Beyond Dynamic Quantization: An Efficient Static Hierarchical Mix-precision Framework for Near-Lossless LLM Compression

Yi Zhang[1,*], **Kai Zhang**[2,1,*], **Zheyang Li**[1], **Wenming Tan**[1,†], **Ye Ren**[1], **Jilin Hu**[2]
[1]Hikvision Research Institute, Hangzhou, China
[2]East China Normal University, Shanghai, China
{zhangyi121, zhangkai23, lizheyang, tanwenming, renye}@hikvision.com
jlhu@dase.ecnu.edu.cn

Abstract

Large language models (LLMs) have achieved overwhelming success but require massive storage and computational resources to support the generative inference. Post-training quantization (PTQ) is a promising approach to reduce memory usage, latency and energy consumption of the deployment of LLMs. However, the presence of outliers makes most existing PTQ methods dedicated to dynamic quantization, which turns out hardware-unfriendly and often leads to large quantization errors in static scenarios. To address the above limitations, we introduce a **S**tatic **H**ierarchical **M**ix-precision **Q**uantization method (SHMQ), which enables near-lossless and hardware-friendly compression of LLMs. Theoretically, our proposed SHMQ quantifies both inter-layer and intra-layer sensitivity through unified derivations involving Hessian. Specifically, SHMQ conducts a systematic precision allocation strategy, which seamlessly integrates coarse-grained inter-layer and fine-grained intra-layer static mix-precision quantization. Furthermore, the permutation procedure, which reorders sensitive channels and insensitive channels that share similar distribution, is leveraged to mitigate static quantization error. Our proposed SHMQ achieves 75.58% on zero-shot reasoning tasks in W4.8A8 Qwen2.5-7B-Instruct, narrowing the accuracy gap to merely 0.13% while yielding averaged 2.86× practical speedup.

1 Introduction

Large language models (LLMs) have demonstrated unprecedented success across various domains, including language understanding, generation, reasoning(Zhang et al., 2022; Touvron et al., 2023; Dubey et al., 2024; Yang et al., 2024), and code generation(Roziere et al., 2023). However, the efficient deployment and generative inference of LLMs require considerable storage and massive computational resources, which becomes an obstacle to the application of LLMs.

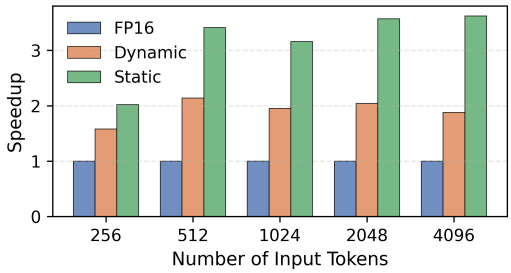


Figure 1: Static quantization exhibits notably higher inference efficiency than dynamic quantization, as its predefined quantization parameters eliminate the real-time computation overhead in dynamic quantization. Evaluation is performed on a mix-precision setup (W4A8 with 20% W8A8), where MatMul is partitioned into W4A8 and W8A8 operations, similar to QUIK.

Post-training quantization (PTQ) serves as a promising technique for tackling computational and memory bottlenecks in LLM inference, which meets the urgent need for efficient deployment on cloud-server and on-device scenarios. However, outliers(Lin et al., 2024) severely damage quantization performance by expanding the quantization range, hindering the efficacy of representations for normal values. Recent research alleviates the effect of outliers by mix-precision quantization. Prior mix-precision methods focus on importance(Ashkboos et al., 2023; Dumitru et al., 2024) or saliency(Huang et al., 2024) metric to identify outliers, and preserves outliers in high precision. They have greatly enhanced the LLMs' capacity under quantization. However, these schemes conduct quantization from a single point view of either inter-layer or intra-layer. The interaction between inter-layer mix-precision quantization and intra-layer mix-precision quantization is not explored. Meanwhile, the vast majority of current PTQ methods

[*]Equal contribution.
[†]Corresponding author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 2573–2587
November 4-9, 2025

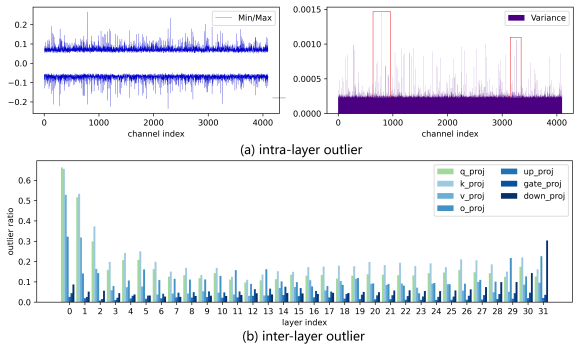


Figure 2: (a) The min/max and variance of outliers vary significantly across different channels. The inter-channel outlier disparities motivates intra-layer mix-precision quantization. (b) The outlier ratio exhibits disparities among different linear layers, which inspires inter-layer mix-precision quantization.

rely heavily on dynamic quantization, which calculates quantization scales runtime, thus enabling better adaptability to distinct distributions. Unfortunately, dynamic quantization leads to low efficiency on GPU and incompatibility on some edge devices, e.g., NOVATEK NT98690 with 6.8TOPS NPU. In contrast, static quantization pre-calculates quantization scales and achieves a substantial reduction in overhead(Chen et al., 2024), as depicted in Figure 1. This leads to an important question: **Can we establish a systematic scheme to handle outliers and improve the performance of efficient static quantization?**

In this paper, we propose a novel quantization method called **S**tatic **H**ierarchical **M**ix-precision **Q**uantization (SHMQ). SHMQ is established based on the insight that outliers exhibit great disparities among different channels and linear layers, as depicted in Figure 2. Theoretically, SHMQ analyzes the perturbation introduced by quantization and derives the quantization sensitivity as a unified metric to assess the sensitivity of inter-layer and intra-layer weights. Concretely, SHMQ establishes a systematic mix-precision quantization scheme via sensitivity metric in two complementary perspectives, i.e., inter-layer and intra-layer mix-precision quantization. Furthermore, the identification and permutation are decoupled and executed sequentially. The permutation procedure guarantees sensitive and insensitive channels that share similar distribution clusters together, mitigating the static quantization error. Experiments demonstrate that, without any fine-tuning or retraining, SHMQ allows the LLMs to achieve practical speedup while maintaining near-lossless performance. The static SHMQ outperforms prior dynamic mix-precision approaches in performance and acceleration, showing great potential for static-only platforms.

2 Related Work

Large language models: Large language models have demonstrated extraordinary performance across domains, including language understanding, generation, reasoning(Touvron et al., 2023; Dubey et al., 2024; Yang et al., 2024), and code generation(Roziere et al., 2023), laying the foundation for artificial general intelligence. However, massive storage and computational resources are required to support the generative inference of LLMs, posing a significant challenge to the deployment of LLMs in resource-constrained scenarios. Prior research mitigates this challenge through quantization(Ma et al., 2024), pruning(Wang et al., 2024; Frantar and Alistarh, 2023), low-rank decomposition(Zhang et al., 2024; Dettmers et al., 2024a) and other techniques(Li et al., 2024).

Post-training quantization for LLMs: Post-training quantization gains remarkable attention for enhancing the inference efficiency of LLMs. However, the presence of outliers in LLMs critically compromises post-training quantization efficacy, driving the development of diverse strategies that eliminate outliers(Lin et al., 2024) and achieve optimal performance-efficiency tradeoffs. The PTQ techniques can be divided into two main categories(Gong et al., 2024; Liu et al., 2024), equivalent transformation(Ma et al., 2024; Shao et al., 2023) and mix-precision quantization(Lee et al., 2023; Zhao et al., 2023). For equivalent transformation, SmoothQuant(Xiao et al., 2023a), OmniQuant(Shao et al., 2023), and OS+(Wei et al., 2023) employ channel-wise scaling strategies to redistribute quantization difficulty between activations and weights. QuaRot(Ashkboos et al., 2024), SpinQuant(Liu et al., 2024), and DuQuant(Lin et al., 2024) harness Hadamard rotation to remove outliers and benefit quantization. As for mix-precision quantization, SpQR(Dettmers et al., 2024b) picks out and stores unstructured outliers in high precision, while the other weights are quantized to much lower bit-width with very small group size. Atom(Zhao et al., 2023) and QUIK(Ashkboos et al., 2023) adopt the diagonal entries of Hessian and l_∞ norm respectively as importance metrics to detect critical channels, followed by quantization with

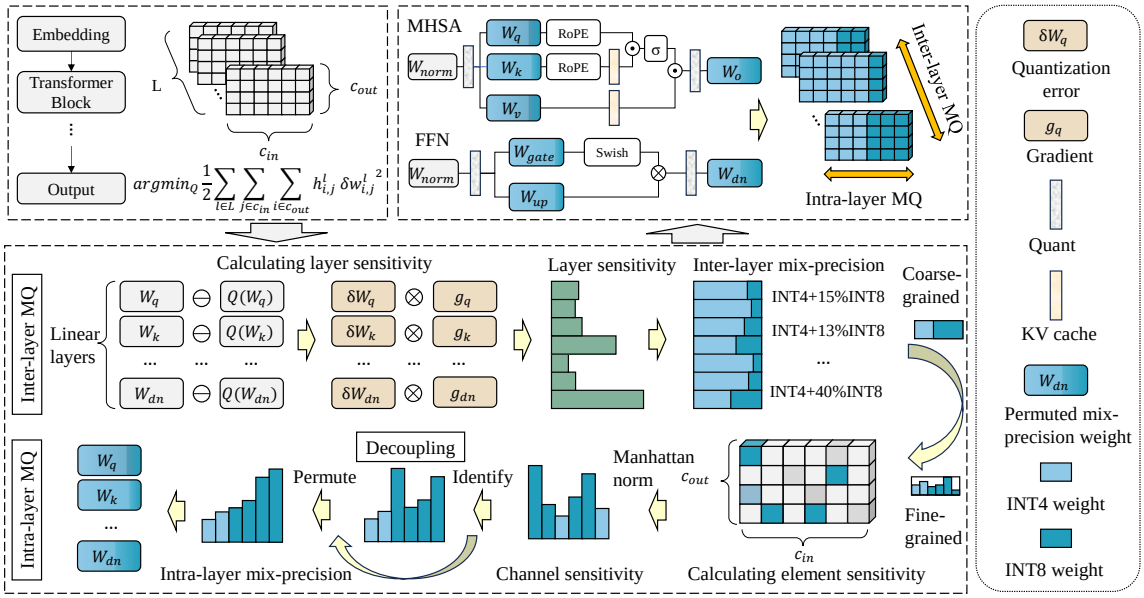


Figure 3: The framework of SHMQ. SHMQ first conducts inter-layer precision allocation based on layer-wise sensitivity to quantization. The intra-layer precision allocation is followed to identify the sensitive channels within the single layer. Eventually, the decoupling of identification and permutation is leveraged to cluster both sensitive and insensitive channels, which share similar distribution, to mitigate the static quantization error.

more bits. SliM-LLM(Huang et al., 2024) utilizes pruning metric from SparseGPT(Frantar and Alistarh, 2023) to quantify the saliency of weights from different groups, thus assigning more bit-widths to important groups. MixLLM(Zheng et al., 2024) calculates the mix-precision metric for output channels in the global view, allocating higher bit-widths to the most salient output features. Recently, another line of work emerges, which addresses outliers through prefixing tokens(Son et al., 2024; Chen et al., 2024) in the KV cache on the basis of attention sinks(Sun et al., 2024; Xiao et al., 2023b). However, the above methods struggles to systematically address outliers in static quantization. To tackle this, we devise SHMQ, a systematic framework rooted in the interaction between inter-layer and intra-layer outliers. To the best of our knowledge, SHMQ represents the pioneering method that achieves near-lossless static quantization.

3 Methods

3.1 Modeling for Optimal Quantization

We theoretically analyze the perturbation induced by quantization in the loss function. The perturbation introduced by quantization can be expressed as:

$$\delta\mathcal{L} = \frac{1}{2}\delta W^\top H \delta W \quad (1)$$

where $\mathcal{L}$ and $\delta\mathcal{L}$ denotes the loss function and the perturbation of loss. $H = \mathbb{E}[\frac{\partial^2}{\partial W^2}\mathcal{L}(W)]$ refers to Hessian. The quantization error of weight matrices can be expressed as $\delta W = W - W_Q$, W and W_Q represent the full precision and quantization versions of weights.

We reconsider the aforementioned formula from an element-wise perspective, the perturbation incurred by every single weight parameter can be formulated as:

$$\delta\mathcal{L} = \frac{1}{2}\sum_{l\in L}\sum_{j\in c_{in}}\sum_{i\in c_{out}} h^l_{i,j}\cdot(w^l_{i,j} - Q(w^l_{i,j}))^2 \quad (2)$$

where $\delta w^l_{i,j} = w^l_{i,j} - Q(w^l_{i,j})$ denotes the weight quantization error of the i_{th} row and j_{th} column from the l_{th} layer. c_{in} and c_{out} are the input and output channel of the weight tensor. $h^l_{i,j} \in H$ represents a single element from Hessian associated with the quantization error $\delta w^l_{i,j}$. Note that the single element of Hessian $h^l_{i,j}$ just represents the scaling factor to each weight quantization error, and requires further derivation for practical calculation.

The optimal comprehensive quantization strategy can be achieved by optimizing the following:

$$Q^l_{i,j}{}^* = \arg\min_{Q^l_{i,j}} \frac{1}{2}\sum_{l\in L}\sum_{j\in c_{in}}\sum_{i\in c_{out}} h^l_{i,j}\cdot \delta w^l_{i,j}{}^2 \quad (3)$$

The overall optimization is challenging due to Hessian complexity and vast search space. We address Hessian complexity with efficient approximation, and tackle the large space via multi-stage optimization. Thus, the objective decomposes into two sub-problems:

$$\begin{cases} Q^{l*} = \arg\min_{Q^l} \frac{1}{2} \sum_{l \in L} h^l \cdot \delta w^l, \\ Q_{i,j}{}^* = \arg\min_{Q_{i,j}} \sum_{j \in c_{in}} \sum_{i \in c_{out}} h^l_{i,j} \cdot \delta w^l_{i,j}{}^2 \\ \quad s.t. \quad Q_{i,j} \in Q^{l*} \end{cases} \tag{4}$$

where $h^l \cdot \delta w^l$ denotes the sum quantization error of l_{th} layer. The overall optimization can be solved through two stages. The first goal of equation 4 is to obtain optimal layer-wise quantization strategy Q^{l*}. Once the layer-wise quantization strategy is fixed, the second goal is to determine the optimal quantization strategy for each individual weight parameter in the same layer.

3.2 The Proposed SHMQ

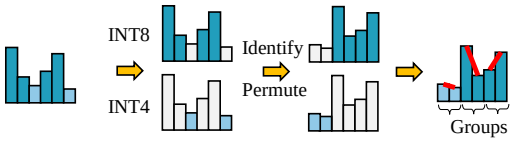
We address two sub-optimization problems in equation 4 via SHMQ. The overall framework of SHMQ is depicted in Figure 3. In order to minimize the perturbation to the final loss $\delta\mathcal{L}$, SHMQ firstly calculates layer sensitivity and determines the optimal inter-layer mix-precision quantization. Secondly, SHMQ calculates the sensitivity of each channel to perform the best intra-layer mix-precision quantization. Following equation 4, we define the quantization sensitivity of the (i_{th}, j_{th}) element from the l_{th} layer as:

$$S^l_{i,j} = \frac{1}{2} h^l_{i,j} \cdot (w^l_{i,j} - Q(w^l_{i,j}))^2 \tag{5}$$

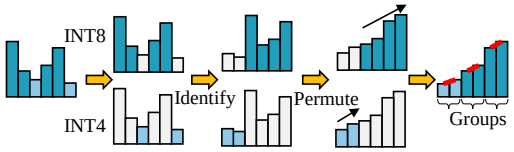
The above metric quantifies the extent to which the weight is sensitive to quantization, serving as a unified metric to guide both inter-layer and intra-layer mix-precision quantization.

3.2.1 Inter-layer Mix-precision Quantization

We present **Inter**-layer **M**ix-precision **Q**uantization to determine the optimal precision allocation to linear layers. We take $\frac{1}{2} \sum_{j \in c_{in}} \sum_{i \in c_{out}} h^l_{i,j} \cdot \delta w^l_{i,j}{}^2$ as sensitivity indicator for each layer and allocate bit-widths to them accordingly. However, it's nearly impossible to construct explicit inter-layer Hessian. We harness the Fisher information matrices(Kim et al., 2023) as an effective alternative for



(a) Coupled Identification and permutation.



(b) Decoupled Identification and permutation.

Figure 4: The comparison of the coupled and decoupled identification and permutation. The red line indicates the variance within the group. The decoupling of the identification and permutation procedure leads to flat distributions, which benefits static quantization.

explicitly computing the layer sensitivity metric. The approximation of Hessian can be expressed as:

$$H \approx F = \frac{1}{|D|} \sum_{D} g g^\top \tag{6}$$

where D denotes the calibration dataset consisting of $|D|$ samples. g represents the gradient vector generated by the backpropagation of the data sample from D. Subsequently, the sensitivity of l_{th} layer can be obtained by:

$$S^l_{InterMQ} = \frac{1}{2} \frac{1}{|D|} \sum_{D} \sum_{i \in c_{out}} (g^\top \delta w^l_{i,:})^2 \tag{7}$$

where $\delta w^l_{i,:}$ represents the quantization error vector of the i_{th} output channel. The detailed proof is provided in Appendix A.

The approximation addresses the challenge of explicitly constructing Hessian in LLMs while also accounting for interactions among different neurons. After that, a novel sensitivity-determined mapping is proposed and leveraged to convert the sensitivity indicator to the bit-widths of each layer. The sensitivity-determined mapping can be written as:

$$U^l = \frac{S^l_{InterMQ} \cdot \sum r_l}{\sum S^l_{InterMQ} \cdot r_l} \cdot (U_t - U_b) + U_b \tag{8}$$

$$r_l = \frac{c^l_{in} \cdot c^l_{out}}{\min_{l \in L} c^l_{in} \cdot c^l_{out}} \tag{9}$$

where $S^l_{InterMQ}$ denotes the sensitivity score of the l_{th} layer, r_l represents the ratio of the l_{th} parameter count to the minimal parameter count of

Table 1: PPL (↓) for LLaMA and Qwen models under W4.8A8 mix-precision quantization. -I denotes Instruct.

Dataset	Type	Method	LLaMA2-7B	LLaMA2-13B	LLaMA3.1-8B	Qwen2.5-1.5B	Qwen2.5-7B-I	Qwen2.5-14B-I
WikiText2	-	FP16	5.47	4.88	6.24	9.26	7.46	5.69
	Dynamic	QUIK	5.70	5.02	6.66	9.95	7.78	6.17
		Atom	5.70	5.02	6.67	9.90	7.80	6.16
	Static	MixLLM	6.04	5.37	7.51	11.51	9.19	7.19
		SHMQ	**5.58**	**4.96**	**6.60**	**9.51**	**7.58**	**6.04**
C4	-	FP16	6.97	6.47	8.96	13.11	10.89	9.38
	Dynamic	QUIK	7.20	6.59	9.53	13.89	11.33	9.75
		Atom	7.21	6.59	9.54	13.80	11.30	9.73
	Static	MixLLM	7.71	7.04	10.75	15.66	12.91	10.58
		SHMQ	**7.12**	**6.59**	**9.46**	**13.48**	**11.06**	**9.68**

all layers. U_t and U_b are the target ratio and base ratio of high precision, respectively. U^l stands for the allocated high precision ratio of l_{th} layer. The sensitivity-determined mapping allocates larger bit-widths to more sensitive layers, in pursuit of optimal utilization of finite precision budget.

3.2.2 Intra-layer Mix-precision Quantization

Once the optimal layer-wise bit-widths are determined, we introduce **Intra**-layer **M**ix-precision **Q**uantization (IntraMQ) to allocate optimal precision to each channel.

Concretely, we proceed with the derivation in equation 5. The quantization error can be easily obtained and the single element of Hessian $h^l_{i,j}$ can be solved following the generalized Optimal Brain Surgeon framework(Frantar et al., 2023). The sensitivity metric for each weight parameter can be expressed as:

$$S^l_{i,j} = \frac{1}{2} \frac{(w^l_{i,j} - Q(w^l_{i,j}))^2}{[(X^l {X^l}^\top + \lambda \,\mathrm{mean}(\mathrm{diag}(X^l {X^l}^\top))^{-1}]_{j,j}} \tag{10}$$

where we adopt $H = XX^\top$ as the alternative way to efficiently compute Hessian. λ is the dampening factor. Our proposed IntraMQ calculates the quantization sensitivity of each parameter using the above formula. The rationale behind using different approximation for Hessian is provided in Appendix A. After that, the Manhattan Norm is leveraged to accumulate the sensitivity of each weight with respect to the input channel. The $S_{IntraMQ}$ can be expressed as:

$$S_{IntraMQ} = ||S^l_{:,j}||_1 = \sum_{i \in c_{out}} |S^l_{i,j}| \tag{11}$$

where $S_{IntraMQ}$ performs as a sensitivity indicator for each input channel and is employed to identify the most sensitive ones accordingly. The identification process can be expressed as:

$$C_{sen} = \mathcal{I}(S_{IntraMQ}, K) \tag{12}$$

where C_{sen} denotes the sensitive channels, $\mathcal{I}$ represents TopK function and K equals to $\lfloor c_{in} \cdot U^l \rceil$.

3.2.3 Decoupled Identification and Permutation

Prior mix-precision methods directly store the sensitive channels in high precision and quantize insensitive channels in low precision. However, we argue that the mix-precision quantization strategy actually couples the identification and permutation procedure, lacking the capability to perceive the distributional properties. The coupling gives rise to fluctuations in the distribution and poses a great challenge to static quantization. The difference of the coupled and decoupled identification and permutation is shown in Figure 4.

We decouple the identification and permutation procedures and execute them sequentially. Firstly, channels are sorted in ascending order of their quantization sensitivity and and partitioned into sensitive C_{sen} and insensitive C_{insen} clusters based on equation 12. Then, we rearrange channels in sensitive cluster C_{sen} based on their magnitude, aiming to minimize group-wise distribution variance. The insensitive cluster C_{insen} are processed in the same way. Eventually, we conduct uniform quantization to the permuted sensitive channels and insensitive ones with different bit-widths, in the pursuit of retaining the maximum capacity of LLMs.

4 Experiments

4.1 Experimental Settings

This paper mainly focuses on W4A8 plus 20% W8A8 (W4.8A8) quantization and conducts ab-

Table 2: Zero-shot QA (↑) results of LLaMA and Qwen models under W4.8A8 mix-precision quantization.

Model	Type	Method	ARC-C	ARC-E	BoolQ	HellaSwag	PIQA	WinoGrande	Avg.
LLaMA2-7B	-	FP16	46.25	74.58	77.77	76.00	79.05	69.22	70.48
	Dynamic	QUIK	**45.65**	**74.41**	77.49	75.10	**79.16**	68.98	70.13
		Atom	45.39	73.99	75.81	75.20	78.62	69.06	69.68
	Static	MixLLM	43.60	71.97	74.22	74.49	77.91	68.43	68.44
		SHMQ	44.45	73.27	**78.50**	**76.13**	78.13	**70.64**	**70.19**
LLaMA2-13B	-	FP16	49.06	77.40	80.61	79.37	80.52	72.30	73.21
	Dynamic	QUIK	48.63	76.77	80.03	78.66	79.71	71.82	72.60
		Atom	**49.32**	76.85	80.28	**78.86**	79.98	71.58	72.81
	Static	MixLLM	48.38	75.38	78.69	77.61	79.16	71.35	71.76
		SHMQ	48.46	**77.15**	**82.23**	77.81	**80.63**	**72.38**	**73.11**
LLaMA3.1-8B	-	FP16	53.50	81.10	82.08	78.93	81.12	73.56	75.05
	Dynamic	QUIK	51.62	79.42	**81.80**	77.78	80.69	73.32	74.11
		Atom	52.30	78.96	81.22	78.28	**81.12**	71.90	73.96
	Static	MixLLM	51.02	76.85	79.42	75.84	79.71	70.01	72.14
		SHMQ	**53.84**	**80.05**	81.56	**78.30**	79.71	**74.19**	**74.61**
Qwen2.5-7B-Instruct	-	FP16	55.03	81.14	86.39	80.50	80.41	70.80	75.71
	Dynamic	QUIK	53.92	78.03	86.18	79.59	78.73	69.38	74.31
		Atom	53.58	76.47	85.93	79.52	77.75	**70.96**	74.04
	Static	MixLLM	51.02	73.32	82.23	77.36	77.64	64.09	70.94
		SHMQ	**55.97**	**80.60**	**86.70**	**79.66**	**80.09**	70.48	**75.58**
Qwen2.5-14B-Instruct	-	FP16	62.20	81.48	88.01	84.33	81.77	76.09	78.98
	Dynamic	QUIK	**60.67**	80.85	**88.41**	83.54	80.36	74.59	78.07
		Atom	60.49	**81.06**	87.65	83.71	80.47	74.43	77.97
	Static	MixLLM	57.59	79.08	85.69	81.31	78.62	71.03	75.55
		SHMQ	60.41	80.35	87.52	**83.92**	**80.63**	**76.32**	**78.19**

lations on the proportion of W8A8. We randomly select 128 samples from WikiText2(Merity et al., 2016) as calibration data, each with 2048 tokens. The base ratio of high precision U_B is set as 12.5% for most LLMs. SHMQ applies per-group symmetric static quantization to weights and activations. The group size is equal to 128. We compare SHMQ with QUIK(Ashkboos et al., 2023), Atom(Zhao et al., 2023) and MixLLM(Zheng et al., 2024). We evaluate SHMQ on LLaMA2(Touvron et al., 2023), LLaMA3.1(Dubey et al., 2024) and Qwen2.5(Yang et al., 2024) series models. We measure the perplexity of these models on the WikiText2(Merity et al., 2016) and C4(Raffel et al., 2020) datasets. Additionally, we assess the zero-shot accuracy on a diverse set of datasets, namely ARC(Clark et al., 2018), BoolQ(Clark et al., 2019), HellaSwag(Zellers et al., 2019), PIQA(Bisk et al., 2020), and WinoGrande(Sakaguchi et al., 2021). Due to page limit, more implementation details, experimental results and visualizations can be found in Appendix A.

4.2 Main Results

As shown in Table 1, our SHMQ demonstrates performance that is comparable to the full precision models. For instance, the quantized LLaMA2-7B using SHMQ achieves 5.58 perplexity, leaving a negligible gap compared to the corresponding full precision model's perplexity of 5.47. The marginal difference validates the effectiveness of SHMQ in preserving LLMs' performance under static quantization. Table 2 exhibits the comparison of SHMQ with other PTQ methods on zero-shot commonsense reasoning tasks. We can observe that the static SHMQ outperforms existing PTQ methods relying on dynamic quantization. In addition, SHMQ obtains a negligible gap compared with FP16 in terms of LLMs' performance. For example, the full-precision and quantized variants of LLaMA2-13B exhibit average zero-shot accuracies of 73.21% and 73.11% respectively, with an extremely slight difference of 0.1%. The negligible gap strongly demonstrate the efficacy of SHMQ. As for LLaMA3.1-8B, previous quantization methods cause approximately 3% decrease in averaged accuracy. Conversely, SHMQ achieves a 74.61% average accuracy across six zero-shot commonsense reasoning tasks, surpassing the second-best approach QUIK by 2.33%. Meanwhile, SHMQ narrows the gap to full-precision to only 0.44% on LLaMA3.1-8B.

Table 3: Layer-wise speedups on a single GPU for different layer sizes. Numbers in brackets indicate the speedup compared to FP16.

Layer Size (c_{in}, c_{out})	**FP16** (ms)	**Dynamic** (ms)	**SHMQ** (ms)
(4096,4096)	0.499	0.469 (**1.06**×)	0.272 (**1.83**×)
(11008,4096)	1.356	0.702 (**1.93**×)	0.504 (**2.69**×)
(14336,4096)	1.758	0.866 (**2.03**×)	0.635 (**2.77**×)
(5120,5120)	0.776	0.535 (**1.45**×)	0.335 (**2.32**×)
(13824,5120)	2.103	0.891 (**2.36**×)	0.656 (**3.21**×)
(8192,8192)	1.953	0.896 (**2.18**×)	0.659 (**2.96**×)
(28672,8192)	6.948	2.195 (**3.17**×)	1.650 (**4.21**×)

The layer-wise speedup ratios of dynamic and static quantization compared to FP16 is shown in Table 3. SHMQ greatly reduces the latency and achieves 1.83× to 4.21× inference speedups across different layer sizes, surpassing dynamic quantization speedups by a significant margin. SHMQ enables both near-lossless compression and efficient deployement of LLMs.

We conduct deployment experiments on a representative edge device: NOVATEK NT98690 with 6.8TOPS NPU. We implement layer-wise inference on the NOVATEK NT98690 with 6.8TOPS NPU and compare the latency between 16-bit integer (Baseline) and our mixed-precision quantized model (W4A8 with 20% W8A8 static quantization) across varying sequence lengths. The layer size is 4096×4096, one of the common layers in LLMs. The results in Table 4 show that SHMQ achieves 2.77× to 2.81× speedup over 16-bit integer, significantly reducing inference latency for large language models on edge devices. These comparisons validate SHMQ's practical efficiency and hardware-friendliness on edge devices.

Table 4: The speedup of SHMQ on edge device NOVATEK NT98690 with 6.8TOPS NPU.

Sequence	**Baseline** (μs)	**SHMQ** (μs)	**Speedup**
512	74522	26917	2.77×
1024	148778	52986	2.81×
2048	297384	106193	2.80×

4.3 Ablation Studies

4.3.1 Module-wise Impact

We validate the effectiveness of each component on Qwen2.5-7B-Instruct. Experimental results in Table 5 manifest that the absence of any one of the three modules will lead to performance degradation. In conclusion, the best quantization performance is acquired by the seamless combination of InterMQ, IntraMQ, and Decoupling components.

Table 5: Ablation study of each component in Qwen2.5-7B-Instruct, evaluated on WikiText2 dataset.

Bits	**InterMQ**	**IntraMQ**	**Decoupling**	**WikiText2** ↓
FP16	-	-	-	7.46
W4.8A8	×	×	×	8.13
	✓	×	×	8.00
	×	✓	×	7.99
	×	✓	✓	7.95
	✓	✓	✓	7.58

4.3.2 Proportion of High Precision

We study the influence of the proportion of W8A8 on quantization performance using LLaMA3.1-8B in Table 6. We vary the proportion of INT8 from 0% to 100%. The performance of the quantized model improves as the bit-width increases. In the setting where all weights are quantized into INT8, i.e., W8A8, the performance gap between the quantized model and the full precision model is minimal. The minimal perplexity gaps on WikiText2 and C4 are 0.08 and 0.09 respectively, which demonstrates lossless capability under static quantization.

Table 6: Ablation study of the proportion of high precision in LLaMA3.1-8B, evaluated on WikiText2 and C4 datasets.

W8A8 Proportion	**Bits**	**WikiText2** ↓	**C4** ↓
FP16	-	6.24	8.96
0%	W4A8	6.70	9.58
10%	W4.4A8	6.65	9.52
20%	W4.8A8	6.60	9.46
50%	W6A8	6.47	9.27
100%	W8A8	6.32	9.05

5 Conclusions

In conclusion, this paper presents SHMQ, an innovative static quantization strategy that effectively addresses the challenge of outliers and enables the near-lossless performance of LLMs. At its core, SHMQ leverages the theoretical quantization sensitivity as a unified metric to conduct both the coarse-grained inter-layer mix-precision quantization and the fine-grained intra-layer mix-precision quantization. Additionally, the decoupling of identification and permutation is proposed to mitigate the static quantization error. SHMQ bridges the accuracy gap between full precision and static quantization of W4.8A8, enhancing the deployment of LLMs in resource-constrained scenarios.

Limitations

In this paper, we propose a novel static mix-precision quantization technique, conducting mix-precision quantization via two complementary perspectives, i.e., inter-layer and intra-layer mix-precision quantization. This method has demonstrated near-lossless performance under efficient static quantization scenarios. The SHMQ can be integrated with other techniques, e.g., LoRA, to further enhance the compressed models' performance. This merits in-depth investigation in our upcoming research endeavors.

References

Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, and Dan Alistarh. 2023. Towards end-to-end 4-bit inference on generative large language models. *arXiv preprint arXiv:2310.09259*.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. 2024. Quarot: Outlier-free 4-bit inference in rotated llms. *arXiv preprint arXiv:2404.00456*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Mengzhao Chen, Yi Liu, Jiahao Wang, Yi Bin, Wenqi Shao, and Ping Luo. 2024. Prefixquant: Eliminating outliers by prefixed tokens for large language models quantization. *arXiv preprint arXiv:2410.05265*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2924–2936.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024a. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Tim Dettmers, Ruslan A. Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. 2024b. SpQR: A sparse-quantized representation for near-lossless LLM weight compression. In *The Twelfth International Conference on Learning Representations*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Razvan-Gabriel Dumitru, Vikas Yadav, Rishabh Maheshwary, Paul-Ioan Clotan, Sathwik Tejaswi Madhusudhan, and Mihai Surdeanu. 2024. Layer-wise quantization: A pragmatic and effective method for quantizing llms beyond integer bit-levels. *Preprint*, arXiv:2406.17415.

Ali Edalati, Alireza Ghaffari, Mahsa Ghazvini Nejad, Lu Hou, Boxing Chen, Masoud Asgharian, and Vahid Partovi Nia. 2025. Oac: Output-adaptive calibration for accurate post-training quantization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(16):16453–16461.

Elias Frantar and Dan Alistarh. 2022. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35:4475–4488.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

Elias Frantar, Sidak Pal Singh, and Dan Alistarh. 2023. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Preprint*, arXiv:2208.11580.

Ruihao Gong, Yifu Ding, Zining Wang, Chengtao Lv, Xingyu Zheng, Jinyang Du, Haotong Qin, Jinyang Guo, Michele Magno, and Xianglong Liu. 2024. A survey of low-bit large language models: Basics, systems, and algorithms. *Preprint*, arXiv:2409.16694.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Wei Huang, Haotong Qin, Yangdong Liu, Yawei Li, Xianglong Liu, Luca Benini, Michele Magno, and Xiaojuan Qi. 2024. Slim-llm: Salience-driven mixed-precision quantization for large language models. *Preprint*, arXiv:2405.14917.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael Mahoney, and Kurt Keutzer. 2023. Squeezellm: Dense-and-sparse quantization. *arXiv*.

Aravindh Krishnamoorthy and Deepak Menon. 2013. Matrix inversion using cholesky decomposition. In *2013 signal processing: Algorithms, architectures, arrangements, and applications (SPA)*, pages 70–72. IEEE.

Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. 2023. OWQ: Lessons learned from activation outliers for weight quantization in large language models. *arXiv preprint arXiv:2306.02272*.

Zheyang Li, Kai Zhang, Qiming Yang, Chaoxiang Lan, Huanlong Zhang, Wenming Tan, Jun Xiao, and Shiliang Pu. 2024. Un-η: An offline adaptive normalization method for deploying transformers. *Knowledge-Based Systems*, 300(000):13.

Haokun Lin, Haobo Xu, Yichen Wu, Jingzhi Cui, Yingtao Zhang, Linzhan Mou, Linqi Song, Zhenan Sun, and Ying Wei. 2024. Duquant: Distributing outliers via dual transformation makes stronger quantized llms. *arXiv preprint arXiv:2406.01721*.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. 2024. Spinquant–llm quantization with learned rotations. *arXiv preprint arXiv:2405.16406*.

Yuexiao Ma, Huixia Li, Xiawu Zheng, Feng Ling, Xuefeng Xiao, Rui Wang, Shilei Wen, Fei Chao, and Rongrong Ji. 2024. Affinequant: Affine transformation quantization for large language models. *arXiv preprint arXiv:2403.12544*.

Donald W Marquardt. 1963. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, and 1 others. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. 2023. Omniquant: Omnidirectionally calibrated quantization for large language models. In *The Twelfth International Conference on Learning Representations*.

Seungwoo Son, Wonpyo Park, Woohyun Han, Kyuyeun Kim, and Jaeho Lee. 2024. Prefixing attention sinks can mitigate activation outliers for large language model quantization. *Preprint*, arXiv:2406.12016.

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. 2024. Massive activations in large language models. *arXiv preprint arXiv:2402.17762*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2024. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. 2023. Outlier suppression+: Accurate quantization of large language models by equivalent and effective shifting and scaling. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1648–1665.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023a. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023b. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

Cheng Zhang, Jianyi Cheng, George A Constantinides, and Yiren Zhao. 2024. Lqer: Low-rank quantization error reconstruction for llms. *arXiv preprint arXiv:2402.02446*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. 2023. Atom: Low-bit quantization for efficient and accurate llm serving. *arXiv preprint arXiv:2310.19102*.

Zhen Zheng, Xiaonan Song, and Chuanjie Liu. 2024. Mixllm: Llm quantization with global mixed-precision between output-features and highly-efficient system design. *Preprint*, arXiv:2412.14590.

A Appendix

We detailed the content of Appendix A here:

Section A.1 gives detailed theoretical derivations of inter-layer sensitivity metric, sensitivity-determined mapping and intra-layer sensitivity metric.

Section A.2 elaborates on the rationale behind using different Hessian approximations for inter-layer and intra-layer mixed precision quantization.

Section A.3 presents more experimental results of our proposed SHMQ, including implementation details, more comparison results, ablation studies, the time of calculating sensitivity metric, and inter-layer mix-precision visualizations.

A.1 Theoretical Derivations

A.1.1 Derivation of Inter-layer Sensitivity Metric

We present theoretical derivation of inter-layer sensitivity metric in this section. Let $w^l_{i,:}$ denote the i_{th} output channel of the l_{th} layer, the inter-layer sensitivity is formulated as:

$$S^l_{InterMQ} = \frac{1}{2}\sum_{i\in c_{out}} {\delta w^l_{i,:}}^\top H \delta w^l_{i,:} \tag{13}$$

$$H \approx F = \frac{1}{|D|}\sum_D gg^\top \tag{14}$$

We leverage Fisher information matrices to approximate Hessian. The above formula can be converted into:

$$\begin{aligned} S^l_{InterMQ} =& \frac{1}{2}\sum_{i\in c_{out}} {\delta w^l_{i,:}}^\top F \delta w^l_{i,:} \\ =& \frac{1}{2}\frac{1}{|D|}\sum_D \sum_{i\in c_{out}} {\delta w^l_{i,:}}^\top gg^\top \delta w^l_{i,:} \\ =& \frac{1}{2}\frac{1}{|D|}\sum_D \sum_{i\in c_{out}} (g^\top \delta w^l_{i,:})^2 \end{aligned} \tag{15}$$

where D denotes the calibration dataset consisting of $|D|$ samples. g represents the gradient vector generated by the backpropagation of the data sample from D.

A.1.2 Derivation of Sensitivity-determined Mapping

This section gives detailed proof that sensitivity-determined mapping guarantees that the allocated bit-widths are equal to the predefined target bit-widths. Assume that we have allocated precision to each layer following:

$$U^l = \frac{S^l_{InterMQ}\cdot\sum r_l}{\sum S^l_{InterMQ}\cdot r_l}\cdot(U_t - U_b) + U_b \tag{16}$$

$$r_l = \frac{c^l_{in}\cdot c^l_{out}}{\min_{l\in L} c^l_{in}\cdot c^l_{out}} \tag{17}$$

where U^l represents the proportion of high precision allocated to l_{th} layer. The overall proportion of high precision U_{all} can be expressed as:

$$\begin{aligned} U_{all} =& \frac{\sum_{l\in L} U^l\cdot c^l_{in}\cdot c^l_{out}}{\sum_{l\in L} c^l_{in}\cdot c^l_{out}} \\ =& \frac{\sum_{l\in L} U^l\cdot r_l}{\sum_{l\in L}\cdot r_l} \\ =& \frac{\sum_{l\in L}(\frac{S^l_{InterMQ}\cdot\sum r_l}{\sum S^l_{InterMQ}\cdot r_l}\cdot(U_t-U_b)+U_b)\cdot r_l}{\sum_{l\in L}\cdot r_l} \\ =& \frac{(U_t-U_b)\frac{\sum S^l_{InterMQ}\cdot r_l\cdot(\sum r_l)}{\sum S^l_{InterMQ}\cdot r_l} + U_b\sum_{l\in L} r_l}{\sum_{l\in L}\cdot r_l} \\ =& \frac{(U_t-U_b)\sum_{l\in L} r_l + U_b\sum_{l\in L} r_l}{\sum_{l\in L}\cdot r_l} \\ =& U_t - U_b + U_b \\ =& U_t \end{aligned} \tag{18}$$

We can conclude that the overall proportion of high precision U_{all} is equal to predefined target U_t.

A.1.3 Derivation of Intra-layer Sensitivity Metric

We proceed on the perturbation caused by quantization as:

$$\delta\mathcal{L} = \frac{1}{2}\delta W^\top H \delta W \tag{19}$$

Assume that we conduct quantization on $w^l_{i,j}$, the quantization can be formulated as:

$$\delta w_{i,j}^l + w_{i,j}^l = Q(w_{i,j}^l) \tag{20}$$

We aim to minimize the perturbation after quantizing $w_{i,j}^l$. This is a convex optimization problem subject to constraints. We can solve this by optimizing the following:

$$\arg\min_{Q_{i,j}^l} \frac{1}{2}\delta W^\top H \delta W + \lambda(e_q^\top \delta w_{i,j}^l + w_{i,j}^l - Q(w_{i,j}^l)) \tag{21}$$

We compute the partial derivatives of the above equation with respect to $\delta w_{i,j}^l$ and λ respectively, and set each derivative to zero.

$$\begin{cases} \frac{1}{2}(H + H^\top)\delta w_{i,j}^l + \lambda e_q^\top = 0 \\ e_q^\top \delta w_{i,j}^l + w_{i,j}^l - Q(w_{i,j}^l) = 0 \end{cases} \tag{22}$$

Then, we can obtain λ and $\delta w_{i,j}^l$. Therefore, the sensitivity of each element is as:

$$S_{i,j}^l = \frac{1}{2}\frac{(w_{i,j}^l - Q(w_{i,j}^l))^2}{[H^{l^{-1}}]_{j,j}} \tag{23}$$

However, the inverse of the Hessian is difficult to compute and quite time-consuming. We refrain from Fisher matrices since computing the inverse of the Fisher matrices presents comparable computational difficulties. We utilize Levenberg-Marquardt approximation(Frantar and Alistarh, 2022; Marquardt, 1963), i.e., $H = XX^\top$, as the alternative way to efficiently compute Hessian. The Cholesky decomposition(Krishnamoorthy and Menon, 2013) is subsequently adopted to compute the inverse of the proxy Hessian. The sensitivity score of $w_{i,j}^l$ is formulated as:

$$S_{i,j}^l = \frac{1}{2}\frac{(w_{i,j}^l - Q(w_{i,j}^l))^2}{[(X^l X^{l^\top} + \lambda\,\mathrm{mean}(\mathrm{diag}(X^l X^{l^\top}))^{-1}]_{j,j}} \tag{24}$$

where we adopt $H = XX^\top$ as the alternative way to efficiently compute Hessian. λ is the dampening factor.

A.2 The Discussion on the approximation of Hessian

We leverage two different Hessian approximations for inter-layer mix-precision quantization and intra-layer mix-precision quantization, i.e., Fisher approximation in interMQ and $H = XX^\top$ in intraMQ. We explain the technical motivations for this dual strategy in this section.

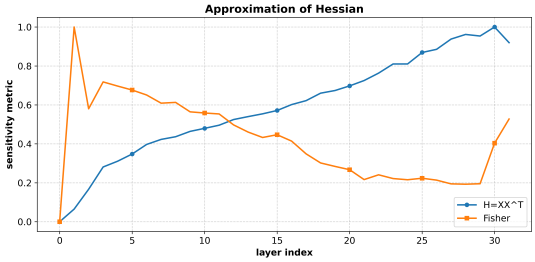


Figure 5: The comparison of $H = XX^\top$ and Fisher approximation of Hessian for interMQ.

Firstly, we demonstrate why the Fisher approximation of the Hessian is unsuitable for intraMQ. The explicit Fisher can be constructed through gradient. However, constructing explicit Fisher consumes massive computional resources and memory. Consequently, prior studies often resort to diagonal approximations of the Fisher to mitigate these challenges(Kim et al., 2023). Yet, the diagonal approximation ignores the interaction among model parameters. We derive as equation 7 to calculate inter-layer sensitivity metric. This equation 7 significantly reduces memory consumption by implicitly constructing the Fisher while accounting for parameter interdependencies. The drawback of this approach is that it can only derive inter-layer sensitivity metric. We still need to explicitly construct the Fisher to compute intra-layer sensitivity metric. Moreover, the inverse of Fisher is necessary since we need to account for the interactions between model parameters. The inverse of Fisher poses significant computational and memory challenges. As a result, we refrain from Fisher for computing intra-layer sensitivity metric.

As mentioned in the main text, we leverage $H = XX^\top$ as the alternative way to efficiently compute Hessian for intra-layer sensitivity metric. This lead to another question: why the $H = XX^\top$ approximation is unsuitable for interMQ. Current research suggests that the hidden states of transformers tend to grow as the depth of layer increases. This trend leads to a bias where deeper layer in the model have higher sensitivity if we adopt $H = XX^\top$ for interMQ. The comparison of $H = XX^\top$ and Fisher for interMQ is shown in Figure 5. The sensitivity metric grows as the the depth of the layer increases under $H = XX^\top$ approximation of Hessian. This bias incorrectly reflects the layer-wise sensitivity, potentially leading to suboptimal precision allocation in quantization. In contrast, the Fisher approximation regard layers

Table 7: Ablation study of different Hessian approximation for inter- and intra-layer mix-precision quantization.

Inter-layer	Intra-layer	Wiki	C4	ARC-C	ARC-E	BoolQ	HellaSwag	PIQA	WinoGrande	Avg.
FP16		5.47	6.97	46.25	74.58	77.77	76.00	79.05	69.22	70.48
Fisher	$XX^\top$	5.58	7.12	44.45	73.27	78.50	76.13	78.13	70.64	70.19
$XX^\top$	$XX^\top$	5.61	7.16	44.37	72.47	78.41	76.00	77.58	69.30	69.69
Fisher	Fisher	5.58	7.11	44.80	72.81	78.44	76.25	77.37	69.93	69.93

at the beginning and end of the LLMs are more sensitive to quantization. Consequently, we leverage Fisher for InterMQ, which is agnostic to the magnitude of the hidden states.

We investigate the impact of different Hessian approximation for inter- and intra-layer mix-precision quantization through ablation studies. The experimental results on Llama-2-7B are presented in the Table 7. The performance of the quantized model degrades when we utilize $H = XX^\top$ for both inter- and intra-layer Hessian approximation. Current research suggests that the hidden states of transformers tend to grow as the layer depth increases. This trend leads to a bias where deeper layers in the model exhibit higher sensitivity when we adopt $H = XX^\top$ for interMQ. The introduced bias accounts for the performance degradation. We compare SHMQ with Fisher information for both inter- and intra-layer Hessian approximation. The performance discrepancies are negligible. However, Fisher for inter- and intra-layer Hessian requires significantly more computational resources and time compared to SHMQ. For instance, on the Llama-2-7B model, Fisher for all consumes an additional 8.9GB of GPU memory and takes 6 minutes longer to complete sensitivity calculations. Note that we implemented Fisher for intra-layer Hessian approximation following the OAC (Edalati et al., 2025) (block-wise diagonal Fisher), since the complete explicit Fisher matrix is computationally infeasible.

A.3 More Experimental Results

A.3.1 Implementation Details

We benchmark SHMQ against state-of-the-art baselines QUIK (Ashkboos et al., 2023), Atom (Zhao et al., 2023), and MixLLM (Zheng et al., 2024). For QUIK and Atom, we adopt group-wise quantization to weights and dynamic quantization to activations. Given that MixLLM's codebase was not publicly available, we reproduced MixLLM by implementing group-wise quantization for weights and static quantization for activations.

SHMQ permutes activation and weight matrices to cluster channels that possess similar distribution, thus mitigating static quantization error. However, the permutation of activation matrices still needs to be performed online, which can be computationally expensive. To address this, we integrate the activation matrix permutation operations with prior operators. For instance, the reordering of the input activation of q_proj/k_proj/v_proj linear layers is fused into the prior normalization layer.

The integration of the permutation operator with the normalization layer necessitates that the allocated high precision ratios and the reordering indices of parallel linear layers are the same. To achieve this, the calculation of both inter-layer quantization sensitivity and intra-layer quantization sensitivity needs to be slightly modified. In inter-layer mix-precision quantization, we calculate the mean sensitivity of parallel linear layers to substitute individual layer sensitivity. Consequently, we guarantee that the allocated high precision ratio of q_proj/k_proj/v_proj linear layers are the same. Similarly, the allocated high precision ratio of up_proj/gate_proj linear layers is consistent. In intra-layer mix-precision quantization, SHMQ first concatenates the element-wise sensitivity matrices of parallel linear layers and applies the Manhattan norm to get the final sensitivity of each input channel. As a result, the permutation procedure of parallel linear layers is the same. Namely, the reordering of q_proj/k_proj/v_proj is consistent, so as to up_proj/gate_proj. The implementation approach effectively minimizes the overhead of the permutation operator and leads to negligible impact on quantized LLMs.

We present a detailed configuration of SHMQ hyperparameters. Firstly, we give a configuration about U_B, which determines the base precision of each linear layer. The base proportion of high precision U_B is set as 12.5% for most LLMs. Secondly, the dampening factor λ is set as 0.1 in equation 10. Finally, we take the product of activations and weights' l_∞ as the permutation metric, which takes

Table 8: Zero-shot QA (↑) results of Qwen2.5-1.5B model under W4.8A8 mix-precision quantization.

Model	Type	Method	ARC-C	ARC-E	BoolQ	HellaSwag	PIQA	WinoGrande	Avg.
Qwen2.5-1.5B	-	FP16	45.05	71.51	72.97	67.73	75.95	63.38	66.10
	Dynamic	QUIK	**44.03**	69.65	71.04	66.30	**75.73**	63.61	65.06
		Atom	43.24	69.39	71.62	66.15	75.36	**63.85**	64.94
	Static	MixLLM	39.68	65.91	57.92	63.23	72.20	59.51	59.74
		SHMQ	43.00	**70.83**	**73.46**	**66.49**	75.63	63.77	**65.53**

Table 9: The performance comparison between rotation-based quantization methods and SHMQ.

Model	Type	Method	Wiki	ARC-C	ARC-E	BoolQ	HellaSwag	PIQA	WinoGrande	Avg.
Llama-2-7B	-	FP16	5.47	46.25	74.58	77.77	76.00	79.05	69.22	70.48
	Dynamic	QuaRot	5.66	45.22	73.74	77.25	74.97	77.97	68.98	69.69
		SpinQuant	5.64	43.17	73.23	76.94	75.12	78.56	69.38	69.40
	Static	SHMQ	**5.58**	44.45	73.27	78.50	76.13	78.13	70.64	**70.19**
Llama2-13B	-	FP16	4.88	49.06	77.40	80.61	79.37	80.52	72.30	73.21
	Dynamic	QuaRot	5.02	50.00	76.52	79.45	78.55	79.87	72.38	72.80
		SpinQuant	5.01	48.63	76.39	81.41	78.43	80.25	72.14	72.88
	Static	SHMQ	**4.96**	48.46	77.15	82.23	77.81	80.63	72.38	**73.11**
Llama3.1-8B	-	FP16	6.24	53.50	81.10	82.08	78.93	81.12	73.56	75.05
	Dynamic	QuaRot	6.70	51.19	77.31	80.24	77.57	80.36	73.32	73.33
		SpinQuant	6.68	50.00	78.11	80.41	80.40	77.64	72.53	73.18
	Static	SHMQ	**6.60**	53.84	80.05	81.56	78.30	79.71	74.19	**74.61**

both activations and weights into consideration.

A.3.2 More Comparison Results

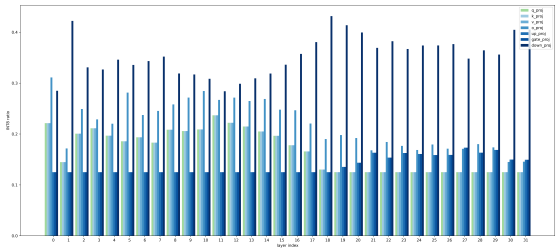
The Comparison on Qwen2.5-1.5B Model. We show zero-shot QA results of Qwen2.5-1.5B model under W4.8A8 mix-precision quantization in Table 8. As shown in Table 8, SHMQ demonstrates consistent improvements across six zero-shot commonsense reasoning tasks. Dynamic QUIK achieves the best averaged accuracy of 65.06% among previous mix-precision quantization methods. However, there is still a significant gap compared to full precision model. Impressively, static SHMQ outperforms dynamic QUIK by 0.47% and narrows the gap relative to FP16 to merely 0.57%. The negligible gap validates the great performance of SHMQ, highlighting its superiority in practical applications.

Evaluations on MMLU. To validate the generality of SHMQ, we also conduct evaluation on Massive Multitask Language Understanding (MMLU)(Hendrycks et al., 2021). The MMLU evaluation results are shown in Table10. The experimental results demonstrates the great potential of proposed SHMQ. For instance, SHMQ achieves 54.66 in LLaMA2-13B on MMLU task, with a mere 0.4% performance drop compared to FP16. As for Qwen2.5-7B-Instruct, FP16 and SHMQ achieve 74.27% and 73.34% on MMLU respectively. This marginal accuracy degradation is remarkable, especially given that static SHMQ reduces computational and memory overhead substantially.

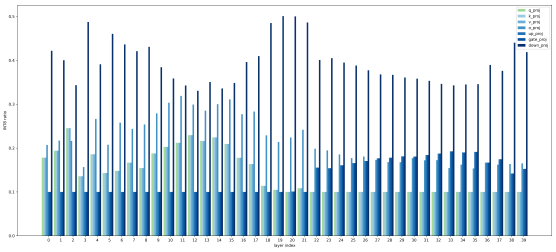
The Comparison with Rotation-based Methods. We also compare SHMQ with QuaRot and SpinQuant, two representative rotation-based approaches. The results in Table 9 demonstrate that our method, SHMQ, achieves superior performance compared to these approaches. For instance, SHMQ demonstrates superior performance over QuaRot and SpinQuant by margins of 1.28% and 1.43%, respectively, when evaluated on the Llama-3.1-8B model for zero-shot common sense reasoning tasks. These improvements maintain their consistency when applied to the Llama-2-7B and Llama-2-13B models.

A.3.3 Ablation on Base Proportion

We also conduct ablations on the base proportion U_B of sensitivity-determined mapping in Table 11. We vary the base proportion U_B from 5% to 15%, and find the lowest perplexity is obtained when U_B equals to 12.5%. The base proportion U_B guarantees that each linear layer possess high precision budget to preserve outliers.



(a) The proportion of high precision allocated to each layer on LLaMA2-7B.



(b) The proportion of high precision allocated to each layer on LLaMA2-13B.

Figure 6: The proportion of high precision allocated to each layer on LLaMA and Qwen series models.

Table 10: MMLU (↑) results of LLaMA and Qwen models under W4.8A8 static quantization.

Model	Method	STEM	humanities	social science	other	ALL
LLaMA2-7B	FP16	36.65	43.32	51.77	52.50	45.86
	SHMQ	36.88	42.32	50.70	51.45	45.09
LLaMA2-13B	FP16	43.57	53.26	63.05	60.76	55.06
	SHMQ	44.00	52.05	62.46	60.98	54.66
LLaMA3.1-8B	FP16	56.06	59.96	76.18	71.62	65.37
	SHMQ	55.20	58.51	74.78	71.28	64.31
Qwen2-7B-Instruct	FP16	70.84	68.23	84.01	76.99	74.27
	SHMQ	69.91	67.08	83.20	76.68	73.34

Table 11: Ablation studies on U_B in Qwen2.5-1.5B, evaluated on C4 dataset.

Base proportion U_B	FP16	5%	10%	12.5%	15%
C4 ↓	13.11	13.51	13.50	13.48	13.52

A.3.4 Ablation on Calibration Dataset

We conduct comprehensive ablation studies on calibration data to validate the robustness of SHMQ. All experiments are performed on Llama-2-7B. First, we evaluate the quantized model's performance using varying numbers of calibration samples (32, 64, 128, and 256) in Table 12. Next, we perform ablation studies on the sequence length of calibration data (512, 1024 and 2048) in Table 13. Finally, we assess performance using different calibration datasets in Table 14: WikiText2, C4, and Pile.

The PPL metric demonstrates negligible fluctuations when tested with different calibration data configurations. The quantized model maintains consistent performance across varying calibration sample sizes, different sequence lengths, and diverse calibration datasets. These findings demonstrate that SHMQ exhibits significant robustness to the calibration data.

Table 12: Ablation study of the calibration samples in Llama-2-7B, evaluated on WikiText2 and C4 datasets.

Samples	WikiText2	C4
32	5.572	7.119
64	5.569	7.116
128	5.581	7.117
256	5.570	7.116

Table 13: Ablation study of the sequence length of calibration data in Llama-2-7B, evaluated on WikiText2 and C4 datasets.

Sequence Length	WikiText2	C4
512	5.573	7.114
1024	5.575	7.114
2048	5.581	7.117

Table 14: Ablation study of the calibration datasets in Llama-2-7B, evaluated on WikiText2 and C4 datasets.

Dataset	WikiText2	C4
WikiText2	5.581	7.117
C4	5.577	7.115
Pile	5.578	7.117

A.3.5 The Time of Calculating Sensitivity Metric

Table 15 shows the time of calculating sensitivity metric in SHMQ. SHMQ identifies and allocates more bit-widths to sensitive layers. Then, sensitive channels are picked out and both sensitive and insensitive channels are reordered to conduct mix-precision quantization. SHMQ demonstrates remarkable efficiency in identifying sensitive channels. Specifically, it requires merely 59 seconds for Qwen2.5-1.5B and 6min56s for Qwen2.5-7B-Instruct on a single GPU, showcasing its rapid and effective operation.

Table 15: The time of calculating sensitive metric in SHMQ on Qwen2.5.

Model	Qwen2.5-1.5B	Qwen2.5-7B-Instruct	Qwen2.5-14B-Instruct
Time	59s	416s	668s

A.3.6 Inter-layer Mix-precision Visualizations

We allocate high precision to each layer based on their sensitivity to quantization. In this section, we present the visualizations of the proportion of high precision allocated to each layer. The visualizations of LLaMA models are depicted from Figure 6a to Figure 6b. The inter-layer mix-precision quantization eliminates the problem of large variations in outlier proportions among different layers.