EAC-MoE: Expert-Selection Aware Compressor for Mixture-of-Experts Large Language Models

Yuanteng Chen[1,2] [*], **Yuantian Shao**[3,1] [*], **Peisong Wang**[1,2,4†], **Jian Cheng**[1,2,4,5]
[1] C^{2}DL, Institute of Automation, Chinese Academy of Sciences
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences
[3]Nanjing University of Science and Technology
[4]AIRIA [5]Maicro.ai
chenyuanteng2024@ia.ac.cn, yuantianshao@njust.edu.cn,
{peisong.wang,jcheng}@nlpr.ia.ac.cn

Abstract

Mixture-of-Experts (MoE) has demonstrated promising potential in scaling LLMs. However, it is hindered by two critical challenges: (1) substantial GPU memory consumption to load all experts; (2) low activated parameters cannot be equivalently translated into inference acceleration effects. In this work, we propose **EAC-MoE**, an **E**xpert-Selection **A**ware **C**ompressor for **MoE**-LLMs, which deeply aligns with the characteristics of MoE from the perspectives of quantization and pruning, and introduces two modules to address these two challenges respectively: (1) The expert selection bias caused by low-bit quantization is a major factor contributing to the performance degradation in MoE-LLMs. Based on this, we propose **Quantization with Expert-Selection Calibration (QESC)**, which mitigates the expert selection bias by calibrating the routers within the MoE; (2) There are always certain experts that are not crucial for the corresponding tasks, yet causing inference latency. Therefore, we propose **Pruning based on Expert-Selection Frequency (PESF)**, which significantly improves inference speed by pruning less frequently used experts for current task. Extensive experiments demonstrate that our approach significantly reduces memory usage and improves inference speed with minimal performance degradation.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in various natural language processing tasks. (Zhou et al., 2024). A recent significant breakthrough in this field is the introduction of the Mixture-of-Experts (MoE) architectures (Shazeer et al., 2017; Anonymous, 2024). By utilizing a sparse architecture that activates a subset of experts via a dynamic routing mechanism tailored to each input, MoE enables efficient computation and scalable network capacity, matching

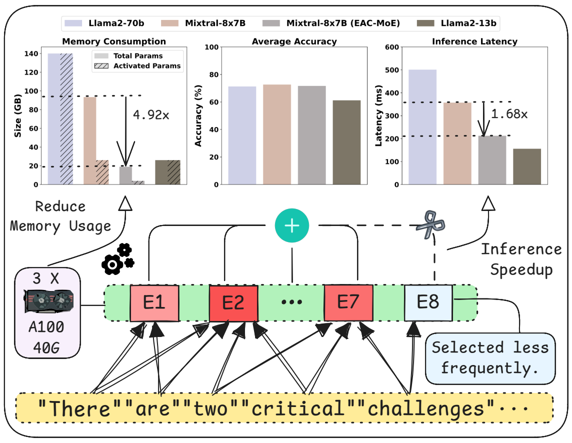


Figure 1: Comprehensive performance of EAC-MoE in reducing memory usage, maintaining model accuracy, and improving inference speed for Mixtral-8x7B. The average accuracy is measured across zero-shot tasks.

or exceeding the performance of dense LLMs with several times more activated parameters.

Although MoE reduces the number of activated parameters through an expert selection mechanism, it does not decrease the total number of model parameters. During inference, all expert weights must be stored in GPU memory, resulting in substantial memory pressure. As shown in Figure 1 top, while Mixtral-8x7B (Jiang et al., 2024) has a similar activated parameter count to LLaMA2-13B (Touvron et al., 2023), its total parameter count is about four times larger, occupying 94GB of GPU memory.

On the other hand, the reduction in activated parameters does not directly result in an equivalent speedup during inference. Although only a subset of experts is selected for each token, in typical long-sequence or batch inference scenarios, different tokens choose different experts. As illustrated in Figure 1 bottom, MoE still requires computing the output of each expert (E1-E8) separately and performing a weighted summation to obtain the final result, experts like E8 are selected less frequently but still cause non-negligible latency.

These challenges hinder the practical deploy-

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

ment of MoE models in resource-constrained, low-latency applications. For dense LLMs, quantization and pruning are commonly employed to address these issues. However, directly applying commonly used quantization methods (such as RTN and GPTQ (Frantar et al., 2022)) and pruning methods designed for dense LLMs to MoE models, without considering the characteristics of MoE models, results in significant performance degradation or brings negligible inference speedup. In this work, we design a method that combines quantization and sparse inference, leveraging the expert selection characteristics of MoE models.

In MoE models, the experts are trained to specialize for different types of tasks, and the router can select the most suitable experts for each token, which is the key for its success (Jordan and Jacobs, 1994). However, low-bit quantization of MoE model can bias expert selection probability and cause the router to choose the wrong experts, which we refer to as **the expert-shift problem**. To address this issue in MoE quantization, we propose Quantizaion with Expert-Selection Calibration (QESC): a layer-by-layer router calibration method to mitigate the bias caused by quantization, thereby reducing the shift in expert selection. This approach effectively preserves the performance of the quantized model.

In contrast, the focus of dynamic pruning lies in skipping experts that are relatively unimportant for the current input during inference. Specifically, certain experts are less frequently selected during inference and have minimal impact on overall performance. Notably, these relatively unimportant experts vary across different types of tasks. Based on this observation, we propose Pruning based on Expert-Selection Frequency (PESF): a dynamic expert pruning method that prunes less frequently selected experts during inference, significantly improving the inference speed of MoE models with minimal performance loss.

Combining QESC and PESF, we propose **EAC-MoE**, exploring the compression of MoE models from both aspects of pre-inference and during-inference. Experiments on four MoE models demonstrate that our method significantly reduces memory usage and improves inference speed. When compressing Mixtral-8x7B, as shown in Figure 1 top, we reduce the memory requirements by 4.92×, enabling deployment on a RTX 3090 GPU. Meanwhile, our method achieve 1.68× inference speedups with an average accuracy loss of less than 1% under simultaneous quantization and pruning, making it practical for real-world applications.

2 Related Work

Quantization for LLMs and MoE-LLMs. Post-Training Quantization (PTQ) is an efficient technique that reduces computational and storage requirements by converting pre-trained models from high-precision to lower-precision formats without requiring extensive retraining. Methods like GPTQ (Frantar et al., 2022) and BiLLM (Huang et al., 2024b) focus on addressing weight-only quantization, while approaches such as SmoothQuant (Xiao et al., 2023a) and OmniQuant (Shao et al., 2023) aim to tackle the challenges of both weight and activation quantization. In this work, we focus primarily on weight-only quantization because the MoE deployment challenges stem mainly from the memory pressure caused by weight parameters. For MoE-LLMs, previous studies have largely focused on mixed-precision quantization strategies based on expert selection frequency (Li et al., 2024a; Huang et al., 2024a). Although these methods have shown certain effectiveness, they may face challenges in generalization and risk overfitting.

Pruning of LLMs and MoE-LLMs. Post-training pruning is another key technique to compress LLMs by reducing model size by selectively removing less important parameters while preserving performance (Han et al., 2016; Zhu and Gupta, 2018; Ashkboos et al., 2024a). For MoE-LLMs, prior efforts have focused mainly on two directions: pruning experts with lower selection frequency before inference (Lu et al., 2024; Kim et al., 2021), and pruning less significant weights for each token among the selected experts (Lu et al., 2024; Huang et al., 2024a). However, while these approaches have made notable progress, there remain opportunities for further improvement. The first direction, for example, can lead to performance degradation in certain types of tasks. The second direction, on the other hand, achieves a relatively low pruning rate, resulting in limited inference speedup.

3 Preliminaries and Motivation

3.1 LLM Quantizaiton

In this work, quantization techniques are employed to compress the weights. Specifically, floating-point weights distributed in $[W_{\min}, W_{\max}]$ are mapped to the integer range $[0, 1, \cdots, 2^B - 1]$, where B represents the target bit-width. The quan-

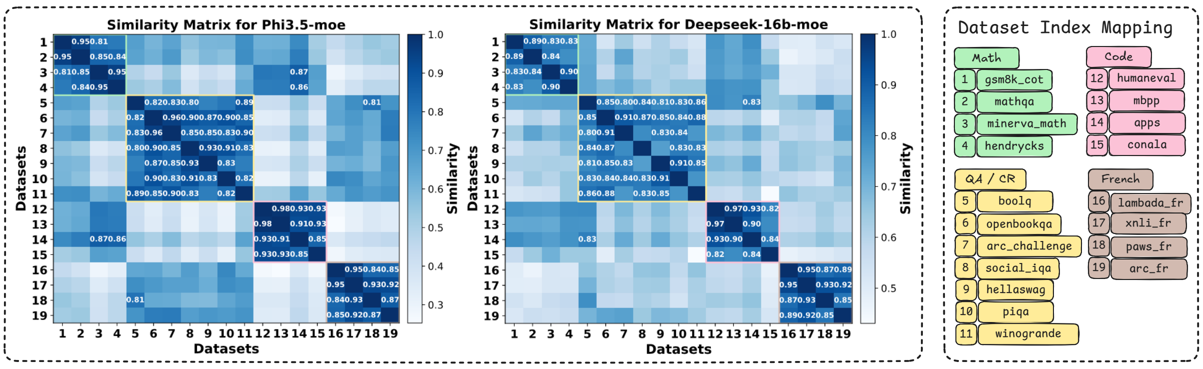


Figure 2: The figure illustrates the pairwise cosine similarity of expert selection frequencies for Phi3.5-moe (left) and Deepseek-moe-16b-base (right) across 19 datasets, which are categorized into four groups distinguished by different colors. Points with cosine similarity greater than 0.8 are highlighted to emphasize high similarity regions.

tization reconstruction problem for the weights $\boldsymbol{W} \in \mathbb{R}^{n_{\text{in}} \times n_{\text{out}}}$ can be formulated as:

$$\arg\min_{\boldsymbol{W}_q} \|\boldsymbol{W}\boldsymbol{X} - \boldsymbol{W}_q\boldsymbol{X}\|_2^2, \tag{1}$$

where $\boldsymbol{W}_q$ denotes the quantized weight, and $\boldsymbol{X}$ is the input to the layer derived from a small subset of calibration data. GPTQ (Frantar et al., 2022) is currently a mainstream weight quantization method, which can efficiently reduce group-wise quantization error by employing Hessian-based estimation ($\boldsymbol{H} = 2\boldsymbol{X}\boldsymbol{X}^\top$) and error compensation techniques. It is utilized in subsequent sections of this paper.

3.2 Mixture-of-Experts

Decoder-only MoE models (Gale et al., 2023) are based on a transformer architecture (Vaswani et al., 2017), but the FeedForward Network (FFN) sublayers of traditional dense models are replaced with MoE layers, each containing N experts. For each input token $\boldsymbol{x}$, the router computes routing logits $\boldsymbol{r} = \{r_0, \cdots, r_{N-1}\}$ and expert selection scores $\boldsymbol{s} = \text{Softmax}(\boldsymbol{r})$. The top-$K$ experts are selected based on $\boldsymbol{s}$, and their outputs $E_{e_j}(\boldsymbol{x})$ are combined as a weighted sum, with normalized weights:

$$\boldsymbol{z} = \sum_{j=0}^{K-1} \frac{\boldsymbol{s}_{e_j}}{\sum_{i=0}^{K-1} \boldsymbol{s}_{e_i}} \cdot E_{e_j}(\boldsymbol{x}). \tag{2}$$

Here, $E_{e_j}(\boldsymbol{x})$ represents the output of the j-th selected expert for the input token $\boldsymbol{x}$. Based on this structure and mechanism, models such as Mixtral-8x7B (Jiang et al., 2024), GPT-4 (OpenAI et al., 2024) and DeepSeek-V3 (DeepSeek-AI et al., 2024) have achieved superior generative abilities.

3.3 Expert-Selection (ES) Analysis

Previous quantization studies for MoE-LLMs have primarily focused on the observation that, during inference, MoE models exhibit significant differences in the selection frequency of different experts (Li et al., 2024a). Consequently, expert selection frequency has been widely adopted as a metric to evaluate the importance of different experts within an MoE layer. However, prior works have overlooked an important pattern: MoE models often demonstrate entirely different expert preferences across different types of tasks.

To investigate this pattern, we examine three common categories of NLP tasks: Math, Code-Generation, and Question-Answering or Commonsense-Reasoning (QA/CR). Additionally, we analyze tasks in specific languages (French in our case) as a separate category. For each dataset, we record the expert selection frequency during inference. Furthermore, we calculate the similarity of expert selection frequencies between every pair of datasets to better understand the diversity in expert preferences across tasks. For a certain MoE layer m in a MoE model, the normalized expert selection frequency for dataset d is defined as:

$$P(m,d) = \frac{C(m,d)}{\sum_{i=0}^{N-1} C(m,d,i)} \tag{3}$$

where $C(m,d) = [C(m,d,0), \cdots, C(m,d,N-1)]$, with $C(m,d,i)$ representing the count of the i-th expert in layer m is selected for all input tokens in the dataset d. Then the normalized expert selection frequencies $P(m,d)$ of all MoE layers are flattened into a single vector $P(d)$. Based on this, the similarity of expert preferences between two datasets d_i and d_j is computed as:

$$\text{Sim}(d_i, d_j) = \frac{P(d_i) \cdot P(d_j)}{\|P(d_i)\| \|P(d_j)\|} \tag{4}$$

As shown in Figure 2, we calculate the expert preference similarities of Phi3.5-moe (Abdin et al.,

2024) and DeepSeek-16b-moe-base models across 19 different datasets. The results indicate that both models reach similar conclusions: expert selection frequencies within datasets of the same task category exhibit high similarity, whereas expert selection frequencies across datasets of different task categories show relatively low similarity.

This observation suggests that MoE models rely primarily on different experts to handle different types of tasks and the importance of the same expert may vary drastically across different tasks, providing us with the following two insights:

1. For **static** quantization, we should focus on the expert selection process itself—ensuring that the model can still select the experts important for each task, as we cannot permanently determine the importance of any expert before inference using a calibration set.

2. For **dynamic** pruning, we should dynamically evaluate the importance of experts based on the type of the current task and prune experts that are not important for the current task.

4 Quantization with ES Calibration

The core idea of our method is to mitigate the performance degradation of quantized MoE models by addressing expert-shift, a critical issue where quantization errors in the multi-head self-attention (MHSA) and MoE blocks distort expert selection probabilities, causing routers to deviate from original expert assignment patterns.

4.1 Importance of ES Calibration

We first verify the importance of calibrating expert selection by observing performance degradation caused by expert-shift and performance improvement achieved by preserving the expert selection. We separately record the expert selection and its corresponding scores (s) for all inputs on the WikiText2 (Merity et al., 2016) validation set for both full-precision model and the 3-bit quantized model. Then, we enforce the quantized model to use the expert selection scores of the original precision model for each input (quantized but without expert-shift) and, conversely, enforce the original precision model to use the expert selection scores of the quantized model (not quantized but with expert-shift). Finally, we calculate the perplexity (PPL) of the inputs under these four conditions respectively.

As shown in Table 1, expert-shift causes significant performance degradation for the original model. Conversely, preserving the expert selection of the original model significantly improves the performance of quantized models, highlighting the importance of calibrating expert selection.

Table 1: The impact of weight quantization itself and its induced expert-shift on perplexity (PPL↓) for Mixtral-8x7B and Deepseek-moe-16b-base models.

Model	Quantized	Expert-Shift	PPL
Mixtral-8x7B	✗	✗	3.84
	✗	✔	4.17
	✔	✗	4.21
	✔	✔	4.65
Deepseek-moe -16b-base	✗	✗	6.51
	✗	✔	6.76
	✔	✗	6.81
	✔	✔	7.17

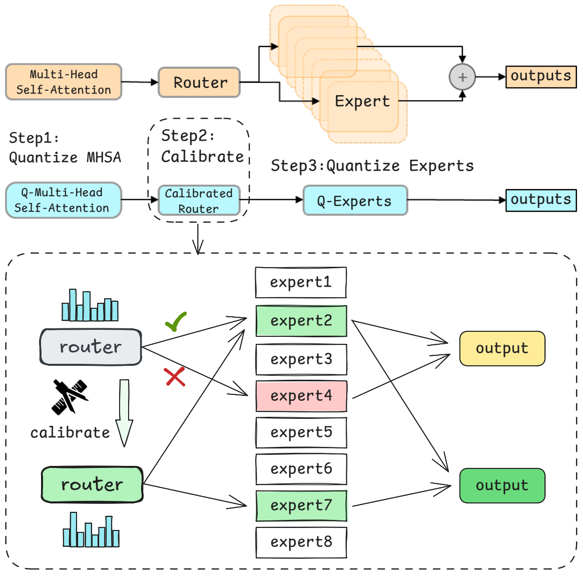


Figure 3: Framework of our proposed Quantization with Experts-Selection Calibration.

4.2 Layer-by-layer Calibration Framework

Then we focus on how to mitigate expert-shift problem. At a hight level, our method performs quantization and calibration layer-by-layer. Concretely, as illustrated in Figure 3, using the WikiText2 calibration dataset, we sequentially quantize the MHSA components, calibrate the routers of the MoE layers, and quantize all experts layer by layer. This process allows the router in each layer to be calibrated in a way that mitigates the expert-shift caused by the quantization of the adjacent layer's MHSA and MoE layer, thereby preventing the cumulative accumulation of expert selection shift across layers.

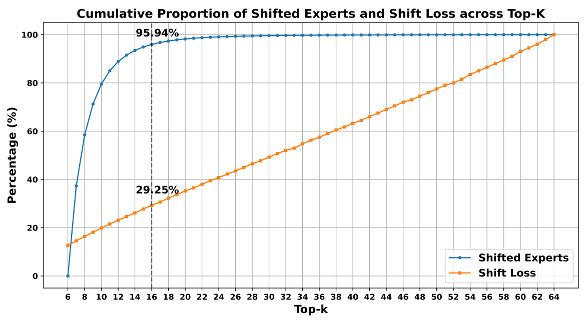


Figure 4: Cumulative proportion of shifted experts in the top-K experts of the probability distribution (blue; i.e., number of shifted experts in top-K / total number of shifted experts), and cumulative proportion of the shift loss of the top-K experts in the probability distribution relative to the total loss of all experts (orange)

4.3 TopK-MSE Loss

To calibrate the router, a natural idea is to align the router's outputs before and after input quantization, such as by using the mean squared error (MSE) loss for optimization. However, this method is not effective for MoE models with a large number of experts, such as Deepseek-moe-16b-base—which selects 6 experts out of 64 (Dai et al., 2024). Comparing expert selection before and after 2-bit quantization, as shown in Figure 4, we observe that among the experts selected in full precision but not selected after quantization (shifted experts), 95.9% still rank within the top 16 in the probability distribution. However, the loss corresponding to the top 16 experts accounts for only 29.25% of the total MSE loss. This indicates that if we directly apply MSE loss to all experts, the loss will be dominated by the majority of experts with very small selection probabilities, which are not selected in full precision, thereby introducing noise into the optimization process.

Based on this insight, we adopt the TopK-MSE loss, which computes the MSE loss over only the top-K classes with the highest probabilities, allowing the optimization process to focus on aligning the experts that are more likely to be selected. The TopK-MSE loss is calculated as follows:

$$\mathcal{L} = \frac{1}{K} \sum_{i \in \text{top-}K(\boldsymbol{W}\boldsymbol{x})} \left((\boldsymbol{W}\boldsymbol{x})_i - (\boldsymbol{W}\hat{\boldsymbol{x}})_i\right)^2, \quad (5)$$

$\boldsymbol{W}$ represent the weight matrix of router and $\hat{\boldsymbol{x}}$ denotes the input obtained from the quantized model.

5 Pruning based on ES Frequency

QESC focuses on ensuring the quantized model can still correctly select the experts important for the current task. A natural consideration is that there are also experts that are not important for the current task. In this section, we introduce a dynamic expert pruning method during inference, which significantly improves inference speed while maintaining almost the same level of accuracy.

Prior work (Lu et al., 2024) has already noted the sparsity in expert selection for MoE models, where certain experts are selected with high frequency for a specific task, while others are rarely selected (shown in Appendix A.11). Meanwhile, as concluded in Section 3.3, it is crucial to dynamically evaluate the importance of each expert during inference for different tasks. Therefore, unlike prior work that performs static expert pruning based on selection frequency before inference, our approach dynamically identifies experts that are less important for the current task during the inference process. This allows us to achieve significant inference speedup with minimal performance degradation.

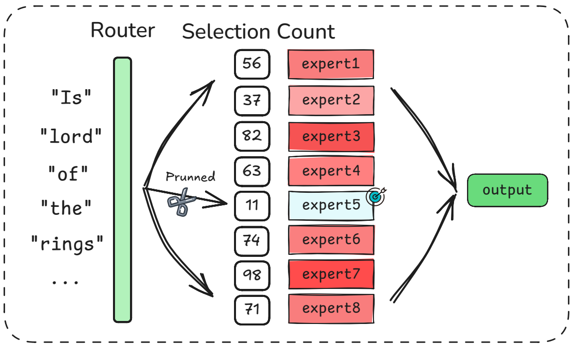


Figure 5: Framework of our proposed Pruning based on Experts-Selection Frequency.

In our method, the dynamic pruning criterion is set as follows: assume each layer of the MoE model has N experts, each token selects K experts, and the input sequence length is l. The dynamic **pruning threshold** is defined as α ($0 < \alpha \leq 1$). If the number of times an expert is selected, denoted as c, satisfies the condition:

$$c < \left(\frac{l \times K}{N}\right) \times \alpha \quad (6)$$

then the expert is pruned. In other words, if an expert is selected less frequently than the average selected count multiplied by the threshold α (like $expert5$ in Figure 5), it is pruned and excluded from the computation for this sequence.

Table 2: Comparison of the average perplexity (PPL) scores on WikiText2 validation set and the average accuracy on 8 zero-shot tasks across four different MoE models. We reproduce results of BSP and PMQ on four models using the official codebases provided in their repositories (the reproduction details are provided in Appendix A.6) and evaluated all the results under the same settings. Full results are in the Appendix A.7.

Bits	Method	Mixtral-8x7B		Phi3.5-moe		Deepseek-moe-16b-base		Qwen1.5-MoE-A2.7B	
		PPL ↓	0-shot[8] ↑	PPL ↓	0-shot[8] ↑	PPL ↓	0-shot[8] ↑	PPL ↓	0-shot[8] ↑
16.00	Baseline	3.84	72.64	3.99	69.62	6.51	61.38	7.22	64.72
2.06	GPTQ	5.51	62.56	5.32	64.45	8.27	54.88	9.92	57.76
	PMQ	5.41	63.25	5.88	61.35	8.42	54.79	9.89	57.79
	QESC	**5.09**	**66.31**	**5.22**	**65.03**	**7.99**	**57.05**	**8.30**	**59.52**
2.54	GPTQ	4.74	68.65	4.74	65.81	7.36	56.83	8.41	57.91
	BSP	4.98	65.44	4.72	66.15	7.32	58.24	8.11	60.40
	PMQ	4.78	67.5	4.73	66.03	7.17	58	8.09	60.47
	QESC	**4.54**	**69.61**	**4.66**	**66.53**	**7.08**	**58.33**	**7.74**	**61.47**
3.03	GPTQ	4.16	68.92	4.28	68.12	6.82	59.33	7.69	62.21
	BSP	4.25	67.22	4.61	67.67	7.05	59.39	7.86	60.88
	QESC	**4.14**	**72.21**	**4.24**	**68.49**	**6.71**	**61.22**	**7.50**	**62.89**

6 Experiment

In this section, we first evaluate the experimental performance of our proposed methods QESC and PESF, respectively. Then we combine quantization and pruning (QESC+PESF) to assess their performance in maintaining model accuracy, memory usage reduction, and actual inference speedup.

6.1 Setup

Models and Dataset. We validate our method on four MoE models: Mixtral-8x7B, Phi3.5-moe, Deepseek-moe-16b-base and Qwen1.5-MoE-A2.7B (Yang et al., 2024). We report perplexity (PPL) on the WikiText2 testset and accuracies of eight zero-shot tasks tested by EleutherAI LM Harness (Gao et al., 2024), including Winogrande (ai2, 2019), PIQA (Bisk et al., 2020), ARC-Easy, ARC-Challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), MathQA (Amini et al., 2019), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021b). Additionally, we present the results of our method on the challenging tasks GSM8K (Cobbe et al., 2021) and HumanEval (Chen et al., 2021).

Implementation Details. We follow the settings of prior work (Li et al., 2024a; Huang et al., 2024a), keeping the MHSA components at 4-bit precision, while quantizing all experts to 2-bit or 3-bit precision, and maintaining the router at its original precision. Overall, we evaluate our method under three average bit-width settings: 2.06-bit, 2.54-bit, and 3.03-bit (detailed bit-width setting is discussed in Appendix A.5). The quantization employs group-wise (group size 128) asymmetric quantization and follows the GPTQ procedure. We use 128 sequences of length 2048 from the WikiText2 training set as the calibration set for QESC.

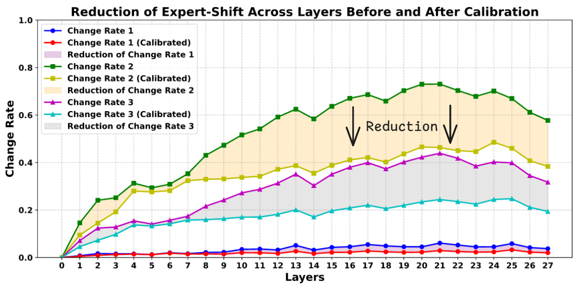


Figure 6: The reduction of expert-shift before and after calibration measured by expert-selection change rate across layers in Deepseek-moe-16b-base under 2.06-bit quantization. Change Rate 1-3 respectively represent three metrics: all expert selections changed, at least one selection changed and half or more selections changed.

6.2 Experiment on Quantization

Reduction in Expert-Shift. First, we intuitively validate the effectiveness of our calibration method by measuring the expert selection change rate before and after calibration on WikiText2 validation set. We calculate the expert selection change rates of the quantized model with or without router calibration relative to the full-precision model on Deepseek-moe-16b-base, and show the relative reduction in Figure 6. The results demonstrate that our calibration method significantly reduces the expert selection change rate in quantized MoE models across three metrics.

Overall Performance. We further validate the overall performance of our method. We compare our quantization method with three other methods: GPTQ, PMQ (Li et al., 2024a), and BSP (Li et al., 2024a). GPTQ serves as the baseline for uniform bit-width quantization, while PMQ (1.57–2.54 bit) and BSP (2.54–3.03 bit) are current SOTA methods for mixed-precision quantization of MoE models.

Table 3: Comparison of the average accuracy on 8 zero-shot tasks and speedup of inference across four different MoE models. The speedup is calculated based on the total inference time of the model with dynamic pruning compared to the original model across 8 tasks. We reproduce results of EES and ODP (details are provided in Appendix A.8) and evaluate all the results under the same settings. Full results can be found in Appendix A.9.

Method	Mixtral-8x7B		Phi3.5-moe		Deepseek-moe-16b-base		Qwen1.5-MoE-A2.7B	
	0-shot ↑	Speedup ↑	0-shot ↑	Speedup ↑	0-shot ↑	Speedup ↑	0-shot ↑	Speedup ↑
Baseline	72.64	1.00	69.62	1.00	61.38	1.00	64.72	1.00
EES	71.40	1.06	67.96	1.05	61.15	1.08	64.42	1.06
ODP	71.98	1.05	68.92	1.04	61.19	1.08	64.48	1.06
PESF ($\alpha = 0.3$)	**72.19**	1.08	**69.27**	1.12	**61.28**	1.11	**64.64**	1.14
PESF ($\alpha = 0.7$)	58.22	**1.13**	67.95	**1.30**	60.41	**1.45**	63.87	**1.47**

It is worth noting that QESC is inherently orthogonal to other weight quantization approaches for LLMs that focus on minimizing quantization error.

As shown in Table 2, when only GPTQ is used to reduce quantization loss, significant performance degradation is still observed. Both BSP and PMQ, as mixed-precision quantization methods, demonstrate performance improvements over GPTQ at certain quantization bit-widths for some models. However, in nearly half of the settings, their results are inferior to those of GPTQ, indicating a certain degree of lack of generalization. In contrast, the proposed QESC method significantly outperforms GPTQ, BSP, and PMQ across all results. For instance, at 2.54-bit, QESC limits the performance loss to around 3% for all four models. Notably, at 3.03-bit, QESC reduces the loss to within 0.5% for Mixtral-8x7B and Deepseek-moe-16b-base, making it suitable for practical application scenarios.

Challenging Tasks. Apart from PPL and commonsense tasks, we also evaluate our QESC method on the challenging tasks GSM8K and HumanEval, with the results provided in Appendix A.2.

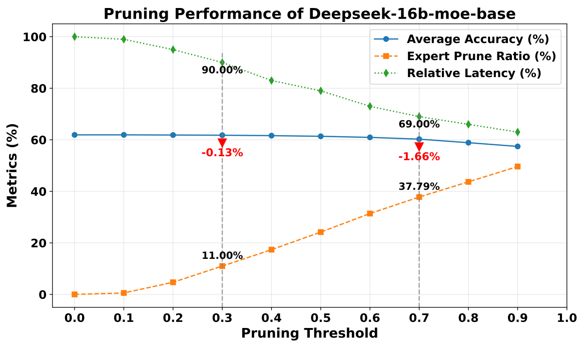


Figure 7: The variations in the model's average accuracy, expert pruning rate, and inference acceleration effect with respect to changes in the pruning threshold α.

6.3 Experiment on Pruning

Pruning Threshold Analysis. To determine a relatively appropriate pruning threshold, we aim to trade off among model accuracy, expert pruning rate, and relative inference latency. As shown in Figure 7, we conduct experiments on Deepseek-16b-moe-base, adjusting the pruning threshold (α) from 0 to 0.9 with an interval of 0.1. For each threshold, we calculate the average accuracy on 8 zero-shot tasks, the average expert pruning rate across all layers, and the percentage of relative inference latency compared to the original model. The results show that pruning thresholds of 0.3 and 0.7 represent two sweet spots. The former achieves approximately 10% speed improvement with almost no loss to the model (average loss within 0.5%), while the latter is more aggressive, achieving over 1.3× average inference speedup while still keeping the average loss within around 1.5%.

Overall Performance. We compare our method with the classical MoE expert pruning method, known as Efficient Experts Skipping (EES) (Lu et al., 2024), and a recently proposed MoE pruning method, ODP (Huang et al., 2024a). EES performs pruning from the perspective of individual tokens, skipping the selected experts with negligible scores for each input token, while ODP incorporates a key token protection mechanism on top of this. However, both methods can only reduce the input size for a subset of experts, resulting in limited inference speedup. In contrast, our PESF method performs pruning from the perspective of experts, directly skipping experts that are selected less frequently for the current sequence. As shown in Table 3, under the more conservative setting ($\alpha = 0.3$), our method significantly outperforms EES and ODP on all four models in both average accuracy and relative speedup. Moreover, compared to EES and ODP, our pruning method demonstrates greater flexibility. Notably, when we adopt a more aggressive setting with ($\alpha = 0.7$), except for Mixtral-8x7B (discussed in Appendix A.12), our method achieves an inference speedup of 1.30x or greater on the other three models, while still maintaining model accuracy comparable to ODP.

Table 4: The overall performance of our compression method which combines QESC (3.03 bit) and PESF ($\alpha = 0.3$). "Params" denotes the parameter size, including quantizer parameters for the compressed model.

Models	Method	Params(GB)	0-shot[8] ↑	Speedup ↑
Mixtral-8x7B	Baseline	93.41	72.64	1.00
	QESC	18.98	72.21	1.54
	QESC+PESF	18.98	71.68	1.68
Phi3.5-moe	Baseline	83.75	69.62	1.00
	QESC	17.08	68.49	1.55
	QESC+PESF	17.08	68.31	1.75
Deepseek-moe -16b-base	Baseline	32.75	61.38	1.00
	QESC	7.19	61.22	1.39
	QESC+PESF	7.19	61.09	1.55
Qwen1.5-MoE -A2.7B	Baseline	28.63	64.72	1.00
	QESC	6.69	62.89	1.36
	QESC+PESF	6.69	62.73	1.58

6.4 Experiment on Quantization + Pruning

Finally, we apply our QESC and PESF methods together to comprehensively compress MoE-LLMs. To achieve a reasonable trade-off between reducing memory usage, inference speed, and maintaining model performance, we apply a relatively mild dynamic pruning strategy ($\alpha = 0.3$) on top of 3.03-bit static quantization. We report the memory usage, average accuracy on zero-shot tasks, and inference speedup measured by the context latency for a batch of 4 sentences of length 512 in Table 4.

Maintain Accuracy. With the aid of effective expert selection calibration, our method limits the average accuracy loss across four models to within 1.25%, effectively maintaining the accuracy of the compressed MoE models.

Memory Saving and Inference Efficiency. By leveraging the BitBLAS tool (Wang et al., 2024) to store quantized weights and efficiently handle mixed-precision BLAS operations on GPUs, we limit the memory usage of Mixtral-8x7B and Phi3.5-moe to within 19GB, and that of Deepseek-moe-16b-base and Qwen1.5-MoE-A2.7B to within 7.2GB. This optimization enables deployment on a single RTX 3090 GPU while achieving an average speedup of 1.49× under 3.03-bit quantization. Furthermore, by integrating efficient dynamic expert pruning, we attain an average actual inference speedup of 1.64× across all four models.

Comparion with MC-MoE. To the best of our knowledge, MC-MoE (Huang et al., 2024a) is currently the only method that leverages both static quantization and dynamic pruning for MoE-LLMs, providing specific implementations for 2.06-bit and 2.56-bit quantization and pruning on Mixtral-8x7B. Therefore, we compare our method with MC-MoE at the corresponding quantization bit-widths on the same model and adopt a more conservative pruning strategy in PESF ($\alpha = 0.3$). As shown in Table 5, our method outperforms MC-MOE in terms of PPL, average accuracy on zero-shot tasks, and actual inference speedup under both quantization settings.

Table 5: Comprehensive comparison of average accuracy on 8 zero-shot tasks and inference speedup of four models under quantization and pruning.

Bits	Method	Mixtral-8x7B PPL ↓	0-shot[8] ↑	Speedup ↑
16.00	Baseline	3.84	72.64	1.00
2.06	MC-MoE	5.51	62.56	1.80
	EAC-MoE (ours)	**5.14**	**65.90**	**1.82**
2.56	MC-MoE	4.74	68.65	1.71
	EAC-MoE (ours)	**4.58**	**68.60**	**1.74**

More Results of EAC-MoE. Additionally, we perform more detailed experiments by combining other quantization bit-widths and more aggressive pruning strategies across all four models. Detailed results can be found in Appendix A.10.

6.5 Ablation Study of Loss Type

We compare the average accuracy on 0-shot tasks after calibration using TopK-MSE and MSE loss on three MoE models with a larger number of experts (the search for the optimal k-values in shown in Appendix A.4). As shown in Table 6, the calibrated models optimized with TopK-MSE demonstrate significantly better performance, proving the effectiveness of our optimization method.

Table 6: The impact of different loss types on the average accuracy of the calibrated model on 0-shot tasks (under 2.06-bit quantization).

Models	Loss Type	PPL ↓	0-shot[8] ↑
Phi3.5-moe	MSE	5.33	64.52
	TopK-MSE	5.22	65.03
Deepseek-moe -16b-base	MSE	8.16	55.91
	TopK-MSE	7.99	57.05
Qwen1.5-MoE -A2.7B	MSE	9.02	58.44
	TopK-MSE	8.30	59.52

7 Conclusion

In this work, we aim to address the challenges faced by MoE-LLMs and the limitations of existing compression methods. Focusing on expert selection, a key characteristic of MoE-LLMs, we propose a compression method specifically designed for MoE-LLMs that combines static quantization and dynamic pruning to enhance their deployment efficiency. Our methods significantly reduce memory usage and improve inference speed while maintaining high model performance.

8 Expert Pruning for MoE-LLMs

Recently we have observed that expert pruning for MoE-LLMs has emerged as a prominent area of research. Based on this, we aim to offer a relatively comprehensive overview of related studies for the reference of other researchers.

First, we narrow the scope to post-training expert pruning, which can generally be divided into two main categories: static expert pruning and dynamic expert pruning. Below, we present the latest advancements under each category:
(1) static expert pruning: To the best of our knowledge, (Lu et al., 2024) was the first to propose using a calibration set to determine the usage frequency of each expert in MoE and to evaluate expert importance based on their usage frequency. Experts with lower usage frequency are pruned prior to inference. Building on this, (Xie et al., 2024) applied knowledge distillation after pre-inference expert pruning to restore the performance of the pruned MoE model. Similar to (Lu et al., 2024) but not identical, (Muzio et al., 2024) leveraged the cumulative scores from the router's softmax output to assess expert importance for pre-inference pruning. Meanwhile, (Liu et al., 2024) introduced an expert merging approach, where less frequently used experts are merged with others to maintain the overall performance of the pruned MoE.
(2) dynamic expert pruning: Research on post-training dynamic expert pruning is relatively limited. (Lu et al., 2024) also proposed a dynamic pruning method based on the weight differences between experts selected for each token. Specifically, when the weight of the top-1 selected expert exceeds that of the top-2 expert by a certain threshold, only the top-1 expert is used. (Huang et al., 2024a) further introduced a critical token protection mechanism in dynamic pruning.

Limitations

Our method can significantly reduce memory consumption and improve inference speed while maintaining the performance of MoE models. However, there are still certain limitations to our approach:
(1) The proposed dynamic pruning method (PESF) calculates expert selection frequencies based on the current input sequence and determines the experts to prune accordingly. This method is only applicable during the prefill stage of model inference but is not suitable for the generate stage, where only a single token is input at a time. In the future, we aim to explore an MoE model pruning method that considers both inference phases, enabling inference acceleration benefits for the whole inference phase.
(2) We validated the effectiveness of our method on two MoE models with approximately 50B parameters and two MoE models with approximately 15B parameters. However, due to limited computational resources, we have not yet tested our method on larger-scale MoE models. For example, a recent significant breakthrough in the MoE field is the open release of DeepSeek-V3 (DeepSeek-AI et al., 2024), which have a total of 671B parameters and 37B active parameters. Deepseek-V3 demonstrates comprehensive performance that even match or surpass some leading closed-source models(OpenAI et al., 2024). However, its enormous parameter count poses significant challenges for practical deployment. In the future, we will continue to explore quantization and pruning techniques for larger-scale MoE models, aiming to contribute to the advancement of MoE.
(3) The quantization method we propose, QESC, is theoretically orthogonal to other existing approaches that primarily focus on reducing quantization error itself, such as (Frantar et al., 2022; Wang et al., 2018; Ashkboos et al., 2024b; Wang et al., 2020; Shao et al., 2023). However, in our experiments, apart from GPTQ, we have not yet evaluated the performance of our QESC method when combined with these techniques. In future work, we hope to further investigate the overall effectiveness of integrating our quantization method with these established approaches. Additionally, similar to (Huang et al., 2024c; Xiao et al., 2023b), we also aspire to validate our method across a broader range of benchmarks.

Ethics Statement

In this work, we analyze the expert selection preferences of Mixture-of-Experts (MoE) models not only across three common types of NLP tasks—QA/CR, Math, and Code—but also including datasets in specific languages, with French being the selected language for the latter category.

We explicitly emphasize that the use of French in this study does not imply any bias, preference, or discriminatory intent towards or against any specific language, culture, or group of people. The selection is made solely to illustrate that MoE models exhibit different expert selection preferences across datasets in different languages.

Acknowledgments

This work was supported in part by the National Key RD Program of China (No.2022ZD0160304), the Strategic Priority Research Program of Chinese Academy of Sciences(No.XDA0480200), the Beijing Municipal Science and Technology Project (No.Z231100010323002), and the Beijing Natural Science Foundation (No.L244046).

References

2019. Winogrande: An adversarial winograd schema challenge at scale.

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Anonymous. 2024. OLMoe: Open mixture-of-experts language models. In *Submitted to The Thirteenth International Conference on Learning Representations*. Under review.

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024a. SliceGPT: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations*.

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

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Loubna Ben Allal, Niklas Muennighoff, Logesh Kumar Umapathi, Ben Lipkin, and Leandro von Werra. 2022. A framework for the evaluation of code generation models. `https://github.com/bigcode-project/bigcode-evaluation-harness`.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, and et al. Greg Brockman. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

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 *NAACL*.

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:1803.05457v1*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, and et al. Chong Ruan. 2024. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. GPTQ: Accurate post-training compression for generative pretrained transformers. *arXiv preprint arXiv:2210.17323*.

Trevor Gale, Deepak Narayanan, Cliff Young, and Matei Zaharia. 2023. Megablocks: Efficient sparse training with mixture-of-experts. In *Proceedings of Machine Learning and Systems*, volume 5, pages 288–304. Curan.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Song Han, Huizi Mao, and William J Dally. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *ICLR*.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. 2021a. Measuring coding challenge competence with apps. *NeurIPS*.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021b. Aligning ai with shared human values. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021c. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Wei Huang, Yue Liao, Jianhui Liu, Ruifei He, Haoru Tan, Shiming Zhang, Hongsheng Li, Si Liu, and Xiaojuan Qi. 2024a. Mc-moe: Mixture compressor for mixture-of-experts llms gains more. *arXiv preprint arXiv:2410.06270*.

Wei Huang, Yangdong Liu, Haotong Qin, Ying Li, Shiming Zhang, Xianglong Liu, Michele Magno, and Xiaojuan Qi. 2024b. BiLLM: Pushing the limit of post-training quantization for LLMs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 20023–20042. PMLR.

Wei Huang, Xingyu Zheng, Xudong Ma, Haotong Qin, Chengtao Lv, Hong Chen, Jie Luo, Xiaojuan Qi, Xianglong Liu, and Michele Magno. 2024c. An empirical study of llama3 quantization: From llms to mllms. *Visual Intelligence*, 2(1):36.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Michael I. Jordan and Robert A. Jacobs. 1994. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6(2):181–214.

Young Jin Kim, Ammar Ahmad Awan, Alexandre Muzio, Andres Felipe Cruz Salinas, Liyang Lu, Amr Hendy, Samyam Rajbhandari, Yuxiong He, and Hany Hassan Awadalla. 2021. Scalable and efficient moe training for multitask multilingual models. *arXiv preprint arXiv:2109.10465*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 3843–3857. Curran Associates, Inc.

Pingzhi Li, Xiaolong Jin, Yu Cheng, and Tianlong Chen. 2024a. Examining post-training quantization for mixture-of-experts: A benchmark. *arXiv preprint arXiv:2406.08155*.

Shiyao Li, Xuefei Ning, Luning Wang, Tengxuan Liu, Xiangsheng Shi, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. 2024b. Evaluating quantized large language models. *Preprint*, arXiv:2402.18158.

Enshu Liu, Junyi Zhu, Zinan Lin, Xuefei Ning, Matthew B. Blaschko, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. 2024. Efficient expert pruning for sparse mixture-of-experts language models: Enhancing performance and reducing inference costs. *Preprint*, arXiv:2407.00945.

Xudong Lu, Qi Liu, Yuhui Xu, Aojun Zhou, Siyuan Huang, Bo Zhang, Junchi Yan, and Hongsheng Li. 2024. Not all experts are equal: Efficient expert pruning and skipping for mixture-of-experts large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6159–6172, Bangkok, Thailand. Association for Computational Linguistics.

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

Alexandre Muzio, Alex Sun, and Churan He. 2024. Seer-moe: Sparse expert efficiency through regularization for mixture-of-experts. *CoRR*, abs/2404.05089.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, and et al. Shyamal Anadkat. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernandez. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

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. *Journal of machine learning research*, 21(140):1–67.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

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*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Lei Wang, Lingxiao Ma, Shijie Cao, Quanlu Zhang, Jilong Xue, Yining Shi, Ningxin Zheng, Ziming Miao, Fan Yang, Ting Cao, Yuqing Yang, and Mao Yang. 2024. Ladder: Enabling efficient low-precision deep learning computing through hardware-aware tensor transformation. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*, pages 307–323, Santa Clara, CA. USENIX Association.

Peisong Wang, Qiang Chen, Xiangyu He, and Jian Cheng. 2020. Towards accurate post-training network quantization via bit-split and stitching. In *International Conference on Machine Learning*, pages 9847–9856. PMLR.

Peisong Wang, Qinghao Hu, Yifan Zhang, Chunjie Zhang, Yang Liu, and Jian Cheng. 2018. Two-step quantization for low-bit neural networks. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pages 4376–4384.

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 *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38087–38099. PMLR.

Yisong Xiao, Aishan Liu, Tianyuan Zhang, Haotong Qin, Jinyang Guo, and Xianglong Liu. 2023b. Robustmq: benchmarking robustness of quantized models. *Visual Intelligence*, 1(1):30.

Yanyue Xie, Zhi Zhang, Ding Zhou, Cong Xie, Ziang Song, Xin Liu, Yanzhi Wang, Xue Lin, and An Xu. 2024. Moe-pruner: Pruning mixture-of-experts large language model using the hints from its router. *Preprint*, arXiv:2410.12013.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Pengcheng Yin, Bowen Deng, Edgar Chen, Bogdan Vasilescu, and Graham Neubig. 2018. Learning to mine aligned code and natural language pairs from stack overflow. In *2018 IEEE/ACM 15th international conference on mining software repositories (MSR)*, pages 476–486. IEEE.

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*.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase Adversaries from Word Scrambling. In *Proc. of NAACL*.

Aojun Zhou, Ke Wang, Zimu Lu, Weikang Shi, Sichun Luo, Zipeng Qin, Shaoqing Lu, Anya Jia, Linqi Song, Mingjie Zhan, and Hongsheng Li. 2024. Solving challenging math word problems using gpt-4 code interpreter with code-based self-verification. In *ICLR*.

Michael H. Zhu and Suyog Gupta. 2018. To prune, or not to prune: Exploring the efficacy of pruning for model compression.

A Appendix

A.1 Time Consumption Analysis

QESC. The quantization process of QESC primarily consists of two parts: GPTQ and the calibrating router, and can be executed on a single A100 40G GPU. We record the time spent on the GPTQ process and the calibration of routers separately, and calculate their respective proportions of the total time. As shown in Table 7, GPTQ accounts for the vast majority of the total quantization time, while the calibration of routers takes an average of only 1.94% of the overall process. This demonstrates that our method introduces only a minimal additional time overhead to the quantization process, making it well-suited for practical applications.

PESF. Our PESF method introduces only a single-step online computation, as shown in Equation (6), resulting in virtually no additional delay.

Table 7: Time consumption analysis for GPTQ and router calibration in our QESC method.

Models	Step of QESC	Time(h)	Proportion(%)
Mixtral-8x7B	GPTQ	1.30	98.48
	Calibrating Router	0.02	1.52
Phi3.5-moe	GPTQ	1.39	97.89
	Calibrating Router	0.03	2.11
Deepseek-moe -16b-base	GPTQ	1.75	97.22
	Calibrating Router	0.05	2.78
Qwen1.5-MoE -A2.7B	GPTQ	1.48	98.67
	Calibrating Router	0.02	1.33

A.2 Performance on Challenging Tasks

In addition to validating the performance of our QESC method in maintaining model performance through Perplexity (PPL) and accuracies on common-sense intelligence tasks, we further evaluate our approach on more challenging mathematical task GSM8K (Cobbe et al., 2021) and code generation task HumanEval (Chen et al., 2021). We compare our method with three other methods—GPTQ, BSP, and PMQ. These two evaluations are conducted within the Bigcode-Evaluation-Harness (Ben Allal et al., 2022) testing framework.

As shown in Table 8, under three different average quantization bit widths, our QESC method significantly outperforms the other three methods in preserving the performance of the post-quantization model on two challenging tasks. Notably, despite prior studies (Li et al., 2024b) indicating that post-quantization models are often highly sensitive to complex mathematical and coding tasks, we manage to limit the performance degradation of Mixtral-8x7B on GSM8K to within 3% under 3.03-bit quantization, highlighting the effectiveness of our expert-selection calibration method.

Table 8: Comparison of the performance on challenging tasks GSM8K and HumanEval on Mixtral-8x7B. In the HumanEval evaluation, the hyperparameters are set as follows: temperature = 0.2, and nsamples = 10.

Models	Method	GSM8K	HumanEval (pass@10)
16.00	Full Precision	58.30	59.15
2.06	GPTQ	33.97	21.13
	PMQ	20.17	11.91
	QESC	**37.15**	**26.83**
2.54	GPTQ	38.97	27.24
	BSP	40.00	31.62
	PMQ	36.94	29.77
	QESC	**43.14**	**33.54**
3.03	GPTQ	52.29	40.83
	BSP	53.72	42.07
	QESC	**55.34**	**46.34**

A.3 Overfitting Analysis of Mixed-Precision Quantization Methods for MoE Models

In Section 3.3, we observe that the importance of the same expert may vary drastically across different tasks. Based on this insight, we deduce that using any calibration set to determine the importance of an expert before inference may be biased and lack generalization. Here, we further substantiate this point through comparative experiments. Specifically, we first utilize QA/CR datasets, Math datasets, Code datasets, datasets in French version, and the C4 dataset (Raffel et al., 2020) as calibration sets, obtaining five distinct expert selection frequencies from these calibration sets. Subsequently, we automatically allocate the quantization bit-width for each expert based on the expert selection frequencies, following the algorithm mentioned in the state-of-the-art mixed-precision quantization method for MoE, PMQ (Huang et al., 2024a) (details is shown in Appendix A.6). We then quantize the model to an average bit-width of 2.06 bits using the five quantization bit-width settings derived (the calibration set used in the paper is the C4 dataset). Finally, we evaluate the performance of these five quantized models on four datasets: Hellswag (QA/CR), MathQA (Math), Lambada_fr (French), and Conala (Code), each representing a different task category. Additionally, we compare the performance of our QESC method at the same average quantization bit-width.

As shown in Table 9, the mixed-precision quantization method based on expert usage frequency exhibits significant overfitting on both models. As highlighted by the red-marked values in the table,

Table 9: A comparison of the performance of quantized Mixtral-8x7B and Deepseek-moe-16b-base, based on the mixed-precision quantization method PMQ, using five different calibration sets under an average quantization bit-width of 2.06 bits, across four types of task datasets.

Models	Bits	Method	Calibration Dataset	Hellaswag (QA/CR)	MathQA (Math)	Lambada_fr (French)	Conala (Code)
Mixtral-8x7B	16.00	Baseline	None	84.03	41.64	65.96	73.86
	2.06	Mixed-Precision	QA/CR	75.49	31.06	49.17	7.25
			Math	69.25	32.26	47.06	7.64
			French	65.55	24.39	51.87	3.80
			Code	67.67	31.66	44.50	37.42
			C4	74.95	31.79	49.12	16.22
	2.06	**QESC**	None	**77.27**	**36.08**	**60.37**	**66.68**
Deepseek-moe -16b-base	16.00	Baseline	None	77.43	31.66	58.08	58.76
	2.06	Mixed-Precision	QA/CR	72.09	24.25	45.76	7.57
			Math	62.15	30.52	41.74	8.69
			French	65.49	23.15	48.86	1.80
			Code	54.87	29.61	35.88	26.34
			C4	68.70	26.26	42.13	16.22
	2.06	**QESC**	None	**68.85**	**28.91**	**52.38**	**42.36**

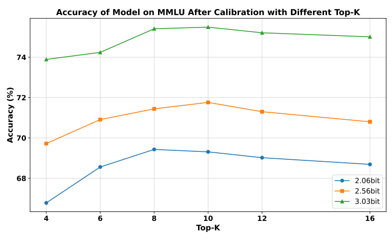
using a specific task dataset as the calibration set results in a post-quantization model that achieves relatively optimal performance on that specific task but shows severe performance degradation on other tasks. This overfitting phenomenon is most pronounced in Code-related tasks. When calibration is performed using datasets from other task types, the quantized models experience a drastic performance collapse on Code tasks. Only the models calibrated with Code-specific task datasets manage to achieve relatively good results on the Conala dataset.

When the C4 dataset is used as the calibration set, due to its inherently balanced nature, the quantized models achieve relatively average performance across all four task types. However, even in this case, there is still severe performance degradation on Code-related tasks. In contrast, our proposed QESC method focuses on the calibration of expert-selection. It significantly outperforms mixed-precision method across all four datasets, demonstrating superior generalization capabilities.

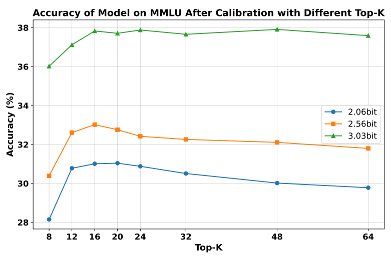
A.4 Search of Optimal Value of K in TopK-MSE Loss

In this subsection, we explore the optimal k-value settings for TopK-MSE in three MoE models with relatively larger numbers of total experts: Phi3.5-moe, Deepseek-moe-16b-base, and Qwen1.5-moe-A2.7B, under three average quantization bit widths—2.06 bits, 2.56 bits, and 3.03 bits. For Phi3.5-moe (selecting 2 experts out of 16), we set the k-values to 4, 6, 8, 10, 12, and 16 (equivalent to MSE loss). For Deepseek-moe-16b-base (selecting 6 experts out of 64) and Qwen1.5-moe-A2.7B (selecting 4 experts out of 64), the k-values are set to 8, 12, 16, 20, 24, 32, 48, and 64 (equivalent to MSE loss). For each k-value, we quantize the MoE models to the three average bit widths while keeping other configurations constant and compare their performance on the MMLU dataset (Hendrycks et al., 2021b), which serves as a comprehensive benchmark to effectively evaluate the models' capabilities across diverse tasks.

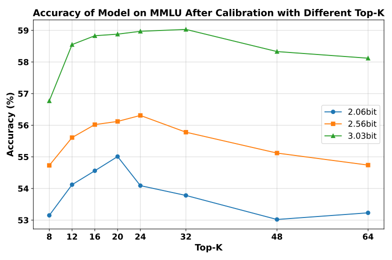
As shown in Figure 8, when the k-value is too small and approaches the number of experts selected per token in the model, significant performance degradation occurs. Referring to Figure 4, this may be attributed to a certain degree of overfitting. For Phi3.5-moe, the optimization results with k-values of 8–10 under all three average quantization bit widths are noticeably better than those with a k-value of 16. For Deepseek-moe-16b-base and Qwen1.5-moe-A2.7B, the optimal results are observed at k-values of 16–24 under 2.06-bit and 2.56-bit quantization. This aligns with our observations in Figure 4, where our optimization effectively covers over 95% of the incorrectly unselected experts while avoiding the noise introduced by experts with low selection probabilities. Under the 3.03-bit quantization, however, the optimization results show minimal variation with changes in k-value, as the higher quantization bit width in-



(a) Phi3.5-moe

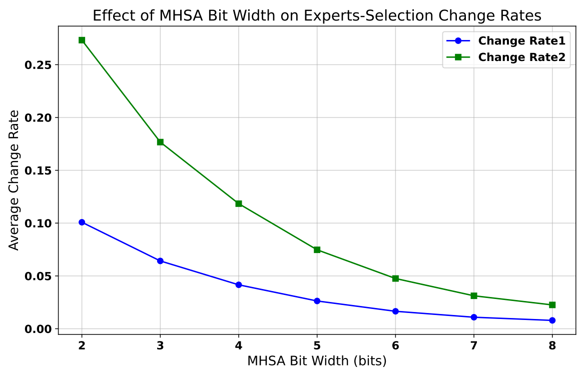


(b) Deepseek-moe-16b-base



(c) Qwen1.5-MoE-A2.7B

Figure 8: The accuracy of Phi3.5-moe (left), Deepseek-moe-16b-base (middle), and Qwen1.5-MoE-A2.7B (right) on the MMLU dataset varies under different TopK-MSE optimization under 2.06bit, 2.56bit and 3.03bit quantization.



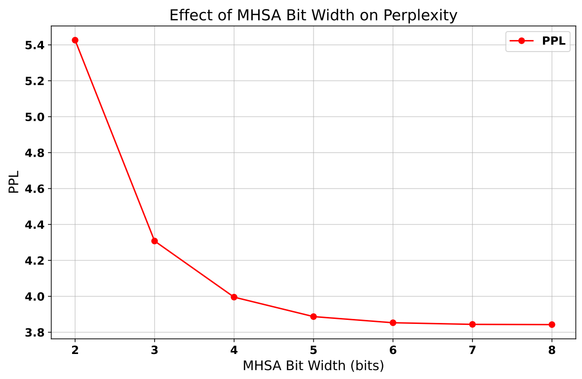


Figure 9: The changes in Experts-Selection rates and WikiText2 perplexity (PPL) with varying MHSA quantization bit-widths on Mixtral-8x7B. "Change Rate 1" corresponds to cases where both selected experts are changed, while "Change Rate 2" corresponds to cases where one or more expert selections are altered.

herently results in a lower rate of expert selection changes.

Based on the above observations, as shown in Table 10, we set the k-values to 8, 20, and 20 for expert selection calibration for the three models, respectively. All quantization-related experimental results in this work are based on this configuration.

Table 10: K Values for expert-selection in MoE models using TopK-MSE calibration

Model	Total Experts	Experts per Token	K
Phi3.5-moe	16	2	8
Deepseek-moe-16b-base	64	6	20
Qwen1.5-MoE-A2.7B	64	4	20

Table 11: Proportion of non-embedding parameters in MoE models

Model	MHSA (%)	Experts (%)	Router (%)
Mixtral-8x7B	2.894%	97.104%	0.002%
Phi3.5-moe	3.226%	96.774%	0.005%
Deepseek-moe-16b-base	2.852%	97.122%	0.022%
Qwen1.5-MoE-A2.7B	2.945%	97.039%	0.022%

A.5 Quantization Bit-Width Settings

In MoE-LLMs, the vast majority of non-embedding parameters come from the experts within the MoE layers, while the router in the MoE layer and MHSA account for only about 3% of the total parameters. For the four MoE models used in the experiments of this paper—Mixtral-8x7B, Deepseek-moe-16b-base, Phi3.5-moe, and Qwen1.5-MoE-A2.7B—we first conduct a detailed analysis of the parameter proportions in each component, as shown in Table 11. Considering that the router accounts for less than 0.03% of the total parameters, yet plays a crucial role in expert selection, this work retains the router at its original precision.

Unlike the experts in the MoE layer, the MHSA component affects all tokens in the input sequence. Previous studies (Li et al., 2024a) have compared the impact of increasing the quantization bit-width of MHSA and the MoE layer on overall model performance, concluding that MHSA is more bit-efficient. In this work, we investigate the effect of MHSA quantization bit-width on overall model performance from the perspective of expert selection change rate. Using Mixtral-8x7B, we conduct the following experiments: keeping the rest of the model at its original precision, the MHSA component is quantized to bit-widths ranging from 2-bit to 8-bit. We then calculate the average expert selection change rate relative to the original model and the perplexity (PPL) of the quantized model on the WikiText2 (Merity et al., 2016).

As shown in Figure 9, within the 2-4 bit range, both the expert selection change rate and PPL are highly sensitive to the quantization bit-width. As the bit-width increases, the expert selection change rate and PPL decrease significantly. This demonstrates that maintaining MHSA at a relatively high quantization bit-width is indeed important for ensuring the quantized model can still select the correct experts and maintain overall model performance. Considering that in the 4-8 bit range, both metrics change more gradually with increasing bit-width, and from a hardware perspective, current systems are unable to achieve efficient acceleration for bit-widths like 5-bit or 6-bit, we choose to set the MHSA quantization bit-width to 4-bit. This choice strikes a balance between maintaining the performance of the quantized model and minimizing overall memory usage.

By comprehensively analyzing the above discussion, we quantize the MHSA section to 4-bit, retaine the router at its original precision, and quantize the experts to 2-bit, 2.5-bit, and 3-bit. Under the 2.5-bit setting, we align with previous research (Li et al., 2024a), which finds that the earlier layers in MoE benefit from higher quantization bit-widths. Therefore, we simply quantize the experts in the first half of the layers to 3-bit and those in the second half to 2-bit. Based on this configuration, we calculate the average bit-width of the experts for the four models under the 2-bit, 2.5-bit, and 3-bit conditions, as shown in Table 12. Since the parameter proportions of different components vary slightly across the four models, the final average bit-width also exhibits minor differences. However, as these differences are negligible, for simplicity, we represent the average quantization bit-widths of the four models in this paper as 2.06-bit, 2.54-bit, and 3.03-bit for the three respective settings.

Table 12: The average quantization bit-width of the four models when the experts' quantization bit-width is set to 2-bit, 2.5-bit, and 3-bit.

Model	**2-bit**	**2.5-bit**	**3-bit**
Mixtral-8x7B	2.058	2.544	3.029
Phi3.5-moe	2.065	2.549	3.033
Deepseek-moe-16b-base	2.060	2.546	3.031
Qwen1.5-MoE-A2.7B	2.062	2.547	3.032

A.6 Reproduction Details of Quantization

BSP (Li et al., 2024a) and PMQ (Huang et al., 2024a) are two methods built upon GPTQ that focus on mixed-precision quantization. BSP reports its results on Mixtral-8x7B and Deepseek-moe-16b-base in its paper, while PMQ only reports results on Mixtral-8x7B. To ensure a fair comparison, we refer to both the papers and official repositories of these methods, and apply their respective quantization bit-width allocation strategies to the models used in this study. We then perform GPTQ quantization and evaluate the final results using the same framework as this paper.

Specifically, for BSP, we first use the same calibration datasets mentioned in its paper (WikiText2 dataset) to obtain the expert usage frequencies for Phi3.5-moe and Qwen1.5-MoE-A2.7B. In Phi3.5-moe, each MoE layer selects 2 experts out of 16. Following BSP's settings for Mixtral-8x7B, at the 3.03-bit bit-width configuration, we allocate 4-bit to the top-8 most frequently used experts and 2-bit to the remaining 8 experts. At the 2.54-bit configuration, we allocate 3-bit to the top-8 experts and 2-bit to the remaining 8 experts. For Qwen1.5-MoE-A2.7B, each MoE layer selects 4 experts out of 60, with an additional 4 shared experts. Following BSP's settings for Deepseek-moe-16b-base, all shared experts are allocated 8-bit. At the 3.03-bit configuration, we allocate 4-bit to the top-20 most frequently used experts and 2-bit to the remaining 40 experts. At the 2.54-bit configuration, we allocate 4-bit to the top-6 experts and 2-bit to the remaining 54 experts.

For PMQ, we also use the same calibration datasets mentioned in its paper (C4 dataset (Raffel et al., 2020)) to obtain expert usage frequencies for all four models. For Mixtral-8x7B and Phi3.5-moe, we directly apply the Integer Programming (IP) optimization algorithm used in PMQ's paper to derive the corresponding mixed-precision bit-width configurations. For Deepseek-moe-16b-base and Qwen1.5-MoE-A2.7B, as PMQ's paper did not discuss MoE models with shared experts, we use the same IP optimization algorithm to determine the mixed-precision bit-width configurations for non-shared experts under the average bit-width settings

Table 13: Comprehensive comparison of average accuracy of Mixtral-8x7B on 8 zero-shot tasks under quantization.

Bits	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
16.00	Full Precision	83.57	83.67	59.3	85.35	84.03	76.24	41.64	67.29	72.64
2.06	GPTQ	76.16	75.47	47.91	77.40	71.57	71.98	31.84	48.14	62.56
	PMQ	79.16	73.06	48.38	80.58	74.95	71.27	31.79	46.80	63.25
	EAC-MoE	79.16	76.30	51.02	76.79	77.27	74.82	36.08	59.01	66.31
2.54	GPTQ	80.56	79.04	56.06	85.84	78.20	73.01	37.62	58.89	68.65
	BSP	80.96	77.86	53.24	72.53	77.95	73.56	35.34	52.04	65.44
	PMQ	80.52	77.10	51.28	82.54	79.03	73.95	39.18	56.37	67.50
	EAC-MoE	81.45	80.77	55.03	84.85	79.44	75.3	37.86	61.37	69.51
3.03	GPTQ	80.63	79.21	55.55	84.59	79.63	74.43	37.49	59.82	68.92
	BSP	81.56	79.71	54.59	75.75	80.06	74.74	36.68	54.69	67.22
	EAC-MoE	82.97	83.25	58.87	85.47	82.77	76.40	41.81	66.10	72.21

Table 14: Comprehensive comparison of average accuracy of Phi3.5-moe on 8 zero-shot tasks under quantization.

Bits	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
16.00	Full Precision	77.69	65.70	53.58	88.35	79.87	76.80	38.32	76.62	69.62
2.06	GPTQ	75.57	57.87	48.46	86.88	73.01	71.72	32.80	69.29	64.45
	PMQ	75.68	50.59	41.55	86.18	78.45	73.86	22.31	62.14	61.35
	EAC-MoE	76.17	59.47	49.06	86.80	73.34	71.35	34.64	69.43	65.03
2.54	GPTQ	77.37	61.49	48.98	86.67	75.32	72.61	32.86	71.19	65.81
	BSP	76.99	61.41	50.71	87.22	76.09	73.51	32.23	71.05	66.15
	PMQ	77.04	57.32	48.21	87.68	77.42	74.27	33.67	72.66	66.03
	EAC-MoE	77.09	61.78	51.19	86.57	75.56	73.32	35.08	71.67	66.53
3.03	GPTQ	78.10	63.17	51.96	87.70	78.81	74.59	35.64	75.01	68.12
	BSP	77.20	64.60	51.71	87.49	78.73	76.01	33.23	72.40	67.67
	EAC-MoE	78.24	62.21	52.56	87.83	78.59	76.16	36.85	75.49	68.49

of 2.06-bit and 2.54-bit. For shared experts, we allocate 2-bit at the 2.06-bit configuration and 3-bit at the 2.54-bit configuration to ensure fairness in comparative experiments with PMQ.

A.7 Completed Results of Quantization

In Tables 13 to 16, we present the full results of Table 2, including detailed accuracies on eight zero-shot tasks, and compare our approach with GPTQ (Frantar et al., 2022), BSP (Li et al., 2024a) and PMQ (Huang et al., 2024a).

A.8 Reproduction Details of Pruning

EES (Lu et al., 2024) and ODP (Huang et al., 2024a) are two popular dynamic pruning methods for MoE models, both focusing on ignoring the least-contributing experts for each input token. ODP extends EES by incorporating a significance-aware token protection mechanism.

For EES, we use the same calibration dataset as in its paper to compute the ratio between the weight of the least-contributing expert and the weight of the most-contributing expert for each token. The median of all these ratios is selected as the pruning threshold. During inference, if the ratio of the least-contributing expert's weight to the most-contributing expert's weight for a given token is less than this threshold, the weight of the least-contributing expert is set to zero.

For ODP, we follow the same procedure as EES to determine the pruning threshold. On top of this, we incorporate the Significance-Aware Token Protection mechanism mentioned in the paper. This mechanism dynamically identifies critical tokens and prevents the pruning of the least-contributing experts for these critical tokens, even if they meet the original pruning condition.

A.9 Completed Results of Pruning

In Table 17, we present the full results of Table 3. We compare our approach with several other methods, including EES (Lu et al., 2024) and ODP (Huang et al., 2024a).

A.10 Completed Results of QESC+PESF

In this subsection, leveraging the three quantization bit-width settings from QESC and the two pruning threshold strategies from PESF, we provide a detailed evaluation of different combinations on four models. The results include the average zero-shot

Table 15: Comprehensive comparison of average accuracy of Deepseek-moe-16b-base on 8 zero-shot tasks under quantization.

Bits	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
16.00	Full Precision	80.52	73.19	47.53	72.57	77.43	69.93	31.66	38.18	61.38
2.06	GPTQ	76.77	65.78	40.02	67.34	69.41	65.43	27.50	26.76	54.88
	PMQ	77.64	64.73	39.85	67.75	68.7	66.22	26.26	27.16	54.79
	EAC-MoE	77.48	70.24	42.66	70.20	68.85	67.01	28.91	31.04	57.05
2.54	GPTQ	78.29	68.48	41.72	69.88	70.97	67.48	28.17	29.65	56.83
	BSP	78.89	70.24	42.32	74.34	72.96	68.59	27.87	30.68	58.24
	PMQ	78.78	69.70	41.89	74.29	71.27	67.96	29.25	32.89	58.00
	EAC-MoE	78.51	69.11	42.32	74.59	73.97	69.14	28.98	33.02	58.71
3.03	GPTQ	78.89	72.22	44.37	72.42	75.15	67.01	29.88	34.72	59.33
	BSP	79.68	72.39	44.37	73.98	74.61	68.90	28.14	33.10	59.40
	EAC-MoE	80.03	73.15	45.90	75.29	75.68	70.48	31.42	37.83	61.22

Table 16: Comprehensive comparison of average accuracy of Qwen1.5-MoE-A2.7B on 8 zero-shot tasks under quantization.

Bits	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
16.00	Full Precision	80.79	69.44	44.37	79.57	77.17	69.77	35.57	61.08	64.72
2.06	GPTQ	75.79	65.53	40.02	72.14	67.06	64.48	30.02	47.07	57.76
	PMQ	76.14	65.69	40.11	69.88	68.71	64.52	29.01	48.23	57.79
	EAC-MoE	78.40	65.28	40.78	70.37	71.50	65.98	29.28	54.56	59.52
2.54	GPTQ	77.48	63.17	39.16	68.69	70.42	64.17	29.28	50.88	57.91
	BSP	79.54	65.03	39.59	68.99	73.77	68.51	31.89	55.91	60.40
	PMQ	78.16	65.75	41.21	71.34	72.34	68.01	31.66	55.31	60.47
	EAC-MoE	78.73	66.50	43.17	73.09	73.53	68.19	32.43	56.12	61.47
3.03	GPTQ	79.27	66.16	41.55	77.89	75.49	67.8	31.66	57.87	62.21
	BSP	79.68	65.11	42.32	70.73	75.36	66.38	30.79	56.65	60.88
	EAC-MoE	80.47	67.30	41.89	77.99	75.79	69.22	31.46	58.97	62.89

Table 17: Comprehensive comparison of average accuracy of four models on 8 Zero-Shot tasks under pruning.

Model	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
Mixtral-8x7B	bf16	83.57	83.67	59.30	85.35	84.03	76.24	41.64	67.29	72.64
	EES	82.70	81.40	58.45	87.74	83.00	75.24	41.14	64.50	71.40
	ODP	82.87	83.21	59.01	84.98	83.69	75.97	40.93	65.17	71.98
	PESF ($\alpha = 0.3$)	83.19	83.29	59.90	85.08	83.40	76.40	40.90	65.36	72.19
	PESF ($\alpha = 0.7$)	68.23	65.15	45.99	78.10	69.39	64.80	29.25	44.86	58.22
Phi3.5-moe	bf16	77.69	65.70	53.58	88.35	79.87	76.80	38.32	76.62	69.62
	EES	76.50	63.85	51.37	86.94	78.60	75.14	36.25	75.02	67.96
	ODP	77.23	64.94	53.55	88.10	79.21	74.93	38.27	75.11	68.92
	PESF ($\alpha = 0.3$)	77.04	65.53	54.01	88.20	79.82	75.30	37.99	76.30	69.27
	PESF ($\alpha = 0.7$)	75.52	64.94	51.54	86.91	79.51	73.16	37.76	74.24	67.95
Deepseek-moe-16b-base	bf16	80.52	73.19	47.53	72.57	77.43	69.93	31.66	38.18	61.38
	EES	79.98	72.85	48.21	72.51	77.33	70.11	29.88	38.33	61.15
	ODP	80.01	72.94	47.41	73.39	77.25	70.14	30.23	38.14	61.19
	PESF ($\alpha = 0.3$)	80.52	73.02	46.93	72.72	77.13	70.32	31.52	38.07	61.28
	PESF ($\alpha = 0.7$)	78.45	73.15	45.90	75.29	75.68	68.51	31.49	34.83	60.41
Qwen1.5-MoE-A2.7B	bf16	80.79	69.44	44.37	79.57	77.17	69.77	35.57	61.08	64.72
	EES	80.52	69.11	44.01	79.45	76.87	69.32	35.11	60.97	64.42
	ODP	80.61	69.24	44.12	79.51	77.01	69.32	35.05	60.95	64.48
	PESF ($\alpha = 0.3$)	80.74	69.28	44.28	79.36	77.24	69.77	35.68	60.80	64.64
	PESF ($\alpha = 0.7$)	80.25	69.11	43.94	77.06	76.44	70.09	35.38	58.72	63.87

accuracy and inference speedup ratio, along with specific accuracy on each dataset.

Detailed results. Here, we provide detailed results of our method under three quantization bit-widths and two pruning thresholds, as shown in Table 18. Furthermore, as shown in Tables 19 to 22, we report the specific results for various configurations of the four models on each dataset. "EAC-MoE"

Table 18: Comparison of the average accuracy on 8 zero-shot tasks and speedup of inference across four MoE models under quantization (QESC) and pruning (PESF). "EAC-MoE" represents the use of conservative PESF pruning strategy ($\alpha = 0.3$) based on QESC, "EAC-MoE*" represents the use of PESF ($\alpha = 0.7$) based on QESC.

Bits	Method	Mixtral-8x7B		Phi3.5-moe		Deepseek-moe-16b-base		Qwen1.5-MoE-A2.7B	
		0-shot[8] ↑	Speedup ↑	0-shot[8] ↑	Speedup ↑	0-shot[8] ↑	Speedup ↑	0-shot[8] ↑	Speedup ↑
16.00	Baseline	72.64	1.00	69.62	1.00	61.38	1.00	64.72	1.00
2.06	EAC-MoE	65.90	1.82	65.26	1.84	56.75	1.59	61.18	1.64
	EAC-MoE*	49.09	1.93	62.16	2.03	54.81	2.01	60.02	2.08
2.54	EAC-MoE	68.60	1.74	65.98	1.80	58.17	1.56	61.41	1.60
	EAC-MoE*	53.01	1.81	63.80	2.02	56.83	1.98	60.57	2.03
3.03	EAC-MoE	71.68	1.68	68.31	1.75	61.09	1.55	62.73	1.58
	EAC-MoE*	57.55	1.76	66.66	1.98	58.82	1.96	61.77	2.01

Table 19: Detailed results of average accuracy of Mixtral-8x7B on 8 Zero-Shot tasks under quantization and pruning.

Bits	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
16.00	Full Precision	83.57	83.67	59.3	85.35	84.03	76.24	41.64	67.29	72.64
2.06	EAC-MoE	79.76	77.78	50.85	77.28	75.80	73.24	35.88	56.60	65.90
	EAC-MoE*	61.04	54.42	38.23	69.54	48.15	61.09	24.69	35.58	49.09
2.54	EAC-MoE	81.28	80.13	54.44	83.85	78.66	74.27	37.39	58.78	68.60
	EAC-MoE*	65.29	58.63	39.25	75.05	58.27	61.56	27.10	38.92	53.01
3.03	EAC-MoE	83.03	82.91	58.28	85.35	82.17	76.87	41.47	63.35	71.68
	EAC-MoE*	69.98	64.01	42.04	81.92	67.24	63.22	29.98	42.01	57.55

Table 20: Detailed results of average accuracy of Phi3.5-moe on 8 Zero-Shot tasks under quantization and pruning.

Bits	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
16.00	Full Precision	77.69	65.70	53.58	88.35	79.87	76.80	38.32	76.62	69.62
2.06	EAC-MoE	74.16	61.28	48.55	86.64	75.41	73.56	33.10	69.35	65.26
	EAC-MoE*	72.80	59.01	49.40	83.27	72.74	69.30	32.09	58.68	62.16
2.54	EAC-MoE	75.73	60.61	48.72	86.64	76.84	72.69	35.44	71.18	65.98
	EAC-MoE*	73.01	58.59	48.38	84.37	74.10	68.67	34.44	68.82	63.80
3.03	EAC-MoE	76.93	62.38	52.39	88.13	78.38	76.01	36.65	75.57	68.31
	EAC-MoE*	74.65	61.28	50.68	86.24	77.77	72.77	36.75	73.17	66.66

represents the use of conservative PESF pruning strategy ($\alpha = 0.3$) based on QESC, while "EAC-MoE*" represents the use of a more aggressive PESF pruning strategy ($\alpha = 0.7$) based on QESC.

A.11 Task-Preference and Sparsity in Experts-Selection

In Section 3.3, we calculate the expert selection frequency and pairwise cosine similarity of MoE models across different datasets for four types of reasoning tasks, providing a macroscopic view of the task preferences in expert selection. In this section, we delve deeper into the microscopic details, specifically discussing the expert selection preferences of the Phi3.5-moe and Deepseek-moe-16b-base models across 8 datasets spanning 4 task types. This analysis also aligns with previous work (Li et al., 2024a), further demonstrating their sparsity. As shown in Figure 10, the four rows, from top to bottom, correspond to four different task types ((QA/CR), Math, Code, Specific Language). For each row, the left and right plots respectively show the expert selection frequencies of Phi3.5-moe on two different datasets of the same task type. From this, a clear pattern can be observed, indicating that the MoE model exhibits remarkably similar expert selection preferences across datasets within the same task category. For example, in the first row, as illustrated in the left and right subfigures, certain experts such as Expert13 in Layer2, Expert9 in Layer8, and Expert12 in Layer11 are frequently selected for the openbookqa and arc-challenge datasets, with average selection frequencies exceeding 30% (note that each layer has 16 experts, and a completely balanced selection would result in a frequency of 6.25% per expert). Conversely, some experts, such as Expert1 in Layer5, Expert7 in Layer14, and Expert8 in Layer27, are rarely selected, with frequencies below 1%.

Similarly, in the second row, for the Math

Table 21: Detailed results of average accuracy of Deepseek-moe-16b-base on 8 Zero-Shot tasks under quantization and pruning.

Bits	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
16.00	Full Precision	80.52	73.19	47.53	72.57	77.43	69.93	31.66	38.18	61.37
2.06	EAC-MoE	77.04	69.11	41.30	72.55	69.85	67.48	28.61	28.07	56.75
	EAC-MoE*	75.19	67.93	39.93	70.45	66.55	63.85	27.77	26.81	54.81
2.54	EAC-MoE	78.29	69.15	42.32	74.50	66.55	63.85	27.77	26.81	54.81
	EAC-MoE*	76.17	71.74	41.13	71.74	68.31	66.85	28.11	30.57	56.83
3.03	EAC-MoE	79.54	73.15	46.16	75.02	75.55	70.24	31.59	37.45	61.09
	EAC-MoE*	77.20	71.55	45.05	72.51	72.86	66.46	31.56	33.33	58.82

Table 22: Detailed results of average accuracy of Qwen1.5-MoE-A2.7B on 8 Zero-Shot tasks under quantization and pruning.

Bits	Method	PIQA	ARC-E	ARC-C	BOOLQ	HS	WG	MATHQA	MMLU	Avg
16.00	Full Precision	80.79	69.44	44.37	79.57	77.17	69.77	35.57	61.08	64.72
2.06	EAC-MoE	79.16	65.07	42.83	75.01	72.44	68.19	32.63	54.12	61.18
	EAC-MoE*	77.91	64.52	42.15	74.86	70.94	67.09	32.26	50.46	60.02
2.54	EAC-MoE	78.78	66.75	43.17	72.57	73.53	68.19	32.29	56.01	61.41
	EAC-MoE*	77.53	65.24	40.44	74.89	72.49	68.27	32.16	53.52	60.57
3.03	EAC-MoE	80.41	66.92	42.15	76.88	75.81	69.22	31.72	58.72	62.73
	EAC-MoE*	79.16	66.71	41.30	75.35	74.97	68.75	31.99	55.96	61.77

task across two datasets, experts such as Expert7 in Layer5, Expert9 in Layer8, and Expert11 in Layer15 are selected with an average frequency exceeding 40%, while others, such as Expert0 in Layer0 and Expert8 in Layer2, are seldom chosen. These results provide detailed evidence that Phi3.5-moe demonstrates a high degree of similarity in expert selection frequencies within task categories while also exhibiting significant sparsity. From another perspective, Phi3.5-moe demonstrates entirely distinct expert selection preferences across different reasoning tasks. For instance, Expert13 in Layer2 is frequently selected for (QA/CR) tasks but is neither prominent nor frequently chosen in the other three tasks. Similarly, Expert7 in Layer14 is heavily utilized in Code tasks but is rarely selected in the other three task categories.

A similar pattern is observed in Deepseek-moe-16b-base, which has 64 experts per layer, as shown in Figure 11. While displaying clear intra-category similarities and inter-category differences in expert selection, the larger number of experts results in an even greater degree of sparsity in expert selection for Deepseek-moe-16b-base.

A.12 Pruning on Mixtral-8x7B

In Section 6.3, when employing a more aggressive pruning strategy, unlike the other three models which maintain relatively stable average accuracy under significant inference speedup, Mixtral-8x7B exhibits notable performance degradation. This section delves into this phenomenon and analyzes its underlying causes.

Similar to Figure 12, we plot the changes in Mixtral-8x7B's average accuracy, expert pruning rate, and inference speedup as the pruning threshold varied. As shown in the figure, unlike Phi3.5-moe and Deepseek-moe-16b-base, where a significant drop in accuracy only occurs when the pruning threshold exceeded 0.7 and the expert pruning rate approached 40%, Mixtral-8x7B begins to show a noticeable decline in accuracy once the pruning threshold surpassed 0.3.

We further analyze the expert selection frequency of Mixtral-8x7B across two different datasets. As illustrated in Figure 13, compared to Phi3.5-moe and Deepseek-moe-16b-base, Mixtral-8x7B exhibits weaker sparsity in expert selection. Apart from a few experts, such as Expert6 in Layer3 (top) and Expert2 in Layer25 (bottom), whose average selection frequencies exceed the mean (0.125), the selection frequencies of the remaining experts are relatively balanced. This phenomenon has also been noted in (Jiang et al., 2024; Li et al., 2024a). Consequently, Mixtral-8x7B is more sensitive to dynamic expert pruning compared to the other three models, making it less suitable for the aggressive pruning settings ($\alpha = 0.7$) proposed in our PESF method. Nevertheless, our approach achieved commend-

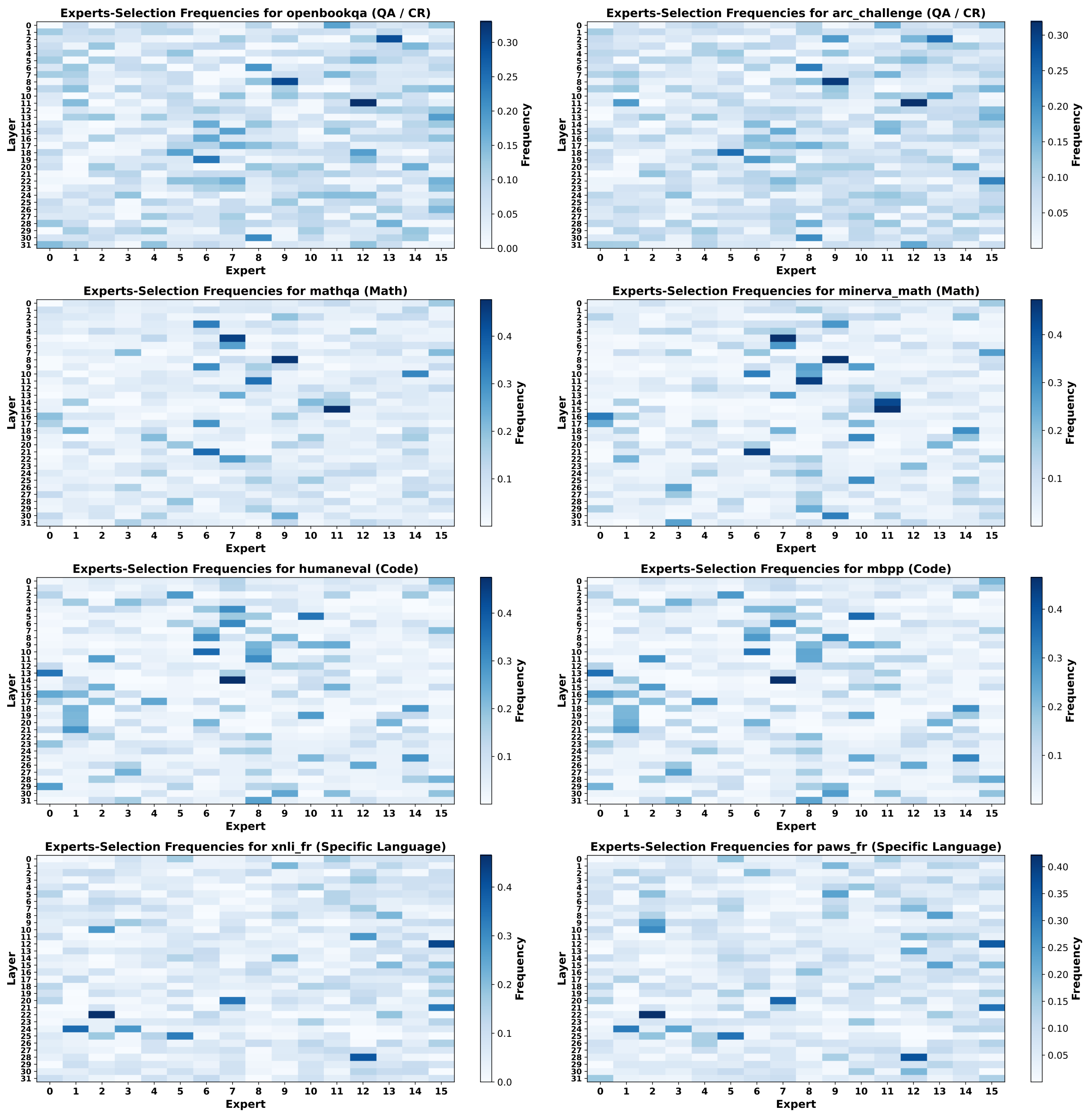


Figure 10: The frequency of expert selection across 8 datasets spanning 4 task types for Phi3.5-moe.

able inference speedup and accuracy retention on Mixtral-8x7B under conservative pruning settings ($\alpha = 0.3$) . In the future, we aim to explore methods to achieve higher pruning rates and speedup while maintaining accuracy on Mixtral-8x7B.

A.13 Datasets used in Expert-Selection Analysis

In Section 3.3, we record Expert-Selection frequency on 15 datasets of three common categories of NLP tasks——Math, Code-Generation, Question-Answering or Commonsense-Reasoning (QA/CR) and 4 datasets in French language. GSM8K (Cobbe et al., 2021), MathQA (Amini et al., 2019), Minerva_Math (Lewkowycz et al., 2022) and Hendrycks_Math (Hendrycks et al., 2021c) are in Math task; Winogrande (ai2, 2019), PIQA (Bisk et al., 2020), ARC-Challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), MathQA (Amini et al., 2019), HellaSwag (Zellers et al., 2019), and Social_iqa (Sap et al., 2019) are in QA/CR task; Humaneval (Chen et al., 2021), Mbpp (Austin et al., 2021), Apps (Hendrycks et al., 2021a) and Conala (Yin et al., 2018) are in code task; Lambada_fr (Paperno et al., 2016), Xnli_fr (Conneau et al., 2018), Paws_fr (Zhang et al., 2019) and Arc_fr (Clark et al., 2018) are datasets in French language.

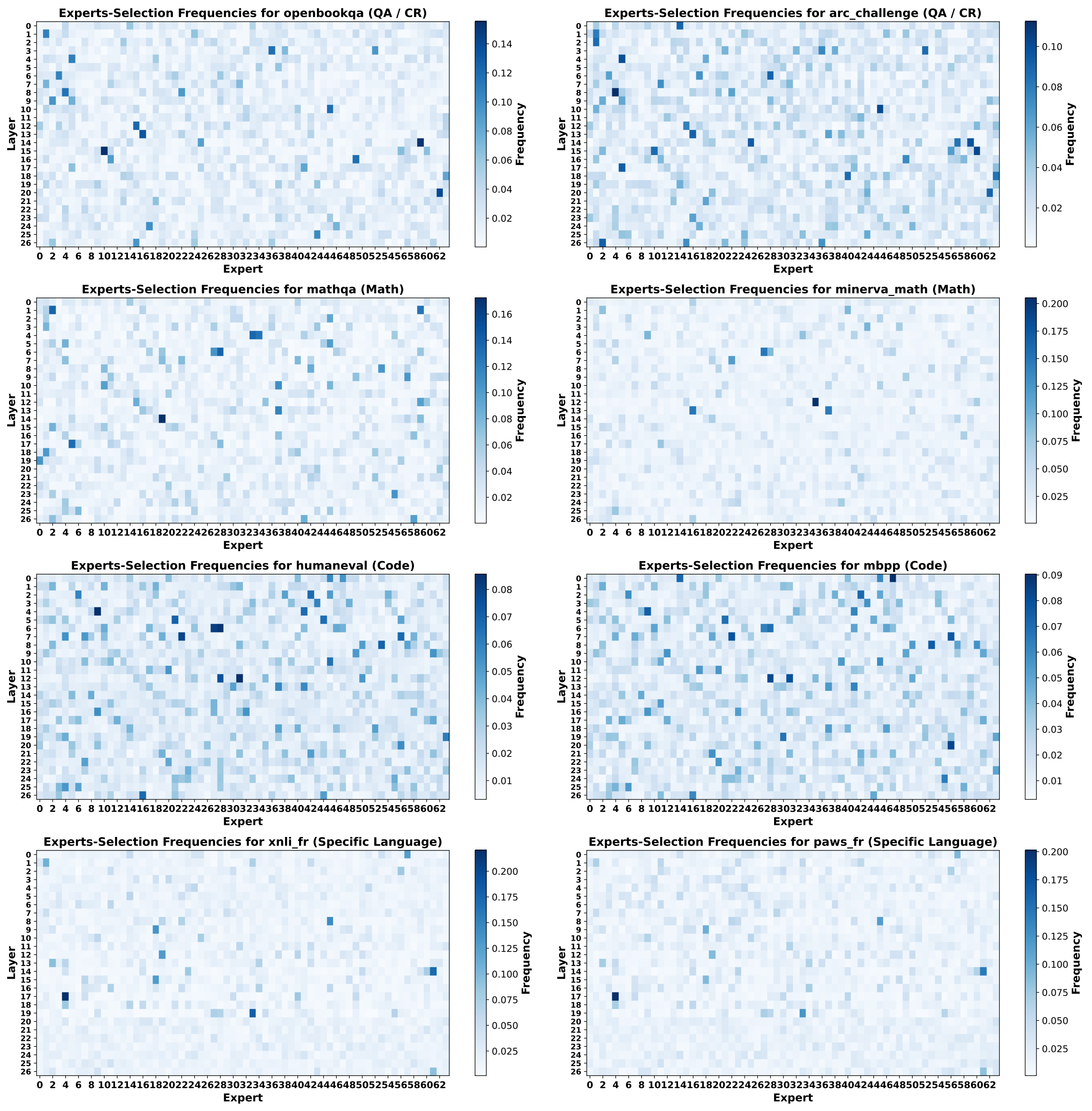


Figure 11: The frequency of expert selection across 8 datasets spanning 4 task types for Deepseek-moe-16b-base

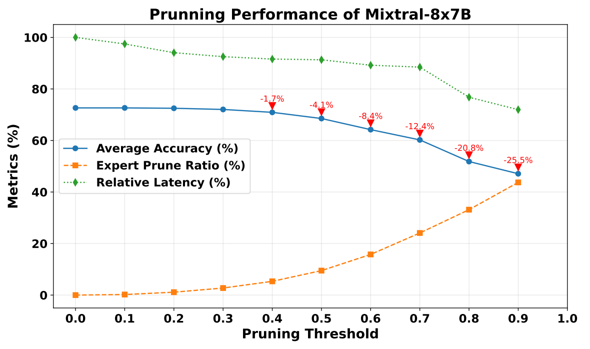


Figure 12: The variations in the model's average accuracy, expert pruning rate, and inference acceleration effect with respect to changes in the pruning threshold for Mixtral-8x7B.

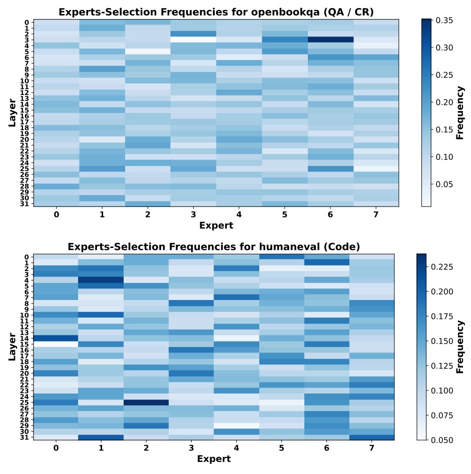


Figure 13: The frequency of expert selection on openbookqa and humaneval for Mixtral-8x7B.