	29	32
	Infer.	2.4	5.0	2.2	5.0	4.7	6.0
Ours	Calib.	22	40	24	42	63	85
	Infer.	2.4	5.2	2.2	5.2	4.8	6.3

Table 9. Computation time comparison (in minutes).

Importantly, our method doesn't induce much inference overhead. Note that SymAlign doesn't introduce additional storage or inference overhead since μ and ϵ can be integrated into the weights, as detailed in Eq. (4) and Eq. (5). This integration is conducted as an offline preprocessing step before quantization and does not require recomputation during inference. Moreover, we adopt layer-wise quantization, avoiding the computational burden commonly associated with channel-wise methods, further enhancing the efficiency of our model without compromising performance.

E. Comparison with Channel-wise Quantization

Method	W/A	ViT-S	ViT-B	DeiT-S	DeiT-B	Swin-S	Swin-B
FP	32/32	81.39	84.53	79.85	81.80	83.21	85.27
RepQ-ViT [43]	4/4	65.05	68.48	69.03	75.61	79.45	78.32
Ours	4/4	55.88	61.84	68.43	76.14	77.20	76.51

Table 10. Top 1 accuracy on ImageNet dataset.

Tab. 10 presents a comparison with RepQ-ViT [43]. While RepQ-ViT [43] achieves notable performance, it employs channel-wise and asymmetric quantization, which introduces additional storage overhead during inference. In contrast, our approach utilizes layer-wise and symmetric quantization, significantly reducing overhead while maintaining comparable performance, except for vulnerable small-sized models, ViT-S and ViT-B.

F. Limitation of 4-bit Case

Achieving full quantization of ViTs to 4-bit, especially for the Softmax operator, is highly challenging. This is why most prior works just optimize the post-Softmax activations but still retain the Softmax operator in floating-point. In contrast, our approach utilizes the Int-Softmax method [32, 36] for full ViT quantization, making it more hardware-friendly but also increasing quantization difficulty. We found that ViTs tend to collapse if the Int-SoftMax layer operates under 4-bit, leading us to omit results under W4A4 mixed-precision. While keeping Softmax in floating-point, our solution outperforms others under W4A4 single-precision, as shown in Tab. 1.

G. More Experimental Setting

For calibration, we randomly select 32 images from the ImageNet dataset for classification and only 1 image from the COCO dataset for instance segmentation. The candidates of s_0 are 100 equally spaced values by linearly dividing [0, 1.2s], where s is set as $max(|X|)/2^{b-1}$.

In Sec. 4.1.2, we choose TSPTQ-ViT [32] as our single-precision baseline. To maintain model performance under low bit-width quantization, we modify the hyperparameter k in the K-means algorithm of O-2SF [32] from 2 to 4. Except for this modification, we follow the same simulation settings as in [32].

In Sec. 4.2, we quantize the inputs of FC1 while preserving other layers in FP. In comparing ϵ, we find s_0 by MinMax quantization rather than Hessian guided metric to align with [34].

In all experiments, Greedy MP refers to executing Algorithm 1 without Line 1 (SymAlign) and Line 16 (AutoScale), which is used to evaluate the benefits of the greedy MP strategy. As for AMP-ViT, we implement the entire Algorithm 1.

H. SymAlign and Asymmetric Quantization

While asymmetric quantization is a viable approach to handling asymmetry in model parameters, it is generally less favorable for hardware implementation. This is due to the additional online computations required for affine transformations and the memory overhead needed to store zero points. These factors can lead to inefficiencies, particularly in resource-constrained environments.

To address this, our objective is to mitigate the asymmetry issue within the framework of symmetric quantization, which is inherently more hardware-friendly. Our approach involves an offline equivalent transformation applied to linear operators, allowing for direct modification of the weights before inference. This offline adjustment eliminates the need for extra computational overhead during inference, thereby retaining the efficiency advantages of symmetric quantization.

Additionally, since we integrate the bias term into the weights, we apply SymAlign specifically to pairs of consecutive linear operators, as marked in Fig. 1. This approach differs from asymmetric quantization, which is typically applied to the entire model.

I. Discussion of Actual Speedup and Memory Efficiency

The primary goal of quantization is to address the limited I/O bandwidth and memory constraints of resource-constrained edge devices. Therefore, in line with works like NoisyQuant [39] and PD-Quant [31], we evaluate our method by comparing the accuracy under various bit-width against prior studies.

The actual speedup from quantization largely depends on the choice of hardware. Linear operations can often be optimized using single-instruction-multiple-data (SIMD) techniques, whereas nonlinear operations may require specialized accelerators. Since our focus is on reducing the bit-width of weights and activations, hardware-specific acceleration is outside the scope of this paper. However, regarding memory efficiency, our layer-wise, symmetric quantization introduces minimal overhead and achieves 5.3× and 8× memory reductions for the 6-bit and 4-bit models.

J. Interchangeability of Algorithm 1

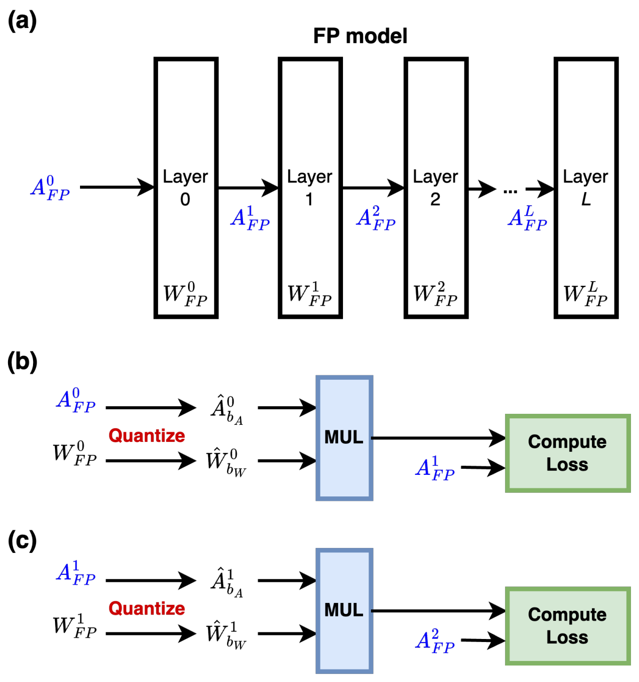

Figure 10. (a) Run the FP model to store the FP inputs (A^l_{FP}) of each layer. (b) Loss calculation for layer 0. (c) Loss calculation for layer 1.

In Algorithm 1, we quantize on a layer-by-layer basis. Specifically, for each layer l, we preserve the raw inputs A^l_{FP} derived from FP models with weights of W^l_{FP}, as shown in Fig. 10(a). Next, we feed the raw input to the target layer, quantize weights and activations under b_W bits and b_A bits to get $\hat{W}^l_{b_W}$ and $\hat{A}^l_{b_A}$, and calculate the loss between the quantized and raw outputs A^{l+1}_{FP}, as shown in Fig. 10(b) and (c). Consequently, from the viewpoint of activations, the only change when swapping steps 2 and 3 is the bit-width of weights (b_w) shifting from the initial 8 bits to a lower bit-width, with almost no impact on the search. We also explored progressive quantization to consider the cumulative quantization errors, resulting in a 2%∼5% drop in accuracy. We attribute this to the overfitting of the calibration dataset, a problem that has also been highlighted in prior work [44].

K. More Comparison with FQ-ViT

In this section, we extend our analysis to include an 8-bit comparison with FQ-ViT [36] under *fully* quantized ViTs, as shown in Tab. 10. It is observed that our method consistently outperforms FQ-ViT [36] across all model types, achieving an accuracy degradation of less than 0.3% compared to the FP models.

Method	W/A	ViT-B	ViT-L	DeiT-S	DeiT-B	Swin-S	Swin-B
FP	32/32	84.53	85.84	79.85	81.80	83.21	85.27
FQ-ViT [36]	8/8	83.31	85.03	79.17	81.20	82.71	82.97
Ours	8/8	**84.24**	**85.79**	**79.84**	**81.80**	**83.10**	**85.10**

Table 11. Top 1 accuracy on ImageNet dataset.

L. Issue of Dataset Discrepancy

Typically, training and test data are assumed to follow similar distributions, making PTQ viable by primarily analyzing the training data alone. However, as the reviewer rightly points out, discrepancies between these datasets can occur in real-world scenarios. To address this issue, it is possible for users to periodically update the parameters to align with the new data distributions [45]. While this approach introduces additional overhead, users can adjust the frequency of these updates to achieve an optimal trade-off between computational cost and model performance.

M. Experiments on Large Language Models

Method	W/A	OPT-1.3B	LLaMa-2-7B
FP	16/16	14.68	5.47
SmoothQuant [33]	8/8	14.79	5.51
Ours	8/8	**14.75**	5.51
SmoothQuant [33]	6/6	17.83	6.49
Ours	6/6	**17.43**	**6.15**

Table 12. Perplexity (↓) on WikiText2 dataset.

Similar to ViTs, transformers used in large language models (LLMs) exhibit issues with activation asymmetry during quantization. Consequently, we adapted the code from [33] and implemented our SymAlign technique to evaluate its effectiveness. The perplexity results (↓) under 8-bit and 6-bit quantization for the WikiText2 [46] dataset are shown in Tab. 12. The results indicate that our method successfully alleviates asymmetry and reduces perplexity, especially in the 6-bit configuration. However, unlike ViTs, which typically use GELU activation functions, these LLMs employ alternative activations. Therefore, we expect that a more tailored analysis accounting for the different properties of data and operators in LLMs will lead to further improvements.

N. More Comparison with OS+

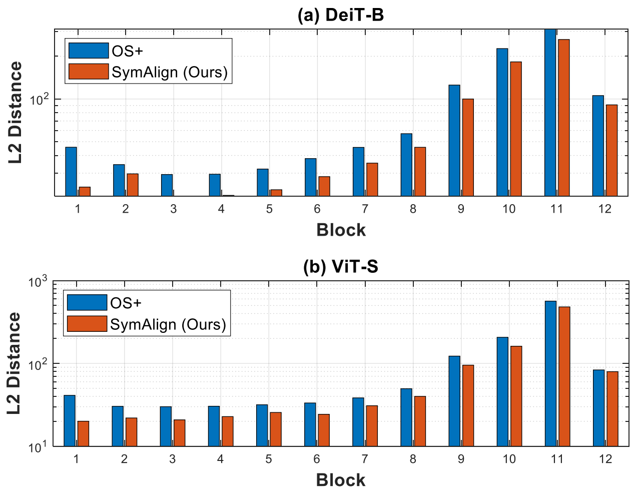

Figure 11. L2 distance between μ and μ_r of (a) DeiT-B and (b) ViT-S.

In Sec. 4.2, we have examined the representativeness of μ by calculating the L2 distance between μ derived from the entire calibration set and each μ_r obtained from individual calibration samples. In this section, we extend the analysis to additional models, as shown in Fig. 11. Consistent with the findings in Fig. 4, our method consistently achieves smaller L2 distances compared to OS+ [34], indicating that SymAlign more effectively captures the general characteristics of the dataset.