MLWQ: Efficient Small Language Model Deployment via Multi-Level Weight Quantization

Chun Hu[1], Junhui He[1], Shangyu Wu[1,2*], Yuxin He[1], Chun Jason Xue[2], Qingan Li[1*]

[1]School of Computer Science, Wuhan University,
[2] MBZUAI

Abstract

Small language models (SLMs) are gaining attention for their lower computational and memory needs while maintaining strong performance. However, efficiently deploying SLMs on resource-constrained devices remains a significant challenge. Post-training quantization (PTQ) is a widely used compression technique that reduces memory usage and inference computation, yet existing methods face challenges in inefficient bit-width allocation and insufficient fine-grained quantization adjustments, leading to suboptimal performance, particularly at lower bit-widths. To address these challenges, we propose multi-level weight quantization (MLWQ), which facilitates the efficient deployment of SLMs. Our method enables more effective bit-width allocation by jointly considering inter-layer loss and intra-layer salience. Furthermore, we propose a fine-grained partitioning of intra-layer salience to support the tweaking of quantization parameters within each group. Experimental results indicate that MLWQ achieves competitive performance compared to state-of-the-art methods, providing an effective approach for the efficient deployment of SLMs while maintaining model accuracy.

1 Introduction

Small language models (SLMs) (Zhang et al., 2022; Allal et al., 2024; AI, 2024a,b) offer remarkable capabilities despite their compact size, making it possible to deploy AI technologies in resource-constrained environments beyond traditional cloud-based settings. However, there is still room to improve memory and bandwidth efficiency, which could facilitate easier deployment.

Weight quantization significantly decreases memory usage and bandwidth requirements by reducing the bit-width of model parameters. OWQ (Lee et al., 2024) selects salient channels but leaves them unquantized, resulting in increased overall bit-width. SliM-LLM (Huang et al., 2024b) allocates bit-widths to predefined groups, which may still contain numerous less important weights, leading to inefficient resource usage. AWQ (Lin et al., 2024) reduces the quantization loss of important weights via per-channel scaling, but assigns a uniform bit-width across the layer, leading to inefficient bit-width allocation. Moreover, these methods commonly overlook inter-layer loss relationships, lacking a global perspective on bit-width distribution.

In this paper, we propose a novel quantization method, Multi-Level Weight Quantization (MLWQ), for efficient deployment of SLMs. Our approach jointly considers inter-layer loss and intra-layer weight salience. The approach makes the following three contributions:

- Due to the varying inter-layer loss, we employ a channel-wise distribution loss strategy to determine the quantization bit-width for each layer. Global bit-width allocation establishes a prior foundation for subsequent intra-layer bit-width refinement.
- Under the guidance of the globally assigned bit-width, we further refine the intra-layer bit-width allocation. Specifically, the weights in each layer are partitioned into salient, ordinary, and non-salient categories, with decreasing bit-widths assigned accordingly.
- To further reduce weight quantization errors, we tweak the quantization parameters for each of the three parts in the second contribution.

Experimental results show that our method outperforms current state-of-the-art approaches on OPT (Zhang et al., 2022), Llama-3.2 (AI, 2024a,b), Phi (Li et al., 2023), and SmolLM2 (Allal et al., 2024) models in terms of model perplexity (PPL) and accuracy. The results demonstrate the capability of MLWQ to optimize the deployment of SLMs for real-world applications on edge devices

*Corresponding author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 8079–8089
November 4-9, 2025

with limited computational resources. The code is publicly available [1].

2 Background

This section provides an overview of the main concepts in quantization techniques.

Fundamentals of quantization: Quantization converts a floating-point number into an integer with a lower bit-width. Uniform quantization partitions a continuous range into equal intervals, mapping all values within each interval to the same integer. The corresponding equation for this process is as follows:

$$W_q = \text{clamp}\left(\left\lfloor \frac{W}{S} \right\rfloor + Z, 0, 2^N - 1\right) \quad (1)$$

where W represents the floating-point number to be quantized, S is the floating-point scaling factor, and Z is the integer zero point. $\lfloor\ \rfloor$ is a rounding operation, and clamp$(\cdot)$ refers to the truncation function.

Weight salience: The Hessian matrix is widely used to assess the relative importance of model parameters (Dettmers et al., 2023; Huang et al., 2024a,b). Leveraging this property, we utilize the Hessian matrix to evaluate the salience of each individual parameters. Specifically, the salience of each weight is computed using the following equation:

$$\delta_{i,k} = \frac{w_{i,k}^2}{([H^{-1}]_{k,k})^2} \quad (2)$$

where H denotes the Hessian matrix, $w_{i,k}$ denotes the value of the weight parameter.

Layer loss: The average of the sum of channel-wise losses is taken as the loss for the layer, as defined in Equation (3) (Li et al., 2024). In this equation, C denotes the number of activation channels, while μ and σ denote the mean and variance of each activation channel within the layer, respectively. The subscripts f and q correspond to the float and quantized models.

$$Dist = \frac{1}{C}\sum_{c=1}^{C}\left(\left\|\mu_f^c - \mu_q^c\right\|_2 + \left\|\left(\sigma_f^c\right)^2 - \left(\sigma_q^c\right)^2\right\|_2\right) \quad (3)$$

3 Related Work

Existing quantization methods can be classified into *Quantization-Aware Training* (QAT) and *Post-Training Quantization* (PTQ). QAT (Kim et al., 2022; Liu et al., 2023; Neill and Dutta, 2023) combines the quantization process with the training process, allowing the model to account for the effects of quantization during training. PTQ is the process of quantizing a model after it has been trained.

weight-only quantization: OPTQ (Frantar et al., 2022), which primarily focuses on parallel quantization of all rows of weights and utilizes lazy batch updates to achieve a higher compute-to-memory ratio. Combining LDLQ and incoherent processing, QuIP(Chee et al., 2023) is the first large language model quantization method that achieves feasible results even with 2-bit weight quantization. SqueezeLLM (Kim et al., 2024) is a post-training quantization framework that enables lossless ultra-low precision compression and improves performance under memory constraints by using sensitivity-based non-uniform quantization and Dense-and-Sparse decomposition. OWQ (Lee et al., 2024) improves upon OPTQ by using mixed-precision quantization to reduce precision loss from activation outliers, but this comes at the cost of increased overall quantization bit-width due to retaining higher bit representations for important weight channels. Based on the observation that weights in large language models do not have the same level of salience, AWQ (Lin et al., 2024) performs per-channel scaling to reduce the quantization loss of salient weights. However, it overlooks inter-layer relationships, which may lead to inefficient use of bit-width resources. Norm Tweaking (Li et al., 2024) corrects the distribution of quantized activations to match their floating-point counterparts, which can easily restore the accuracy of LLMs. SliM-LLM (Huang et al., 2024b) allocates bit-widths based on predefined groups, but even within crucial groups, many less salient weights may remain, leading to suboptimal utilization of the available bit-widths. In addition, traditional DNN quantization methods, such as BitsEnsemble (Cui et al., 2022), employ a differentiable and parallelizable bit-sharing scheme to significantly reduce storage overhead while preserving member performance and inference efficiency.

While the aforementioned methods have achieved remarkable results in the quantization of large models, they lack a comprehensive consideration of the inter-layer loss relationships and the intra-layer weight salience distribution. In comparison, this paper proposes to reduce weight quantization error further by jointly considering both inter-layer loss characteristics and the intra-layer

[1] https://github.com/hudevictor/mlwq

distribution of weight salience. To achieve this goal, we interpret the inter-layer bit-width as a resource allocation indicator, where a higher bit-width suggests that the corresponding layer is more sensitive to quantization and thus requires greater representational capacity to prevent a significant increase in block-level quantization loss. Specifically, channels are categorized into important, moderate, and less important groups based on their salience, and are assigned bit-widths accordingly. This approach ensures global resource efficiency while preserving precision where it matters most. Additionally, to further reduce quantization error, we apply fine-grained calibration of quantization parameters within each group.

4 Motivation

In this section, we first investigate the inter-layer loss characteristics, focusing on the variations in loss across different layers. Next, we explore the intra-layer salience distribution, aiming to understand the weight importance within individual layers. This dual analysis will guide the more effective allocation of bit-widths and the efficient tweaking of corresponding quantization parameters.

4.1 Inter-Layer Loss Characteristics

The performance of each layer during the quantization process varies, particularly with noticeable differences in loss across different bit-widths. To minimize the model's storage overhead, it is crucial to understand how each layer performs under different bit-width configurations.

Observation 1: We observe that under the same bit-width, the loss varies across layers, while within the same layer, it fluctuates with different bit-widths. We begin by conducting an empirical analysis of the layer-wise loss distribution within a block, utilizing Equation (3). Figure 1 shows the results for block 3 of the opt-350m model. Regardless of whether 2-bit or 4-bit quantization is applied, the *self.attn.out_proj* layer exhibits the lowest loss, while the *fc1* layer incurs the highest loss. Moreover, increasing the quantization bit-width for a given layer leads to a reduction in the corresponding loss. **This motivates us to allocate the bit-widths for each layer more appropriately based on the corresponding loss.** For example, a higher bit-width can be allocated to *fc1* due to its higher sensitivity to quantization, while a lower bit-width can be allocated to *self.attn.out_proj*, which exhibits lower sensitivity.

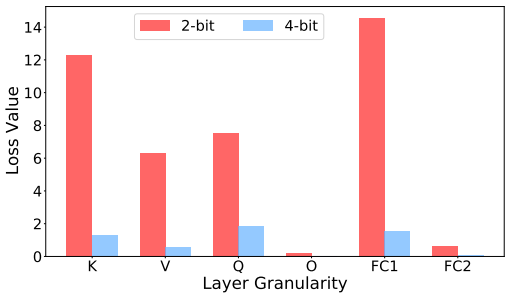
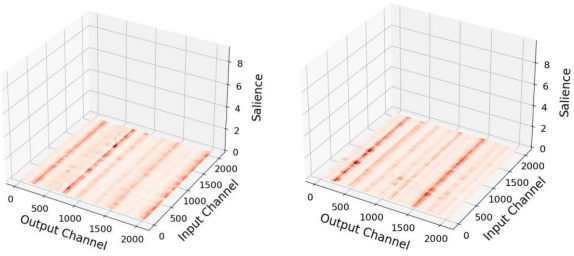


Figure 1: Layer loss in a block. Through performing 2-bit and 4-bit quantization separately, the corresponding loss for each layer was calculated.

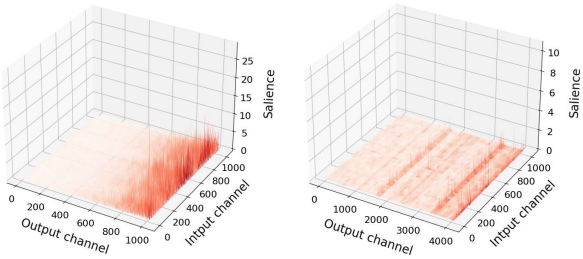
4.2 Intra-Layer Salience Distribution

Analyzing the distribution of weight importance within each layer enables more precise and adaptive bit-width allocation. Rather than classifying weights as simply salient or non-salient, the salience of weights often follows a distribution with multiple levels of importance.



$6\_self\_attn.o\_proj$ $10\_self\_attn.o\_proj$

Figure 2: Salience in Llama-3.2-1B. The salience distributions of weights for $6\_self\_attn.o\_proj$ layer and $10\_self\_attn.o\_proj$ layer are provided according to their level of salience.



$1\_self\_attn.v\_proj$ 14_fc2

Figure 3: Salience in OPT-350M. The salience distributions of weights for $1\_self\_attn.v\_proj$ layer and 14_fc2 layer are provided according to their level of salience.

Observation 2: We observe that weights can-

not be simply categorized as salience or non-salience, requiring a more nuanced quantization approach. We perform an empirical analysis of the weight salience distribution. Figure 2 shows results of the $6\_self\_attn.o\_proj$ layer and $10\_self\_attn.o\_proj$ layer in Llama-3.2-1B, indicating that certain channels exhibit significantly higher salience, while some channels are notably less salient, and others have a certain degree of salience, though not as prominent. In Figure 3, a similar distribution is also observed in the $1\_self\_attn.v\_proj$ layer and 14_fc2 layer of OPT-350M. This distribution of weight salience highlights the need for a more nuanced quantization approach, where bit-width allocation is adapted to the varying grades of salience. **Based on Observation 2, this motivates us to introduce an intermediate level of salience to smooth the gap between salient and less salient weights.**

To evaluate the impact of category granularity on quantization, we vary the number of categories with bit-width=2, 3, 4, 5, 6. The corresponding bit-width allocations for each setting are summarized in Table 1.

CATEGORY	2	3	4	5
BIT	(2,4)	(2,3,4)	(2,3,4,5)	(2,3,4,5,6)

Table 1: Comparison of different categories

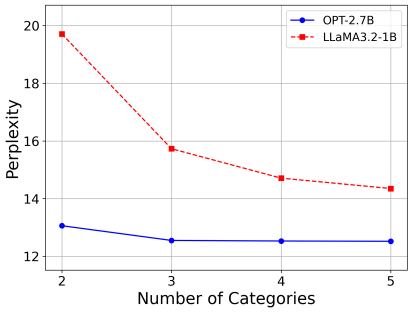


Figure 4: Perplexity results of OPT-2.7B and Llama-3.2-1B on WikiText2 across different numbers of bit-width categories.

Figure 4 illustrates the perplexity of OPT-2.7B and Llama-3.2-1B on WikiText2 under varying numbers of weight categories(n) used for quantization. As n increases from 2 to 5, the perplexity consistently decreases, and the gain from increasing the categories from n=2 to n=3 is greater than that from n=3 to n=4. To balance effectiveness and efficiency, we use n=3 in our experiments, though higher values are recommended to further reduce perplexity.

5 MLWQ

Our approach jointly considers both inter-layer loss characteristics and the intra-layer distribution of weight salience. MLWQ consists of three stages. Figure 5 illustrates the first two stages, while the third stage will be introduced in Section 5.3.

(1) Bit-width Preallocation based on Layer Loss (BPLL): employs a channel-wise distribution loss strategy to determine the optimal quantization bit-width for each layer, offering a global consideration of the relationship between layers.

(2) Mixed-precision Quantization Based on Salience Awareness (MQSA): allocates bit-widths according to the salience of group weights within each layer. Moreover, layers assigned higher bit-widths are granted enhanced channel protection by allocating a greater proportion of high-precision channels.

(3) Tweaking Quantization Parameters (TQP): involves adjusting the quantization parameters of the three groups of weights, categorized based on salience, to further minimize quantization errors.

5.1 Bit-width Preallocation based on Layer Loss

Based on Observation 1, we propose the BPLL strategy, which initially allocates bit-widths to each layer based on its associated loss. After calculating the loss for each layer, we use Equation (4) to find the optimal bit-width combination:

$$\begin{aligned} &\text{Objective:} \quad \underset{B}{\operatorname{argmin}} \sum_{i=1}^{n} \mathit{Dist}(W_i X, W_{b_i} X) \\ &\text{Constraint:} \quad \sum_{i=1}^{n} n_i \cdot b_i < \mathit{Total\_bits}, \end{aligned} \tag{4}$$

where W_i denote the weights of the i-th layer, b_i represent the bit-width allocated to the i-th layer, and W_{b_i} denote the quantized weights of the i-th layer under the bit-width b_i in objective equation. In constrain equation, n_i refers to number of weight in a layer, and $Total\_bits$ represents the target compression's overall bit-width. Our objective is to find a bit-width set B, where each bit-width is allocated to a layer, such that the sum of the *Dist* value across all layers within a block is minimized.

Additionally, We solve this constrained optimization problem using enumeration. For example, if

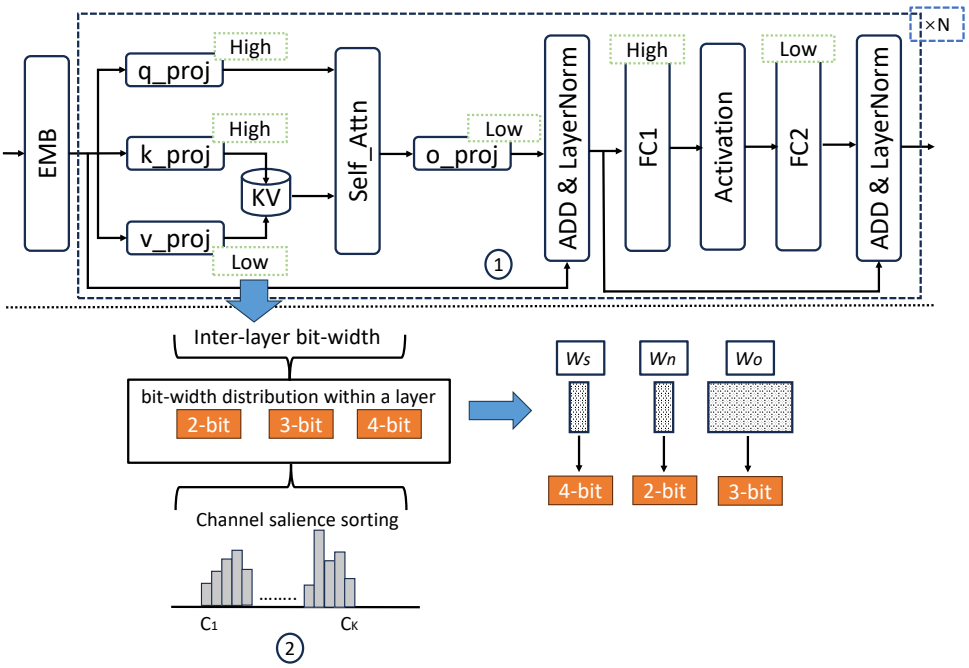


Figure 5: This framework diagram illustrates the first two stages of MLWQ: Bit-width Preallocation based on Layer Loss (BPLL) and Mixed-precision Quantization based on Salience Awareness (MQSA).

the available bit-widths are 2-bit and 4-bit, then each layer within a block has two choices. As a result, the total number of bit-width configurations for a block with l layers is 2^l.

In this strategy, each layer's bit-width is determined not in isolation, but in coordination with others, reflecting its relative sensitivity. More sensitive layers are assigned higher precision to mitigate their contribution to overall quantization loss, while less sensitive ones are allocated fewer bits. This inter-layer allocation further serves as a structural prior for the subsequent salience-aware intra-layer quantization.

5.2 Mixed-precision Quantization based on Salience Awareness

Previous methods (Lee et al., 2024; Dettmers et al., 2022) that use only two categories, salient and non-salient, may limit the potential for compression. While non-salience weights can be quantized to lower bit-widths to save storage space, an overly coarse division may lead to excessive compression of salient weights, resulting in significant loss of precision and adversely affecting the model's final performance.

To smooth the distribution of weights with different levels of salience, based on observation 2, we divide the weights into three parts: the most salient weights, which are allocated higher bit-widths; the least salient weights, which are allocated the lowest bit-widths; and the remaining weights, which are allocated intermediate bit-widths. This fine-grained bit-width allocation ensures that the quantization process preserves the essential information in high-impact weights, thereby mitigating potential degradation in accuracy. The Equation (5) represents the importance of k-th weight channel:

$$\delta_k = \sum_{i=1}^{m} \frac{w_{i,k}^2}{([H^{-1}]_{k,k})^2} \tag{5}$$

where the number of rows in the weight matrix is represented by m, $[H^{-1}]_{k,k}$ refers to the k-th diagonal entry of the inverse Hessian. Furthermore, H^{-1} can be efficiently calculated by Cholesky decomposition.

Based on the salient of each channel, we select the top-H and bottom-L channels, corresponding to the most and least salient weights, respectively. The values of H and L are influenced by the inter-layer bit-width allocation and can be dynamically adjusted. This enables layers with higher bit-widths, which are typically more sensitive to quantization, to retain a larger number of important channels. Moreover, such intra-layer precision assignment is inherently dependent on the inter-layer bit-width allocation established in the pre-

vious stage and further influences the distribution and calibration of scaling factors in the subsequent step.

5.3 Tweaking Quantization Parameters for Grouped Weights

To further mitigate quantization errors, building on observation 2 and the weight partitioning presented in the previous section, we individually adjust the quantization parameters of the three partitioned groups. This strategy necessitates only the adjustment of clipping strengths to identify an optimal clipping threshold, thereby alleviating the difficulty of the optimization. When clipped with the optimal threshold, the original weights become easier to quantize. Unlike existing methods, such as learnable weight clipping (LWC) (Shao et al., 2023), which apply a uniform set of clipping parameters to the entire block, the proposed TQP identifies group-specific optimal clipping thresholds tailored to the unique characteristics of each weight group.

TQP employs learnable clipping strength parameters, γ and β, in the quantizer, adjusting the scaling factors and zero-point:

$$\begin{aligned} \text{where} \quad s &= \frac{\gamma(\max(\mathbf{W}) - \min(\mathbf{W}))}{2^N - 1}, \\ z &= -\left\lfloor \frac{\beta \min(\mathbf{W})}{s} \right\rfloor \end{aligned} \tag{6}$$

Based on the previous partitioning, each group underwent targeted adjustments of its respective quantization parameters. As illustrated in Equation (7), where L represents the l_2 loss, w_s, w_n, w_o correspond to the salient, the non-salient, and the ordinary weight, respectively. The quantized counterparts are denoted as $\hat{\boldsymbol{w}}_{sq}$, $\hat{\boldsymbol{w}}_{nq}$, $\hat{\boldsymbol{w}}_{oq}$.

In this way, the adjustment of quantization parameters is not merely arbitrary but is tailored to the characteristics of the weight distribution within each group, effectively reducing quantization errors.

$$\begin{aligned} &\underset{\gamma_1,\, \beta_1}{\operatorname{argmin}} \mathcal{L}_s(\boldsymbol{w}_s, \operatorname{dequant}(\hat{\boldsymbol{w}}_{sq})) \\ &\underset{\gamma_2,\, \beta_2}{\operatorname{argmin}} \mathcal{L}_n(\boldsymbol{w}_n, \operatorname{dequant}(\hat{\boldsymbol{w}}_{nq})) \\ &\underset{\gamma_3,\, \beta_3}{\operatorname{argmin}} \mathcal{L}_o(\boldsymbol{w}_o, \operatorname{dequant}(\hat{\boldsymbol{w}}_{oq})) \end{aligned} \tag{7}$$

Although the adjustment of scaling factors is primarily aimed at reducing local quantization errors, its effectiveness is significantly influenced by the decisions made in the previous two stages. For instance, lower bit-widths tend to increase quantization errors, while salience-based grouping can lead to uneven distributions of scaling factors. Therefore, this step serves not only as a standalone precision calibration mechanism, but also as an adaptive feedback and correction process in response to earlier bit-width and salience assignments.

5.4 Synergistic Optimization of Inter-layer loss and Intra-layer salience

MLWQ begins with inter-layer bit-width allocation, which assigns precision budgets to each layer based on their relative the corresponding loss to quantization. This global allocation influences the intra-layer optimization process, ensuring that layers with higher sensitivity prioritize the retention of high-bit channels. Within each layer, intra-layer bit-width assignment further distributes the allocated precision among weight channels according to their salience. By allocating higher precision to more important weights, this step reduces quantization error where it matters most and avoids wasting bits on unimportant parameters. Following this, scaling factor refinement is applied to recalibrate the quantization parameters in light of the uneven precision distribution. This co-designed pipeline enables a globally informed, locally adaptive quantization strategy, offering advantages that cannot be achieved by applying these steps in isolation.

6 Experiments

6.1 Experimental Setup

We evaluated the quantization effectiveness of the MLWQ to validate the performance of our proposed approach. For calibration, 128 samples are randomly drawn from WikiText2, with each sample comprising 2048 tokens. The Experiments are performed on an RTX 4090 GPU with 24 GB of memory.

Models. To comprehensively demonstrate the performance advantages of low-bit quantization in MLWQ, we tested several SLMs, specifically including OPT families (Zhang et al., 2022), Phi familes (Li et al., 2023) SmolLM2 families (Allal et al., 2024), and Llama-3.2 families (AI, 2024a,b).

Baselines. Our primary baseline for comparison is OPTQ (Frantar et al., 2022), RTN, OWQ (3.01-bit confguration) (Lee et al., 2024), AWQ (Lin et al., 2024), OMNIQUANT (Shao et al., 2023), and SliM-LLM (Huang et al., 2024b).

Evaluation Metrics. To assess the performance of

MLWQ, we concentrate on two key metrics: perplexity and zero-shot performance. Perplexity is employed as the evaluation metric in this paper, as it is widely acknowledged for its stability in assessing language generation performance. The experiments are conducted using the WikiText2 datasets. Additionally, in order to evaluate the practical applicability of the quantized LLMs, we assess their performance on zero-shot benchmarks, including PIQA (Bisk et al., 2020), ARC (Clark et al., 2018), WINOGRANDE (Sakaguchi et al., 2020), MATHQA (Amini et al., 2019), LOGIQA (Liu et al., 2020) and ANLI_R2 (Nie et al., 2019).

6.2 Perplexity Results

The results presented in table 2 and table 3 demonstrate the competitive performance of MLWQ across all tested model and quantization bit-widths. The results clearly demonstrate that MLWQ consistently preserves model quality across various architectures and bit-width settings. In the 4-bit configuration, MLWQ closely matches full-precision performance while outperforming all competing methods, indicating its strong capability in minimizing quantization-induced degradation. Under more aggressive 3-bit quantization, MLWQ maintains competitive performance across models of different sizes, showcasing its robustness and adaptability.

OPT	BIT	125M	350M	1.3B	2.7B
-	16	27.65	22.01	14.62	12.47
SliM-LLM	4	30.46	23.99	15.15	12.68
OWQ	4.01	31.33	23.53	14.91	12.39
AWQ	4	29.14	24.98	14.94	12.28
RTN	4	37.28	25.93	48.45	16.92
OPTQ	4	31.31	23.82	15.72	13.03
MLWQ	3.99	**27.93**	**23.37**	**14.85**	**11.92**
SliM-LLM	3	43.41	29.71	15.98	13.67
OWQ	3.01	38.96	27.13	15.95	13.27
AWQ	3	35.71	26.36	16.31	13.28
RTN	3	1.2e3	64.61	1.3e4	1.5e4
OPTQ	3	53.37	34.37	21.46	16.12
OmniQuant	3	73.31	59.45	22.20	22.48
MLWQ	2.99	**28.98**	**23.81**	**15.24**	**12.55**

Table 2: Perplexity of OPT under 3-bit and 4-bit quantization.

6.3 Zero-shot Performance Results

Table 4 compares the accuracies of different quantization methods on downstream tasks. The experimental results demonstrate that MLWQ achieves competitive performance compared to other quantization methods (SliM-LLM, OPTQ, and OWQ) across a diverse set of models and tasks. For OPT-125M on ARC-Easy, MLWQ achieves 40.02, outperforming SliM-LLM (36.27) by 10.3%, OPTQ (38.09) by 5.1%, and OWQ (37.46) by 6.8%, demonstrating robust preservation of commonsense knowledge. When evaluated on PIQA with Llama-3.2-1B, MLWQ scores 67.13, exceeding SliM-LLM (66.43) by 1.1%, OPTQ (56.15) by 19.6%, and OWQ (66.46) by 1.0%, indicating better retention of physical world knowledge. The experimental results for SmolLM2 and Phi family are provided in Appendix A.

6.4 Ablation Study

To assess the contribution of each component, we performed a single-variable elimination study by removing one component at a time while retaining the others. Specifically, the perplexity results for BPLL+MQSA, BPLL+TQP, and MQSA+TQP represent the performance after eliminating TQP, MQSA, and BPLL, respectively. Among these, the elimination of TQP (BPLL+MQSA) causes the most significant increase in perplexity, indicating that TQP has the largest contribution to the overall performance. Figure 6 illustrates the results of an ablation study conducted to evaluate the impact of individual components within the proposed BPLL+MQSA+TQP framework on Wikitext2 perplexity across models of varying sizes (0.125B, 0.35B, 1.3B and 2.7B parameters). The full method (BPLL+MQSA+TQP) consistently achieves the lowest perplexity, confirming its effectiveness in reducing perplexity by leveraging all three components.

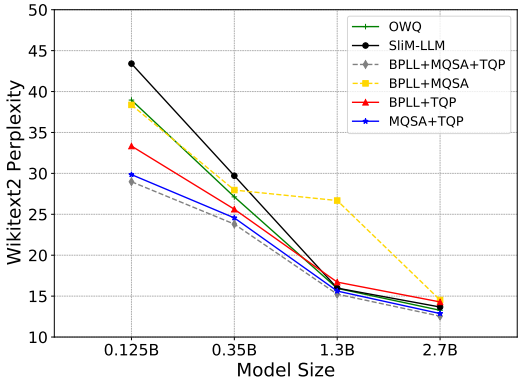


Figure 6: Ablation results on OPT models. The perplexity results for BPLL+MQSA, BPLL+TQP, and MQSA+TQP reflect the performance after eliminating TQP, MQSA, and BPLL, respectively, followed by 3-bit quantization.

PPL ↓		SmolLM2			Llama-3.2		Phi	
Model	BIT	135M	360M	1.7B	1B	3B	1.3B	2.7B
-	16	15.8	11.62	8.24	9.75	7.81	20.81	9.45
SLIM-LLM	3	64.73	18.33	11.47	25.01	21.26	23.52	11.55
OWQ	3.01	45.26	23.85	24.99	15.76	10.24	24.06	11.48
AWQ	3	31.27	20.53	13.15	18.03	10.31	23.63	12.32
RTN	3	8.1E3	1.3E3	6.7E4	2.6E3	477.65	49.06	27.97
OPTQ	3	222.7	51.29	334.45	55.72	7E3	26.01	14.94
MLWQ	2.99	**29.78**	**18.13**	**11.35**	**15.73**	**10.23**	**21.03**	**11.35**

Table 3: Perplexity of SmolLM2, Llama-3.2 and Phi on WikiText2 under 3-bit quantization

Model / Acc↑	BIT	Method	PIQA	ARC-Easy	Winogrande	MathQA	LogiQA	ANLI_R2	Avg
	16	-	63.01	43.51	50.19	22.11	22.88	37.58	39.88
	3	SLIM-LLM	59.03	36.27	51.22	21.57	21.96	33.75	37.31
OPT-125M	3	OPTQ	59.19	38.09	50.28	**21.84**	21.66	33.71	37.46
	3	OWQ	60.28	37.46	50.83	21.47	20.43	32.51	37.16
	2.99	MLWQ	**60.32**	**40.02**	**51.53**	21.61	**22.56**	**33.98**	**38.33**
	16	-	64.63	44.02	52.48	22.61	21.04	33.81	39.77
	3	SLIM-LLM	61.86	38.72	51.69	22.04	21.35	33.25	38.15
OPT-350M	3	OPTQ	60.07	39.39	50.59	22.28	22.58	32.92	37.97
	3	OWQ	62.68	39.94	52.81	22.24	23.04	33.01	38.95
	2.99	MLWQ	**62.73**	**40.78**	**52.95**	**22.57**	**23.19**	**33.67**	**39.31**
	16	-	71.76	57.02	59.35	23.31	22.42	33.86	44.62
	3	SLIM-LLM	69.04	54.84	57.22	23.55	20.89	33.15	43.12
OPT-1.3B	3	OPTQ	67.74	47.01	57.21	22.47	**22.58**	33.81	41.80
	3	OWQ	70.24	**55.72**	57.54	21.75	21.27	33.11	43.27
	2.99	MLWQ	**70.26**	52.01	**57.58**	**23.82**	21.73	**34.89**	**43.38**
	16	-	73.77	60.77	61.01	23.89	21.04	33.72	45.70
	3	SLIM-LLM	71.76	56.14	**59.74**	24.45	23.34	33.72	44.86
OPT-2.7B	3	OPTQ	71.38	48.19	59.68	23.52	19.82	33.40	42.67
	3	OWQ	71.76	**59.09**	59.27	23.42	19.82	33.50	44.48
	2.99	MLWQ	**71.81**	57.36	58.92	**24.65**	**23.96**	**33.89**	**45.10**
	16	-	74.21	65.44	60.61	28.91	21.96	32.95	47.35
	3	SLIM-LLM	66.43	52.61	54.89	23.45	19.66	32.15	41.53
LLAMA-3.2-1B	3	OPTQ	56.15	31.14	52.01	20.94	25.04	32.81	36.35
	3	OWQ	66.46	**54.12**	54.99	23.89	21.22	34.10	42.46
	2.99	MLWQ	**67.13**	49.43	**55.03**	**24.47**	**25.18**	**34.36**	**42.60**
	16	-	74.65	74.41	70.01	34.64	22.73	34.18	51.77
	3	SLIM-LLM	68.33	52.86	62.03	**32.22**	22.27	34.86	45.43
LLAMA-3.2-3B	3	OPTQ	54.68	29.67	50.91	21.94	22.73	32.62	35.43
	3	OWQ	71.14	**64.91**	**64.17**	27.67	24.58	34.10	47.76
	2.99	MLWQ	**71.16**	61.95	62.38	28.49	**28.42**	**34.98**	**47.89**

Table 4: Quantization Results for Zero-Shot Tasks on OPT and LLAMA-3.2

6.5 Extended Comparisons with KL-Divergence-Based Methods

Compared to the KL-divergence method, the BPLL-based method consistently achieves lower perplexity across all model sizes, indicating improved performance in preserving model accuracy under aggressive quantization.

Table 5 presents the perplexity (PPL) results of two 3-bit quantization methods KL+MQSA+TQP and BPLL+MQSA+TQP on OPT models of varying sizes (125M to 2.7B parameters).

OPT	BIT	125M	350M	1.3B	2.7B
-	16	27.65	22.01	14.62	12.47
KL+MQSA+TQP	2.99	33.87	27.46	16.38	13.01
BPLL+MQSA+TQP	2.99	**28.98**	**23.82**	**15.24**	**12.55**

Table 5: Comparison of different loss functions

6.6 End-to-End Inference Speedups

We adopt the open-source AutoGPTQ framework along with the mixed-precision computation support provided by SliM-LLM. Our evaluation focuses on the deployment performance of Llama-

3.2-1B and Llama-3.2-3B under 3-bit quantization. The experiment reveals that the proposed mixed-precision strategy significantly reduces perplexity, while maintaining a high compression ratio on GPU hardware. The inference speed on the RTX 4090 is comparable to that of SliM-LLM. Moreover, the overhead from dequantization and aligning inference across different bit-widths results in a slower inference speed compared to the original model. The experimental results are shown in Appendix B.

7 Conclusion

This paper introduces the Multi-Level Weight Quantization (MLWQ) method, which addresses the challenges of deploying small language models (SLMs) on resource-constrained edge devices. By employing a three-step optimization approach, MLWQ effectively reduces the bit-width of model parameters while preserving model performance. We adopt a channel-wise distribution loss to guide bit-width assignment across layers. Under the guidance of this global allocation, each layer further categorizes channels into three groups based on their importance. Finally, quantization parameters are fine-tuned for each group to further reduce quantization error. Experimental results demonstrate that MLWQ significantly outperforms state-of-the-art post-training quantization methods, offering a promising solution for the efficient deployment of SLMs without compromising performance.

Limitations

Although MLWQ significantly reduces the model size, it faces several inherent limitations: (1) The memory and bandwidth overhead caused by unquantized activations still limits the acceleration of end-to-end inference. (2) Decoding overhead, introduces complex branching and bit manipulation in CUDA kernels, which may reduce parallel efficiency.

Acknowledgements

We thank all the reviewers for their insightful comments. This work is supported by the National Natural Science Foundation of China (No. 62472330) and the Wuhan City Joint Innovation Laboratory for Next-Generation Wireless Communication Industry Featuring Satellite-Terrestrial Integration (No. 4050902040448).

References

Meta AI. 2024a. Llama-3.2-1b. https://huggingface.co/meta-llama/Llama-3.2-1B.

Meta AI. 2024b. Llama-3.2-3b. https://huggingface.co/meta-llama/Llama-3.2-3B.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Lewis Tunstall, Agustín Piqueres, Andres Marafioti, Cyril Zakka, Leandro von Werra, and Thomas Wolf. 2024. Smollm2 - with great data, comes great performance.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

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.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M De Sa. 2023. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems*, 36:4396–4429.

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*.

Yufei Cui, Shangyu Wu, Qiao Li, Antoni B Chan, Tei-Wei Kuo, and Chun Jason Xue. 2022. Bits-ensemble: Toward light-weight robust deep ensemble by bits-sharing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(11):4397–4408.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35:30318–30332.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. 2023. Spqr: A sparse-quantized representation for near-lossless llm weight compression. *arXiv preprint arXiv:2306.03078*.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Optq: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Wei Huang, Yangdong Liu, Haotong Qin, Ying Li, Shiming Zhang, Xianglong Liu, Michele Magno, and

Xiaojuan Qi. 2024a. Billm: Pushing the limit of post-training quantization for llms. *arXiv preprint arXiv:2402.04291*.

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

Minsoo Kim, Sihwa Lee, Sukjin Hong, Du-Seong Chang, and Jungwook Choi. 2022. Understanding and improving knowledge distillation for quantization-aware training of large transformer encoders. *arXiv preprint arXiv:2211.11014*.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. 2024. Squeezellm: Dense-and-sparse quantization. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. 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, pages 13355–13364.

Liang Li, Qingyuan Li, Bo Zhang, and Xiangxiang Chu. 2024. Norm tweaking: High-performance low-bit quantization of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18536–18544.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: **phi-1.5** technical report. *arXiv preprint arXiv:2309.05463*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*.

James O' Neill and Sourav Dutta. 2023. Self-distilled quantization: Achieving high compression rates in transformer-based language models. *arXiv preprint arXiv:2307.05972*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740.

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. *arXiv preprint arXiv:2308.13137*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

Model / Acc↑	bit	Method	PIQA	ARC-Easy	Winogrande	MathQA	LogiQA	ANLI_R2	Avg.
SmolLM2-135M	16	-	68.38	64.39	52.41	21.32	21.96	34.38	52.57
	3	SliM-LLM	58.97	40.99	51.85	20.51	21.35	32.58	37.71
	3	OPTQ	55.61	38.82	50.75	22.04	21.26	32.15	36.77
	3	OWQ	57.94	43.69	49.25	21.04	**23.51**	33.35	38.13
	2.99	MLWQ	**59.17**	**43.81**	**52.09**	**22.08**	21.04	**35.68**	**38.98**
SmolLM2-360M	16	-	71.76	70.49	58.87	19.89	21.85	34.59	46.24
	3	SliM-LLM	65.12	55.81	52.48	19.49	**21.96**	34.25	41.52
	3	OPTQ	61.53	47.98	51.38	22.02	21.12	33.01	39.51
	3	OWQ	62.95	45.75	51.85	21.41	19.05	33.82	39.14
	2.99	MLWQ	**65.27**	**57.28**	**53.35**	**22.91**	21.87	**35.15**	**42.64**
SmolLM2-1.7B	16	-	77.04	77.69	66.29	19.25	21.65	32.85	49.13
	3	SliM-LLM	**70.62**	**64.73**	58.87	18.25	21.35	33.29	44.52
	3	OPTQ	53.92	30.85	50.91	20.84	21.32	33.41	35.21
	3	OWQ	58.61	37.33	50.12	22.28	21.04	33.56	37.16
	2.99	MLWQ	67.38	62.25	**58.93**	**23.91**	**21.59**	**33.75**	**44.63**
Phi-1.5	16	-	76.49	76.26	72.84	30.15	23.96	34.68	52.40
	3	SliM-LLM	**74.42**	72.93	70.56	26.36	23.34	32.71	50.05
	3	OPTQ	73.56	70.24	69.14	26.97	22.27	33.48	49.28
	3	OWQ	73.88	73.02	70.38	27.64	23.42	**33.51**	50.31
	2.99	MLWQ	72.63	**73.05**	**70.76**	**28.91**	**23.96**	32.77	**50.35**
Phi-2	16	-	78.56	79.88	75.29	31.05	25.81	38.21	54.80
	3	SliM-LLM	**77.52**	**77.81**	68.97	29.34	23.51	36.01	52.19
	3	OPTQ	74.05	71.93	67.41	26.83	24.68	34.40	49.88
	3	OWQ	76.06	76.11	69.32	29.15	23.66	34.22	51.42
	2.99	MLWQ	76.59	76.88	**69.85**	**30.82**	**24.93**	**36.95**	**52.67**

Table 6: Quantization Results for Zero-Shot Tasks on SmolLM2 and Phi

#W		Llama-3.2-1B				Llama-3.2-3B			
		WM	RM	PPL↓	Token/s	WM	RM	PPL↓	Token/s
FP16	-	2.35G	2.38G	9.75	56.81	6.13G	6.17G	7.81	41.62
3-bit	SliM-LLM	0.36G	0.91G	25.01	45.13	1.06G	1.88G	21.26	33.05
	MLWQ	0.36G	0.89G	15.73	45.21	1.02G	1.78G	10.23	32.58

Table 7: Deployment results of MLWQ and Slim-LLM on GPU.

A SmolLM2 & Phi

Table 6 compares SmolLM2 and the Phi family. For SmolLM2-360M on ARC-Easy, MLWQ attains 57.28, surpassing SliM-LLM (55.81), OPTQ (47.98), and OWQ (45.75), thereby demonstrating better preservation of reasoning capability. For Phi-2 on MathQA, MLWQ achieves 30.82, exceeding SliM-LLM (29.34), OPTQ (26.83), and OWQ (29.15), underscoring its advantage in numerical reasoning. Overall, the experimental results indicate that MLWQ consistently outperforms competing methods across most tasks and models.

B Accelerate experiment

The specific memory and throughput experimental results are shown in Table 7. The experiments demonstrate that the proposed mixed-precision strategy effectively lowers perplexity while sustaining a high compression ratio on GPU hardware.