MxQBound: Efficient Mixed-Precision Quantization for MoE-LLMs Guided by Expert Importance and Error Bound Guarantees

Anonymous authors
Paper under double-blind review

Abstract

While mixed-precision quantization reduces the memory overhead of Mixture-of-Experts (MoE)-Large Language Models (LLMs), existing methods often misestimate expert importance, causing suboptimal precision allocation and severe performance degradation under aggressive quantization. We propose **MxQBound**, an efficient and robust **Mix**ed-precision **Q**uantization framework for MoE-LLMs, guided by holistic assessment of expert importance and equipped with quantization error **Bound** guarantees. With only a single forward pass, MxQBound directly assesses each expert's importance from its output norms, supported by a provable upper bound on quantization error. Guided by the importance assessment, MxQBound jointly optimizes the per-expert bit-widths of weights and activations to minimize quantization errors under an overall bit-allocation budget. It further adapts each expert's activation bit-width according to its input strength during inference with negligible overhead. Extensive experiments on diverse MoE-LLMs and benchmarks show that, under comparable compression ratios, MxQBound outperforms state-of-the-art (SOTA) methods by 4.08-22.85% in model performance and 2.2-141× in computational efficiency. Under aggressive W4A4, MxQBound incurs an average accuracy drop of only 15.07% relative to the full-precision baseline, whereas SOTA methods collapse to near-zero accuracy.

1 Introduction

Mixture-of-Experts (MoE) architectures scale parameter capacity of Large Language Models (LLMs) without proportionally increasing per-token computation. However, all experts typically reside in memory during inference, making memory overhead a major deployment bottleneck. Low-bit quantization is therefore crucial for scalable and memory-efficient deployment of MoE-LLMs.

Post-training quantization has been widely used for compressing dense LLMs (Frantar et al., 2023; Lin et al., 2024; Xiao et al., 2023; Ashkboos et al., 2024). These methods typically adopt a layer- or group-level strategy that assigns a shared quantization configuration to each layer or group. In sparsely routed MoE-LLMs, however, each token activates only a subset of experts, whose activation statistics and quantization sensitivities can differ substantially (Li et al., 2024). As a result, these methods are ill-suited to determining the optimal allocation of quantization precision across experts.

This has motivated mixed-precision quantization methods for MoE-LLMs, which allocate distinct precisions to experts according to their importance. Specifically, they either rely on routing norm statistics (e.g., routing frequency) (Chowdhury et al., 2026; Duanmu et al., 2025) or use token-expert affinity calibration (e.g., routing weight) (Chen et al., 2025) as importance metrics to guide precision allocation. However, as highlighted in Sections 2.2 and 2.3, neither frequency nor weight reliably matches actual expert importance: the former exhibits significant variance across different calibrations, failing to provide a stable and reliable assessment of expert importance, whereas the latter is primarily geared towards load-balanced routing, rather than reflecting the experts' actual contribution to the output. Under aggressive quantization regimes (e.g., W4A4), this misalignment can induce a systematic bias in bit allocation, ultimately leading to catastrophic model collapse.

Given that some neural network parameters can be pruned with little performance degradation, parameter importance assessment has evolved from second-order criteria (LeCun et al., 1990; Hassibi

Code is publicly available at https://anonymous.4open.science/r/MxQBound.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

& Stork, 1993), to first-order and data-free criteria (Lee et al., 2019; Wang et al., 2020; Tanaka et al., 2020), and more recently to efficient column-level importance approximations (Frantar & Alistarh, 2023; Sun et al., 2024). Unfortunately, these methods operate at the granularity of individual parameters. Since each expert in an MoE model consists of multiple layers interconnected by non-linear activation functions (e.g., SiLU), the non-linearities disrupt holistic importance evaluation, making it impossible to derive expert-level importance. Moreover, as activation and weight quantization differ substantially while remaining interdependent, the traditional importance assessment methods cannot be directly applied to MoE quantization, as detailed in Section 2.4. This raises a key question:

How can we assess fine-grained expert importance to enable cost-effective and robust quantization of MoE-LLMs, with marginal performance loss even under highly aggressive quantization?

To address this question, we propose **MxQBound**, an efficient and robust **Mix**ed-precision **Q**uantization framework for MoE-LLMs, guided by holistic assessment of expert importance and equipped with quantization error **Bound** guarantees. Specifically, the main contributions include:

- **Efficient and theoretically grounded assessment of expert importance with a provable quantization error bound.** We propose a metric that directly assesses expert-level importance from expert outputs collected in a single forward pass, thereby avoiding the disruption of holistic importance evaluation caused by non-linear activations. Guided by the metric, we estimate weight- and activation-quantization errors, and formally derive a theoretical upper bound on the errors.
- **An importance-aware and adaptive quantization pipeline.** Based on the importance metric, we propose a three-stage quantization pipeline for expert-level bit allocation. For activations, we reshape the product activation distribution to suppress outliers, and optimize the clipping threshold and bit-width for each expert, thereby minimizing quantization error under a given activation bit budget. For weights, we optimize the bit-width of each expert based on its importance and quantization error under a given weight bit budget. During inference, we adaptively adjust each expert's default activation bit-width according to its input strength, enabling input-dependent adaptation with negligible overhead.
- **Robust performance across extremely low to moderate budgets.** We evaluate MxQBound on diverse MoE-LLMs with parameter scales ranging from 7.6B to 80B, across benchmarks covering text generation quality and multiple downstream tasks. Under comparable compression ratios, MxQBound outperforms SOTA methods by 4.08-22.85% in model performance and 2.2-141× in computational efficiency. Under aggressive W4A4, MxQBound incurs an average accuracy drop of only 15.07% relative to the full-precision baseline, whereas SOTA methods collapse to near-zero accuracy.

2 Preliminaries and Motivations

2.1 The Goal of MoE-LLM Quantization

Consider a standard top-K MoE design in Transformer layer l. For each token, the router selects a subset $\mathcal{E}_K^{(l)} \subseteq \mathcal{E}^{(l)}$ of experts. Under the feed-forward design, the MoE output is

$$Y_{\text{MoE}}^{(l)} = \sum\nolimits_{e \in \mathcal{E}_K^{(l)}} s_e^{(l)} Y_e(x_e), \quad Y_e(x_e) = W_{\text{down},e}\big[\sigma(W_{g,e}x_e) \odot (W_{u,e}x_e)\big], \tag{1}$$

where x_e is the input hidden state of expert e, $s_e^{(l)}$ is the routing weight for expert e, and $Y_e(\cdot)$ denotes the output function of e. $W_{g,e}$, $W_{u,e}$ and $W_{\text{down},e}$ are the gate, up and down projection matrices of e, respectively. $\sigma(\cdot)$ denotes the SiLU activation and $\odot$ denotes the element-wise (Hadamard) product.

Given the total number of model parameters (denoted as N_e), the bit-width budgets for expert weights (denoted as B_w), and activations (denoted as B_a), we aim to find per-expert weight and activation bit-widths $\{b_e^w, b_e^a\}$ that minimize the expected quantization error subject to average-bit budgets:

$$\min_{\{b_e^w, b_e^a\}} \mathbb{E}\Big[\|Y_{\text{MoE}} - \hat{Y}_{\text{MoE}}\|_2^2\Big] \quad \text{s.t.} \quad \bar{b}^w \leqslant B_w,\ \bar{b}^a \leqslant B_a,$$

where $\bar{b}^w \triangleq \sum_e N_e b_e^w / \sum_e N_e$, and $\bar{b}^a$ is defined analogously. To minimize expert-level quantization error, bit-widths should be allocated in proportion to each expert's contribution to the model output. Otherwise, insufficient protection of critical experts may cause severe performance degradation. Therefore, the central challenge is how to reliably assess expert importance.

2.2 Routing-Frequency-Based Assessment of Expert Importance Is Unstable

Existing routing-frequency-based quantization strategies assume that frequency is stable in assessing expert importance. We examine this on Qwen1.5-MoE-A2.7B (Qwen Team, 2024) using five

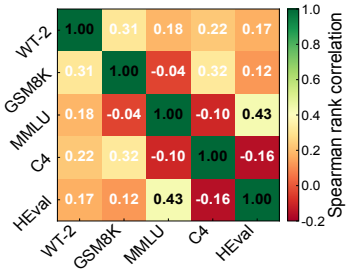


Figure 1: Routing-frequency-based assessment of expert importance varies across domain-diverse calibration datasets.

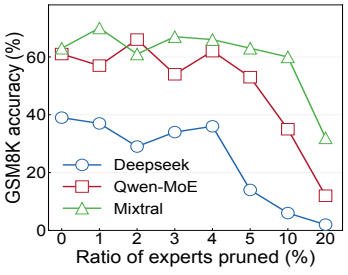


Figure 2: Pruning experts in descending order of routing weights reversely improves the GSM8K accuracy of the models.

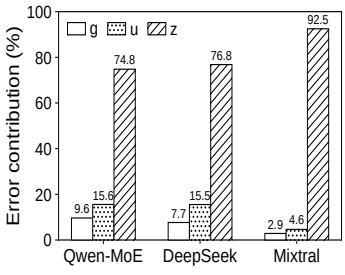


Figure 3: Decomposition of contributions on activation quantization error.

calibration datasets from diverse domains (i.e., WT-2 (Merity et al., 2017), GSM8K (Cobbe et al., 2021), MMLU (Hendrycks et al., 2021), C4 (Raffel et al., 2020) and HEval (Chen et al., 2021)), and compare the resulting frequency-based expert rankings. Specifically, for each dataset, we collect the routing frequency vectors of all 60 experts per layer. For every pair of calibration datasets, we then compute the Spearman rank correlation between their routing frequency vectors at each corresponding layer, and average the correlations across all layers. A higher correlation indicates that routing frequency provides a more consistent assessment of expert importance across the datasets.

The heatmap matrix of the calculated correlations is illustrated in Fig. 1. A warmer color for a square corresponding to a pair of calibration datasets indicates greater similarity between the routing-frequency-based expert rankings induced by the two datasets. We can see that the correlations are mostly very low ($\leqslant$0.43) and vary substantially across calibration datasets, indicating that the expert ranking is highly dependent on the domain of the calibration data, and therefore cannot serve as a stable metric of expert importance. Additional cross-calibration results are provided in Appendix B.

2.3 Routing-Weight-Based Assessment of Expert Importance Is Unreliable

Existing routing-weight-based quantization strategies assume that experts with larger routing weights are more important. To validate this assumption, we conduct expert-pruning experiments on DeepSeek-V2-Lite (DeepSeek-AI, 2024), Qwen1.5-MoE-A2.7B (Qwen Team, 2024) and Mixtral-8$\times$7B (Jiang et al., 2024), and evaluate their accuracy on the GSM8K dataset. Specifically, for each model, we first record the routing weight assigned to each expert whenever the expert is activated. Then, we compute the average routing weight over all the records for each expert. Subsequently, for each model, we rank all the experts in descending order of their average routing weights and prune the top-ranked 1%, 2%, 3%, 4%, 5%, 10% and 20% of the experts, respectively. Finally, we evaluate the GSM8K accuracy of each model at each pruning ratio, which is illustrated in Fig. 2.

We can see that as the pruning ratio increases, all models exhibit varying degrees of accuracy degradation on GSM8K. This trend is expected, as pruning more experts generally leads to greater performance loss. Interestingly, however, pruning the top-ranked 4%, 2% and 1% of experts in DeepSeek, Qwen and Mixtral, respectively, reversely improves the GSM8K accuracy of the corresponding models by nearly 10%. This means that a larger routing weight does not necessarily imply that an expert is more important to model performance. The underlying reason is that routing weights primarily reflect load-balancing behavior in token assignment and token-domain routing patterns, rather than an expert's actual contribution to model performance. As a result, they are unreliable indicators of expert importance. More analysis can be found in Appendix C.

2.4 Structural Analysis of Quantization Error in MoE-LLMs

According to Eq. (1), the MoE output can be decomposed into the following activations:

$$g = W_{g,e}x_e, \quad u = W_{u,e}x_e, \quad z = \sigma(g) \odot u, \quad Y_e(x_e) = W_{\text{down},e}z.$$

Since g and u undergo subsequent nonlinear activation function (i.e., $\sigma(\cdot)$) and element-wise multiplication (i.e., $\odot$), the quantization errors introduced into them may have trivial impact on the output. In contrast, the quantization error in z is propagated directly to the output through $W_{\text{down},e}$, and may therefore be the dominant contributor to the output error.

To validate this conjecture, we conduct a structural error analysis experiment in which only the activations of the MoE models are quantized, while the weights remain in full precision. Specifically,

for each expert in DeepSeek-V2-Lite, Qwen1.5-MoE-A2.7B and Mixtral-8×7B, we first uniformly quantize the g, u and z activations to 4 bits, and use their respective perplexities on the WT-2 dataset as the baselines. Then, we conduct an ablation study on each model, in which one activation is retained in full precision at a time, while the other two are quantized to 4 bits. Finally, for each model, we compute the error contribution of each of the g, u and z activations as the relative quantization error reduced by the corresponding ablation setting: Error contribution $= \frac{\text{Baseline quantization error - Ablation quantization error}}{\text{Baseline quantization error}}$.

The collected results are shown in Fig. 3. We can see that the error contribution of z is significantly higher than those of the other activations across all three models. This finding confirms that z is indeed the dominant source of quantization error. The underlying reason is that the element-wise multiplication $z = \sigma(g) \odot u$ tends to induce a heavy-tailed distribution with prominent outliers, resulting in its substantially higher sensitivity to quantization.

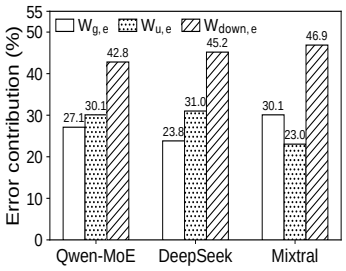


Figure 4: Contributions on weight quantization error.

Similarly, the error induced by weight quantization may also vary across the weight matrices (i.e., $W_{u,e}$, $W_{g,e}$ and $W_{\text{down},e}$). To validate this, we also quantize all three weight matrices per model to establish the baselines, and adopt a weight ablation setup analogous to that used in the analysis of activation quantization error. Fig. 4 shows the calculated error contributions of $W_{u,e}$, $W_{g,e}$ and $W_{\text{down},e}$, respectively, across the models. We can observe that quantizing $W_{\text{down},e}$ consistently contributes larger error than quantizing $W_{u,e}$ and $W_{g,e}$ across the models. Based on these observations, when deriving the quantization error and designing the quantization strategy, we explicitly prioritize the bit-width assignments of z and $W_{\text{down},e}$, as detailed in Section 3.3.1.

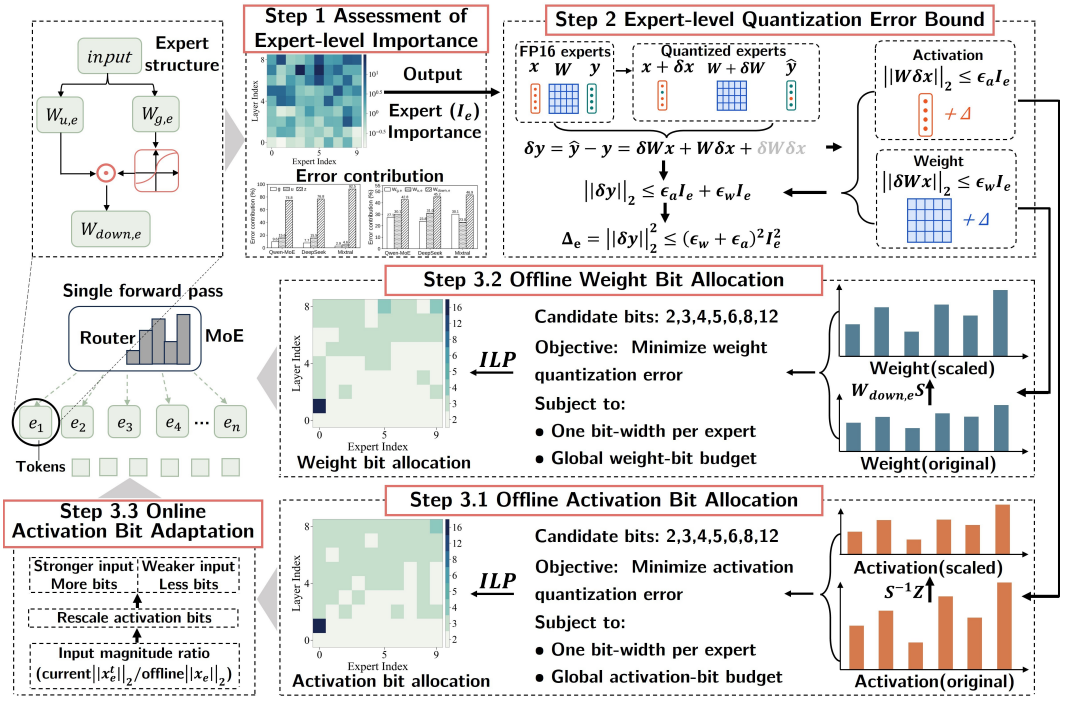


Figure 5: Workflow of MxQBound.

3 METHODOLOGY

As shown in Fig. 5, MxQBound consists of: **Step 1. Assessment of Expert-level Importance**. MxQBound assesses expert-level importance based on expert outputs using a single forward pass. **Step 2. Expert-level Quantization Error Bound**. MxQBound first estimates the relative errors introduced by weight and activation quantization per expert. Then, guided by the expert importance, it combines the estimated quantization errors to derive an upper bound on quantization error. **Step 3. Three-stage Quantization Pipeline**. Based on the error bound and expert importance, MxQBound

realizes expert-level bit allocation through: **Step 3.1 Offline Activation Bit Allocation.** MxQBound applies a scaling transformation to smooth activation outliers in z, and optimizes the clipping threshold and default activation bit-width per expert under a given activation bit budget. **Step 3.2 Offline Weight Bit Allocation.** MxQBound optimizes the weight bit-width per expert under a given weight bit budget. **Step 3.3 Online Activation Bit Adaptation.** MxQBound adaptively adjusts each expert's default activation bit-width according to its input strength at inference time.

3.1 Assessment of Expert-level Importance

As explained in Section 2.4, the multiplicative operation (i.e., $\sigma(W_{g,e}x_e) \odot (W_{u,e}x_e)$ in Eq. (1)) dominates the quantization error in MoE models and directly propagates the error to the output through $W_{\text{down},e}$. We therefore directly define expert importance in terms of the expert output:

$$I_e := \|Y_e(x_e)\|_2.$$

Specifically, the rationale is as follows. Let $a = \sigma(W_{g,e}x_e)$ and $b = W_{u,e}x_e$. Based on Eq. (1), by submultiplicativity of matrix norms and the Hadamard-product inequality, we have

$$\|Y_e(x_e)\|_2 = \|W_{\text{down},e}(a \odot b)\|_2 \leqslant \|W_{\text{down},e}\|_2 \|a \odot b\|_2 \leqslant \|W_{\text{down},e}\|_2 \|a\|_2 \|b\|_2.$$

Since $\|a\|_2 \leqslant \|W_{g,e}\|_2\|x_e\|_2$ and $\|b\|_2 \leqslant \|W_{u,e}\|_2\|x_e\|_2$, thus we can conclude that

$$\|Y_e(x_e)\|_2 \leqslant \|W_{\text{down},e}\|_2 \|W_{g,e}\|_2 \|W_{u,e}\|_2 \|x_e\|_2^2. \tag{2}$$

This means that $\|Y_e(x_e)\|_2$ is related to both the expert strength $\|W_{\text{down},e}\|_2\|W_{g,e}\|_2\|W_{u,e}\|_2$ and input magnitude $\|x_e\|_2^2$, which naturally characterizes the expert's holistic contribution to the output.

3.2 Expert-level Quantization Error Bound

Next, we formally analyze the error bound caused by expert-level quantization under the guidance of I_e. To simplify the analysis, we follow (Alizadeh et al., 2020) to locally approximate the output of an expert around a given full-precision input x using a first-order Taylor expansion: $y = Wx$, where W denotes the effective gain of the expert given the input x. Let $\hat{W}$ and $\hat{x}$ denote the quantized weight and input, and let δW and δx denote the corresponding quantization errors. Then

$$\hat{W} = W + \delta W, \qquad \hat{x} = x + \delta x,$$

Therefore, the output of the quantized expert network is

$$\hat{y} = \hat{W}\hat{x} = (W + \delta W)(x + \delta x) = Wx + \delta W x + W\delta x + \delta W \delta x. \tag{3}$$

Considering that the contribution of the second-order term $\delta W \delta x$ is trivial under the quantization settings considered (as elaborated in Appendix D), the output error can be approximated as

$$\delta y \approx \delta W x + W \delta x.$$

Then, we bound both terms using the expert output norm I_e. For the weight perturbation term $\delta W x$, we define the relative weight quantization error factor as $\epsilon_w := \frac{\|\delta W\|_2\|x\|_2}{\|Wx\|_2}$. Thus, we have

$$\|\delta W x\|_2 \leqslant \|\delta W\|_2 \|x\|_2 = \frac{\|\delta W\|_2\|x\|_2}{\|Wx\|_2}\|Wx\|_2 = \epsilon_w I_e.$$

For the activation perturbation term $W\delta x$, we define the relative activation error amplification factor as $\epsilon_a := \frac{\|W\|_2\|x\|_2}{\|Wx\|_2} \cdot \frac{\|\delta x\|_2}{\|x\|_2}$. Thus, we have

$$\|W\delta x\|_2 \leqslant \|W\|_2 \|\delta x\|_2 = \frac{\|W\|_2\|x\|_2}{\|Wx\|_2} \cdot \frac{\|\delta x\|_2}{\|x\|_2}\|Wx\|_2 = \epsilon_a I_e.$$

Combining the two inequalities yields the upper bound on the expert-level quantization error:

$$\Delta_e := \|\delta y\|_2^2 \leqslant (\epsilon_w + \epsilon_a)^2 I_e^2. \tag{4}$$

This means that Δ_e is jointly bounded by the relative quantization error factors (i.e., ϵ_w and ϵ_a) and expert importance (i.e., I_e), all of which should be considered in expert-level bit allocation.

3.3 Three-stage Quantization Pipeline

We further develop a three-stage quantization pipeline for expert-level bit allocation. Stage I (Section 3.3.1) optimizes the offline activation bit-widths after reshaping activation distribution, Stage II (Section 3.3.2) optimizes weight bit-widths by expert importance and quantization error, and Stage III (Section 3.3.3) adaptively adjusts activation bit-widths online according to input strength.

3.3.1 Offline Activation Bit Allocation

As illustrated in Section 2.4, the quantization of the product activation z is the dominant source of activation-side quantization error. This is because the wide dynamic range and prominent outliers of z result in large errors under aggressive low-bit quantization. To address this issue, we introduce a scaling matrix S_e for each expert to smooth the activation outliers in z, thereby transferring part of the quantization difficulty from z to $W_{\text{down},e}$ and lowering the risk of activation clipping. Specifically, we rewrite the expert mapping as

$$Y_e(x_e) = W_{\text{down},e} z = (W_{\text{down},e} S_e)(S_e^{-1} z), \tag{5}$$

where S_e is determined by following SmoothQuant (Xiao et al., 2023), of which details are elaborated in Appendix G.2. After scaling, we apply a Randomized Hadamard Transformation (RHT) to further suppress residual outliers. Since the scaled activations and weights are mathematically equivalent, this transformation preserves the full-precision output while rebalancing their dynamic range. Thus, following Eq. (4), the resulting expert-level quantization error is bounded by

$$\Delta_e \leqslant \left(\epsilon_w(b_e^w; S_e) + \epsilon_a(T_e, b_e^a; S_e)\right)^2 I_e^2, \tag{6}$$

where $\epsilon_w(b_e^w; S_e)$ denotes the relative weight quantization error associated with the weight bit-width b_e^w and scaling matrix S_e, and $\epsilon_a(T_e, b_e^a; S_e)$ denotes the relative activation quantization error associated with the clipping threshold T_e, activation bit-width b_e^a, and scaling matrix S_e. We define the activation-side error as:

$$\Delta_e^a(T_e, b_e^a) := \left\| W_{\text{down},e} S_e \left(Q_{T_e, b_e^a}(S_e^{-1} z) - S_e^{-1} z \right) \right\|_2^2 \leqslant \epsilon_a^2(T_e, b_e^a; S_e) I_e^2.$$

$Q_{T_e, b_e^a}(\cdot)$ denotes the quantization strategy with the clipping threshold T_e and default offline activation bit-width b_e^a. We obtain the optimal T_e and b_e^a by formulating and solving a 0-1 Integer Linear Program problem. Specifically, let $\mathcal{B}_a = \{b_{e,k}^a\}$ be the candidate set of activation bit-widths containing N_a elements, $\mathcal{T}_e = \{T_{e,n}\}$ be the candidate set of clipping thresholds containing N_T elements, and $x_{e,n,k}^a \in \{0,1\}$ be a binary variable indicating whether expert e selects $T_{e,n}$ and $b_{e,k}^a$, the offline activation bit allocation problem can be formulated as:

$$\min \quad \sum_{e \in \mathcal{E}^{(l)}} \sum_{n=1}^{N_T} \sum_{k=1}^{N_a} x_{e,n,k}^a \cdot \epsilon_a^2(T_{e,n}, b_{e,k}^a; S_e)\, I_e^2 \tag{7}$$

$$\text{s.t.} \quad \sum_{n=1}^{N_T} \sum_{k=1}^{N_a} x_{e,n,k}^a = 1, \qquad \forall e \in \mathcal{E}^{(l)}, \tag{8}$$

$$\sum_{e \in \mathcal{E}^{(l)}} \sum_{n=1}^{N_T} \sum_{k=1}^{N_a} x_{e,n,k}^a \cdot b_{e,k}^a \leqslant B_a. \tag{9}$$

Eq. (8) requires each expert to choose exactly one combination of T_e and b_e^a, while Eq. (9) imposes the total activation bit budget B_a. The optimal solution is denoted as $(T_e^\star, b_{e,\text{off}}^a)$.

3.3.2 Offline Weight Bit Allocation

According to Eq. (6), once $\epsilon_a(T_e^\star, b_{e,\text{off}}^a; S_e)$ is fixed, the offline weight bit allocation problem reduces to minimizing $\epsilon_w^2(b_e^w; S_e)\, I_e^2$. Specifically, let $\mathcal{B}_w = \{b_{e,k}^w\}$ be the candidate set of weight bit-widths containing N_w elements, $x_{e,k}^w \in \{0,1\}$ be a binary variable indicating whether expert e selects weight bit-width $b_{e,k}^w$, and N_e be the number of parameters in expert e. The problem can be formulated as:

$$\min \quad \sum_{e \in \mathcal{E}^{(l)}} \sum_{k=1}^{N_w} x_{e,k}^w \cdot \epsilon_w^2(b_{e,k}^w; S_e)\, I_e^2 \tag{10}$$

$$\text{s.t.} \quad \sum_{k=1}^{N_w} x_{e,k}^w = 1, \qquad \forall e \in \mathcal{E}^{(l)}, \tag{11}$$

$$\sum_{e \in \mathcal{E}^{(l)}} \sum_{k=1}^{N_w} x_{e,k}^w \cdot N_e\, b_{e,k}^w \leqslant B_w. \tag{12}$$

Eq. (11) requires each expert to choose exactly one weight bit-width, while Eq. (12) imposes the total weight bit budget B_w. The optimal solution is denoted as $b_{e,\text{off}}^w$.

3.3.3 Online Activation Bit Adaptation

Although Stage I assigns a default activation bit-width to each expert, activation statistics may still vary across tokens during inference. Stage III accounts for this variation by dynamically adjusting the activation bit-widths without re-solving the full allocation problem. Specifically, the key observation

is that each expert acts as a relatively stable amplifier of its input signal, so the input activation norm serves as a reliable proxy for the corresponding output activation. Accordingly, we rescale $b^a_{e,\text{off}}$ as:

$$b^a_e(t) = \frac{\|x_e^{(t)}\|_2}{\overline{\|x_e\|}_2} b^a_{e,\text{off}}, \tag{13}$$

where $\|x_e^{(t)}\|_2$ is the current input norm of expert e, and $\overline{\|x_e\|}_2$ is expert e's average baseline calculated offline during calibration. $\frac{\|x_e^{(t)}\|_2}{\overline{\|x_e\|}_2}$ reflects whether, and by how much, the current input strength has increased or decreased relative to its calibration mean. As a result, more bits will be assigned to stronger-than-usual inputs, while fewer bits will be assigned to weaker ones, enabling input-dependent adaptation at inference time with negligible overhead. Detailed validation of this scheme is in Appendix E. Finally, the algorithm of the quantization pipeline is elaborated in Appendix F.

4 Performance Evaluation

4.1 Experimental Setup

Models and datasets. We evaluate MxQBound on DeepSeek-V2-Lite (DS-V2), Mixtral-8×7B (Mixtral) and Qwen1.5-MoE-A2.7B (Qwen). More results on larger MoE-LLMs are in Appendix H.1. We evaluate text generation quality via perplexity (PPL) on WikiText-2 (WT2) (Merity et al., 2017) and C4 (Raffel et al., 2020). For downstream evaluation, we report exact match on GSM8K (Cobbe et al., 2021), Pass@1 on HumanEval (HEval) (Chen et al., 2021), normalized accuracy on HellaSwag (HSwag) (Zellers et al., 2019) and OpenBookQA (OBQA) (Mihaylov et al., 2018), and accuracy on MathQA (Amini et al., 2019), MMLU (Hendrycks et al., 2021) and BoolQ (Clark et al., 2019).

Baselines. For Weight-and-Activation (W+A) quantization, we compare MxQBound with RTN (Round-to-Nearest) using uniform bit allocation, GPTQ (Frantar et al., 2023), MxMoE (Duanmu et al., 2025) and CodeQuant (Yin et al., 2026). GPTQ and MxMoE determine linear-block-level mixed-precision assignments via ILP, whereas CodeQuant combines learnable rotation with codebook-based weight clustering to suppress outliers. For weight-only quantization, we compare MxQBound with GPTQ and MoEQuant (Chen et al., 2025), with detailed evaluation results provided in Appendix H.2.

Implementation details. We implement the methods under three W+A configurations: W4A4, W4A8 and W5A5, and one weight-only configuration W4A16, with average weight-activation bit-width budgets $(B_w, B_a) \in \{(4,4),(4,8),(5,5),(4,16)\}$. Only the MoE blocks are quantized, the attention modules remain in full precision. More implementation details are provided in Appendix G.

4.2 Impact on Model Performance

Table 1 summarizes the quantization performance of different methods across varying W+A quantization configurations. Specifically, under the aggressive W4A4 quantization regime, MxQBound achieves average accuracy scores of 50.53, 50.65 and 58.40 on DS-V2, Qwen and Mixtral, outperforming the best competing method (CodeQuant) by 4.08%, 22.85% and 22.6%, respectively. The advantage is particularly pronounced on GSM8K and HEval, where most methods collapse to near-zero accuracy, whereas MxQBound incurs an average accuracy drop of only 15.28% relative to the FP16 results across the three models. Under W4A8 and W5A5, MxQBound remains the best in average accuracy score, incurring only 1.11-3.12% accuracy drop compared with the FP16 results. As for text generation quality, MxQBound achieves the lowest PPL in almost all cases on both WT2 and C4, with the only exception on DS-V2 under W5A5, where its PPL is merely 0.08 higher than the best one. These advantages primarily stem from incorporating expert importance and the quantization error bound into bit allocation. By further mitigating activation outliers at inference time, MxQBound adaptively reduces the overall quantization error under a fixed bit budget.

4.3 Compression Cost-Effectiveness

Quantization efficiency. We further measure the total time consumed from collecting calibration statistics to completing quantization, including both activation and weight bit-width allocation, as shown in Table 2. Across different models and quantization settings, MxQBound completes the entire procedure within 15-52 minutes, yielding speedups of 1.6-4.4×, 20-139.2× and 2.2-141.6× over GPTQ, MxMoE and CodeQuant, respectively. The inefficiency of the competing methods primarily stems from the additional computation required to assess expert importance, whether via layer-wise Hessian information calculation (GPTQ), multi-round forward evaluation (MxMoE), or iterative fine-tuning of rotation matrices (CodeQuant). In contrast, MxQBound assesses expert-level

Table 1: Comparison of impact on model performance. Model performance under Full Precision (FP) is provided for reference. ↓ means lower perplexity is better, and ↑ means higher accuracy is better.

Conf	Model	Method	PPL (↓)		Accuracy score (↑)							
			WT2	C4	GSM8K	MMLU	HEval	BoolQ	HSwag	OBQA	MathQA	Avg
FP16	DS-V2	FP	5.72	8.74	41.12	58.09	29.90	79.88	77.99	46.00	41.21	53.46
	Qwen	FP	6.22	9.30	63.91	59.81	34.15	79.82	77.24	44.00	37.45	56.63
	Mixtral	FP	3.45	6.63	61.26	70.42	40.85	86.20	83.89	48.40	44.00	62.15
W4A4	DS-V2	RTN	15.18	24.21	3.34	40.63	7.32	62.97	67.79	38.60	29.41	35.72
		GPTQ	15.85	24.88	3.30	8.34	7.30	63.30	67.80	38.60	29.40	31.15
		MxMoE	8.96	15.45	12.89	44.27	10.98	57.92	72.96	42.40	30.35	38.82
		CodeQuant	6.69	9.89	31.77	51.30	23.78	75.81	**76.23**	**42.80**	38.16	48.55
		MxQBound	**6.05**	**9.31**	**36.32**	**55.76**	**24.42**	**78.62**	76.16	42.60	**39.80**	**50.53**
	Qwen	RTN	4.87×10^4	4.23×10^4	0.38	25.17	0.00	41.87	34.10	34.10	30.00	23.66
		GPTQ	1.42×10^5	4.06×10^5	0.10	25.10	0.00	40.40	27.01	28.40	20.00	20.14
		MxMoE	37.76	44.86	0.00	34.15	0.00	60.52	50.31	31.40	21.91	28.33
		CodeQuant	14.34	25.99	13.50	53.35	15.24	75.90	59.48	42.00	29.11	41.23
		MxQBound	**6.52**	**9.83**	**46.63**	**57.57**	**31.10**	**79.14**	**62.16**	**43.40**	**34.54**	**50.65**
	Mixtral	RTN	20.80	26.54	4.32	39.13	3.66	68.32	66.19	36.40	26.60	34.95
		GPTQ	21.69	25.56	3.34	38.03	3.66	66.85	64.93	35.20	26.50	34.07
		MxMoE	17.68	98.31	0.00	29.82	0.61	66.42	55.68	36.40	26.00	30.70
		CodeQuant	5.87	10.04	25.55	54.55	18.29	79.82	76.64	42.80	35.88	47.65
		MxQBound	**5.27**	**6.82**	**56.48**	**67.97**	**33.54**	**84.59**	**82.68**	**43.40**	**40.17**	**58.40**
W4A8	DS-V2	RTN	6.79	10.62	12.28	54.75	23.78	78.62	76.34	38.19	37.22	45.88
		GPTQ	6.86	10.65	34.34	54.83	24.40	76.60	56.65	43.20	37.20	46.75
		MxMoE	6.36	10.31	33.28	54.85	25.00	74.34	76.05	43.40	37.22	49.16
		CodeQuant	6.45	9.53	36.85	**57.74**	25.61	79.02	77.51	**45.00**	38.63	51.48
		MxQBound	**5.97**	**9.11**	**39.12**	57.21	**28.05**	**79.85**	**78.05**	43.00	**39.70**	**52.14**
	Qwen	RTN	8.57	12.17	43.52	55.69	21.95	77.22	74.38	35.71	34.54	49.00
		GPTQ	8.89	12.33	38.40	52.90	11.00	73.00	51.98	40.80	32.20	42.90
		MxMoE	7.39	10.89	35.71	54.39	28.05	77.86	76.55	42.60	34.10	49.89
		CodeQuant	12.43	21.68	26.38	57.03	29.27	77.74	63.98	**43.40**	32.60	47.20
		MxQBound	**6.41**	**9.54**	**62.83**	**58.53**	**33.54**	**79.82**	**76.67**	42.40	**35.04**	**55.54**
	Mixtral	RTN	4.54	8.08	48.52	66.85	**37.80**	81.71	81.94	44.60	39.23	57.24
		GPTQ	4.54	8.06	49.20	67.23	34.18	82.50	81.08	45.40	39.30	56.98
		MxMoE	4.59	23.70	35.78	56.92	33.53	81.01	76.86	42.80	36.42	51.90
		CodeQuant	5.45	9.45	30.71	58.72	36.59	81.77	78.42	43.20	36.95	52.33
		MxQBound	**4.22**	**6.73**	**57.92**	**69.12**	37.20	**84.71**	**83.28**	**48.20**	**41.07**	**60.21**
W5A5	DS-V2	RTN	7.56	11.78	10.22	50.89	22.61	72.21	75.43	44.40	35.50	44.47
		GPTQ	7.60	11.93	30.02	50.24	23.20	69.70	55.96	40.80	36.60	43.79
		MxMoE	**6.50**	10.65	30.10	54.49	23.17	77.61	76.36	45.00	36.88	49.09
		CodeQuant	6.51	9.61	35.25	57.35	24.39	78.59	77.27	45.00	38.39	50.89
		MxQBound	6.58	**8.95**	**38.74**	**57.54**	**27.44**	**79.17**	**77.40**	**45.40**	**40.50**	**52.31**
	Qwen	RTN	27.59	196.38	18.01	23.51	0.00	55.84	47.52	33.00	27.90	29.40
		GPTQ	56.18	2.89×10^3	9.63	23.32	0.00	55.34	0.00	34.40	28.10	21.54
		MxMoE	7.99	12.32	25.17	56.40	21.34	72.69	73.32	37.60	28.81	45.05
		CodeQuant	12.79	22.22	23.65	56.35	20.73	76.09	63.42	42.60	31.83	44.95
		MxQBound	**6.32**	**9.36**	**63.76**	**59.13**	**33.54**	**79.72**	**77.11**	**42.80**	**35.95**	**56.00**
	Mixtral	RTN	5.95	10.18	42.91	61.99	26.22	79.85	80.35	45.80	36.58	53.39
		GPTQ	5.99	10.20	41.93	62.31	26.83	79.69	79.86	43.60	36.42	52.95
		MxMoE	5.12	24.22	42.15	56.90	30.49	81.47	82.16	46.80	36.82	53.83
		CodeQuant	5.55	9.57	48.96	57.81	32.13	81.47	78.27	43.40	36.85	54.13
		MxQBound	**5.07**	**6.67**	**61.33**	**69.10**	**40.24**	**85.20**	**83.90**	**47.20**	**40.64**	**61.09**

importance directly from expert outputs collected in a single forward pass, substantially reducing the computational overhead. The only exception is vanilla RTN. However, given its severe degradation of model performance, as reflected in Table 1, RTN remains inapplicable to modern MoE-LLMs, as concurred in prior studies (Chowdhury et al., 2026; Chen et al., 2025).

Compression effectiveness. As shown in Table 3, MxQBound achieves compression ratios of 2.60×-3.44×, comparable to the competing methods across models and quantization settings. Given its superior model performance, substantially lower computational overhead, and negligible performance degradation even under aggressive quantization regimes, MxQBound prevails in delivering robust and efficient mixed-precision quantization for MoE-LLMs.

4.4 Ablation Studies

We conduct two ablation studies of MxQBound on Qwen1.5-MoE-A2.7B under W4A8 to respectively examine the contributions of the three-stage quantization pipeline and the activation quantization

Table 2: Time consumption of different methods.

Conf	Method	DS-V2	Qwen	Mixtral
W4A4	RTN	1 s	1 s	3 s
	GPTQ	34 min	1.6 h	1.1 h
	MxMoE	16 h	15.5 h	34.8 h
	CodeQuant	1.3 h	3.2 h	35.4 h
	MxQBound	**19 min**	**32 min**	**15 min**
W4A8	RTN	1 s	1 s	2 s
	GPTQ	1.5 h	1.3 h	1.4 h
	MxMoE	16 h	15.5 h	34.8 h
	CodeQuant	1.6 h	8.3 h	36 h
	MxQBound	**44 min**	**37 min**	**50 min**
W5A5	RTN	1 s	1 s	2 s
	GPTQ	1.5 h	2.6 h	1.4 h
	MxMoE	16 h	15.5 h	34.8 h
	CodeQuant	1.8 h	9.4 h	40 h
	MxQBound	**47 min**	**40 min**	**52 min**

Table 3: Compression ratio of different methods.

Conf	Method	DS-V2	Qwen	Mixtral
W4A4	RTN	3.13×	2.83×	3.53×
	GPTQ	3.13×	2.83×	3.53×
	MxMoE	3.43×	3.29×	3.62×
	CodeQuant	3.13×	2.83×	3.53×
	MxQBound	3.06×	2.77×	3.44×
W4A8	RTN	3.13×	2.83×	3.53×
	GPTQ	3.13×	2.83×	3.53×
	MxMoE	3.22×	3.09×	3.38×
	CodeQuant	3.13×	2.83×	3.53×
	MxQBound	3.06×	2.77×	3.44×
W5A5	RTN	2.65×	2.45×	2.91×
	GPTQ	2.65×	2.45×	2.91×
	MxMoE	2.93×	2.83×	3.05×
	CodeQuant	2.65×	2.45×	2.91×
	MxQBound	2.60×	2.41×	2.85×

components in Stage I. Specifically, for the stage ablation, we use the complete MxQBound with *All stages* as the baseline, and individually remove Stage I (offline activation bit allocation), Stage II (offline weight bit allocation) and Stage III (online activation bit adaptation) to construct three pipeline variants: *No Stage I*, *No Stage II* and *No Stage III*. Then, we respectively measure their text generation perplexity and downstream task accuracy, as shown in Table 4. For ease of comparison, we report the performance degradation of each variant relative to the baseline. As weights primarily govern the mapping from inputs to outputs, removing Stage II causes obvious performance degradation, a finding consistent with most prior works. Albeit marginal in terms of benchmark score decrement, Stage III is pivotal to the robustness of MxQBound, especially under aggressive quantization regimes. In contrast, removing Stage I causes the most severe degradation, reaffirming that the quantization of the product activation z is the dominant source of quantization error in MoE-LLMs.

Given the criticality of Stage I, we further examine the contribution of each of its individual components on Qwen1.5-MoE-A2.7B. Specifically, we use the complete MxQBound with *All components* in Stage I as the baseline, and individually remove the scaling matrix S_e, the RHT, and the clipping search to construct three variants: *No scaling*, *No RHT* and *No clipping*. Similar to Table 4, we also respectively measure their performance degradation relative to the baseline, as shown in Table 5. We observe that *No scaling* and *No RHT* cause substantially higher performance degradation than *No clipping*. This is primarily because the scaling matrix and RHT are essential for suppressing activation outliers, and hence pivotal to the robustness of activation quantization against frequent outliers. Clipping search, which adapts the quantization range to reduce activation reconstruction error, is comparatively less important, as range fluctuation occurs far less frequently than outliers.

Table 4: Ablation study of the three-stage quantization pipeline.

Variant	PPL (↓)		Accuracy score (↑)	
	WT2	C4	GSM8K	MMLU
No Stage I	+0.692	+0.845	-9.5	-5.5
No Stage II	+0.471	+0.480	-6.5	-3.8
No Stage III	+0.104	+0.090	-1.0	-1.3
All stages	**6.384**	**9.439**	**61.5**	**57.3**

Table 5: Ablation study of activation quantization components in Stage I.

Variant	PPL (↓)		Accuracy score (↑)	
	WT2	C4	GSM8K	MMLU
No scaling	+0.729	+0.328	-18.0	-6.2
No RHT	+0.556	+0.107	-5.0	-2.1
No clipping	+0.078	+0.045	-6.0	-0.3
All components	**6.578**	**9.443**	**60.0**	**57.6**

5 Conclusion

In this paper, we have systematically analyzed the dominant sources of quantization error in MoE-LLMs and revealed a fundamental misalignment between existing importance proxies and actual expert importance. To this end, we proposed MxQBound, a cost-efficient and theoretically grounded mixed-precision quantization framework for MoE-LLMs. It assesses expert importance directly from expert outputs using a single forward pass, derives a theoretical upper bound on expert-level quantization error, enables importance and error-aware bit allocation for weights and activations, and adaptively adjusts activation bit-widths according to input strength at inference time. Extensive experiments across diverse MoE-LLMs and benchmarks demonstrated the effectiveness of MxQBound, highlighting it as an efficient and robust mixed-precision quantization solution for MoE-LLMs across extremely low to moderate bit budgets.

6 AI USE STATEMENT

In this work, we used generative AI tools for paper writing, including spell checking, grammar checking, and polish writing. We did not use generative AI to conduct research, generate research ideas, or draft any section of the manuscript from scratch; all text originates from the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

7 REPRODUCIBILITY STATEMENT

The paper has fully disclosed all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and conclusions of the paper. We have provided a link (https://anonymous.4open.science/r/MxQBound) to a downloadable source code with detailed explanations and annotations to support the reproducibility of our work. In addition, the necessary experimental and implementation details are described in Section 4.1 and Appendix G.

REFERENCES

Milad Alizadeh, Arash Behboodi, Mart van Baalen, Christos Louizos, Tijmen Blankevoort, and Max Welling. Gradient ℓ_1 regularization for quantization robustness. In *Proc. of ICLR*, 2020.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proc. of NAACL-HLT*, 2019.

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

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

Zhixuan Chen, Xing Hu, Dawei Yang, Zukang Xu, Chen Xu, Zhihang Yuan, Sifan Zhou, and Jiangyong Yu. MoEQuant: Enhancing quantization for mixture-of-experts large language models via expert-balanced sampling and affinity guidance. In *Proc. of ICML*, 2025.

Mohammed Nowaz Rabbani Chowdhury, Kaoutar El Maghraoui, Hsinyu Tsai, Naigang Wang, Geoffrey W. Burr, Liu Liu, and Meng Wang. Efficient quantization of mixture-of-experts with theoretical generalization guarantees. In *Proc. of ICLR*, 2026.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proc. of NAACL-HLT*, 2019.

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. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepSeek-AI. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

Haojie Duanmu, Xiuhong Li, Zhihang Yuan, Size Zheng, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. Mxmoe: Mixed-precision quantization for moe with accuracy and performance co-design. In *Proc. of ICML*, 2025.

Marco Federici, Boris van Breugel, Paul Whatmough, and Markus Nagel. Dissecting quantization error: A concentration-alignment perspective. *arXiv preprint arXiv:2603.04359*, 2026.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *Proc. of ICML*, 2023.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *Proc. of ICLR*, 2023.

Zhongqian Fu, Tianyi Zhao, Ning Ding, Xianzhi Yu, Xiaosong Li, Yehui Tang, and Yunhe Wang. EAQuant: Enhancing post-training quantization for MoE models via expert-aware optimization. *arXiv preprint arXiv:2506.13329*, 2025.

Babak Hassibi and David G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *Proc. of NeurIPS*, 1993.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *Proc. of ICLR*, 2021.

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, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In *Proc. of NeurIPS*, 1990.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *Proc. of ICLR*, 2019.

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

Zichong Li, Chen Liang, Zixuan Zhang, Ilgee Hong, Young Jin Kim, Weizhu Chen, and Tuo Zhao. SlimMoE: Structured compression of large MoE models via expert slimming and distillation. In *Proc. of COLM*, 2025.

Darryl D. Lin, Sachin S. Talathi, and V. Sreekanth Annapureddy. Fixed point quantization of deep convolutional networks. In *Proc. of ICML*, 2016.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proc. of MLSys*, 2024.

Eldad Meller, Alexander Finkelstein, Uri Almog, and Mark Grobman. Same, same but different: Recovering neural network quantization error through weight factorization. In *Proc. of ICML*, 2019.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *Proc. of ICLR*, 2017.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proc. of EMNLP*, 2018.

Qwen Team. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2024.

Qwen Team. Qwen3-Next: Towards ultimate training & inference efficiency. https://qwen.ai/blog?from=research.latest-advancements-listHugging&id=4074cca80393150c248e508aa62983f9cb7d27cd, September 2025. Accessed: 2026-09-07.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 21(140):1–67, 2020.

Chenyang Song, Weilin Zhao, Xu Han, Chaojun Xiao, Yingfa Chen, Yuxuan Li, Zhiyuan Liu, and Maosong Sun. BlockFFN: Towards end-side acceleration-friendly mixture-of-experts with chunk-level activation sparsity. In *Proc. of COLM*, 2025.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models. In *Proc. of ICLR*, 2024.

Hidenori Tanaka, Daniel Kunin, Daniel L. K. Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. In *Proc. of NeurIPS*, 2020.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *Proc. of ICLR*, 2020.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proc. of ICML*, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Xiangyang Yin, Xingyu Liu, Tianhua Xia, Bo Bao, Vithursan Thangarasa, Valavan Manohararajah, Eric Sather, and Sai Qian Zhang. CodeQuant: Unified clustering and quantization for enhanced outlier smoothing in low-precision mixture-of-experts. In *Proc. of ICLR*, 2026.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proc. of ACL*, 2019.

Appendix

List of Contents

A: Related Work
B: Cross-Calibration Stability of Expert-Importance Metrics
C: Pruning-Based Validation of Expert-Importance Metrics
D: Effect of the Second-Order Term on Output Quantization Error
E: Validation of Online Activation Bit-Width Adaptation
F: Algorithm of Three-Stage Quantization Pipeline
G: Experimental and Implementation Details
- G.1 Models and Calibration
- G.2 Quantization Configurations
- G.3 Baselines and Evaluation Protocols
- G.4 Hardware and Software

H: Performance Results on More Settings
- H.1 Impact on Larger MoE-LLM Performance
- H.2 Performance under Weight-Only Quantization Configurations

I: Limitations
J: Broader Impact

A Related Work

Quantization of Dense LLMs. Weight-only methods such as GPTQ (Frantar et al., 2023) and AWQ (Lin et al., 2024) reduced weight quantization error using Hessian information or activation statistics, while joint weight-activation methods such as SmoothQuant (Xiao et al., 2023) and QuaRot (Ashkboos et al., 2024) mitigated activation outliers through channel-wise scaling or orthogonal transformations. Despite their differences, these approaches share a common premise: *intra-layer homogeneity*. That is, all parameters within the same layer are assumed to be quantized with the same strategy. However, in the context of MoE-LLMs, each token activates only a small subset of experts, which exhibit substantially distinct activation statistics and quantization sensitivities (Li et al., 2024). As a result, directly transferring these quantization strategies to MoE layers will inevitably misallocate precision, over-protecting robust experts while under-protecting sensitive ones.

Quantization of MoE-LLMs. Duanmu et al. (2025) introduced a mixed-precision quantization framework that jointly determines sensitivity- and activation-aware bit allocation via integer programming. Chen et al. (2025) proposed a quantization method that mitigates routing-induced calibration imbalance via token-expert affinity. Chowdhury et al. (2026) proposed an expert-wise mixed-precision quantization strategy that allocates bit-width per expert based on routing norm statistics. However, existing guidance metrics for quantization cannot reliably match actual expert importance. Under extreme compression (e.g., W4A4), the misalignment can induce a systematic bias in bit allocation, ultimately leading to catastrophic model collapse. Several recent approaches have been proposed to fix this deficiency. For example, Fu et al. (2025) proposed an expert-aware smoothing method, which rescales each expert's activations to suppress outliers. Yin et al. (2026) proposed CodeQuant, which uses quantization-aware training to learn a rotation that decorrelates activation outliers. However, the computational overhead associated with rescaling or relearning remains a critical bottleneck.

Parameter Importance Assessment for LLMs. Neural network pruning motivated the development of parameter importance assessment. In particular, OBD (LeCun et al., 1990) and OBS (Hassibi & Stork, 1993) are two earliest approaches that utilize Hessian information for parameter importance assessment. As model sizes grew larger, the strictness of this assessment paradigm was gradually relaxed. For instance, SNIP (Lee et al., 2019) and GraSP (Wang et al., 2020) demonstrated that first-order sensitivity alone is sufficient for assessment. SynFlow (Tanaka et al., 2020) further confirmed that parameter importance can be assessed without data. The emergence of LLMs made retraining-based model pruning impractical, thereby shifting the pruning objective from global loss minimization to local output reconstruction. Under this paradigm, SparseGPT (Frantar & Alistarh, 2023) adopted localized second-order reconstruction, while Wanda (Sun et al., 2024) employed the activation-weighted score, to assess column-level importance of parameters. However, in an expert network of MoE-LLMs, the gate, up and down projection matrices are intertwined via SiLU gating, creating multiplicative interactions that cannot be captured by assessment metrics applied per parameter or per column. Consequently, existing methods cannot be directly extended to assess expert-level importance.

B Cross-Calibration Stability of Expert-Importance Metrics

To examine whether the observations in Section 2.2 generalize to other MoE architectures, we extend the cross-calibration-dataset stability analysis to DeepSeek-V2-Lite and Mixtral-8×7B. We compare two expert-importance metrics: routing frequency and the ℓ_2 norm of expert outputs. The experimental procedures are identical to those used in Section 2.2.

Routing Frequency. As shown in Fig. 6 to 8, expert importance measured by routing frequency exhibits limited consistency across calibration sets. The average Spearman correlations across calibration-dataset pairs are 0.32, 0.15 and 0.21 for DeepSeek-V2-Lite, Qwen1.5-MoE-A2.7B and Mixtral-8×7B, respectively. For most calibration-dataset pairs, the correlations are below 0.4. Some are close to zero or negative, such as that between C4 and HumanEval on Qwen1.5-MoE-A2.7B.

These results indicate that routing-frequency-based expert importance is consistently sensitive to the choice of calibration datasets across all three MoE architectures, indicating that the observations in Section 2.2 are not model-specific. Routing frequency alone therefore has limited stability as an expert-importance metric across calibration datasets from different domains.

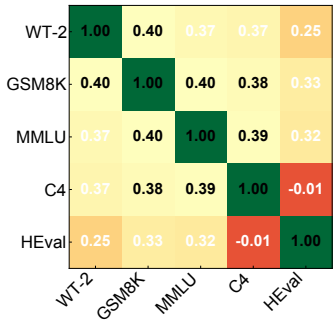


Figure 6: Cross-calibration stability of routing frequency on DeepSeek-V2-Lite.

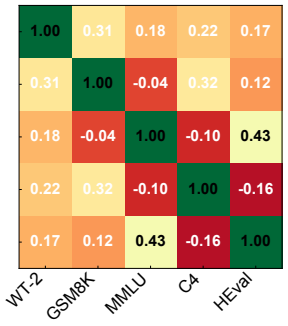


Figure 7: Cross-calibration stability of routing frequency on Qwen1.5-MoE-A2.7B.

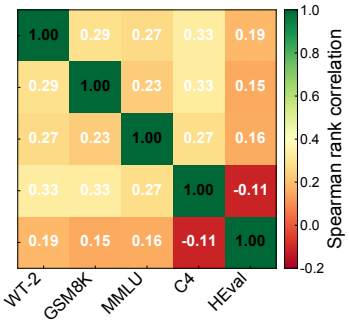


Figure 8: Cross-calibration stability of routing frequency on Mixtral-8×7B.

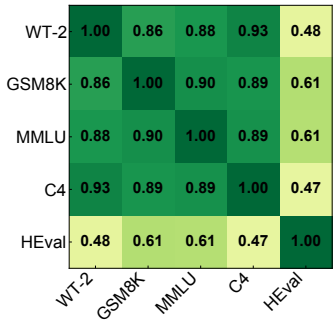


Figure 9: Cross-calibration stability of expert output norm on DeepSeek-V2-Lite.

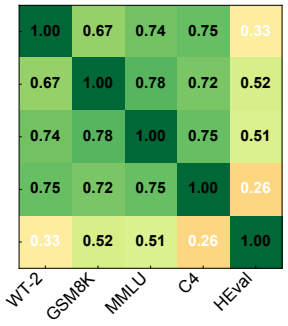


Figure 10: Cross-calibration stability of expert output norm on Qwen1.5-MoE-A2.7B.

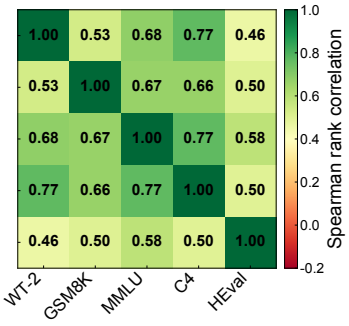


Figure 11: Cross-calibration stability of expert output norm on Mixtral-8×7B.

ℓ_2 Norm of Expert Outputs. In comparison, as shown in Fig. 9 to 11, the ℓ_2 norm of expert outputs exhibits greater consistency across calibration datasets. The average Spearman correlations are 0.75, 0.60 and 0.61 for DeepSeek-V2-Lite, Qwen1.5-MoE-A2.7B, and Mixtral-8×7B, respectively, which are much higher than those measured using routing frequencies. For DeepSeek-V2-Lite, the correlations for most calibration-dataset pairs fall between 0.7 and 0.9. Although compared to DeepSeek-V2-Lite, the overall correlations vary more on Qwen1.5-MoE-A2.7B and Mixtral-8×7B, they remain more consistent than those in Fig. 7 and 8 across calibration datasets. These results indicate that expert output norm is more stable in measuring expert importance.

C Pruning-Based Validation of Expert-Importance Metrics

Following the observations in Section 2.3, we further evaluate whether the mean ℓ_2 norm of expert outputs calculated over the calibration datasets is more reliable in reflecting the importance of the expert to model performance. The experimental procedures are identical to those in Section 2.3.

As shown in Fig. 12, in contrast to Fig. 2, pruning experts based on the mean ℓ_2 norm monotonically degrades the GSM8K accuracy of all the models. Specifically, on DeepSeek-V2-Lite and Mixtral-8×7B, GSM8K accuracy decreases almost linearly as the pruning ratio increases. In comparison, on Qwen1.5-MoE-A2.7B, pruning only 1% of the experts causes a sharp drop in GSM8K accuracy from 61.0% to 2.0%. The contrast indicates that, compared with routing weights, the ℓ_2 norm of expert outputs better aligns with the actual importance of the expert, and can therefore more reliably guide expert-level bit allocation.

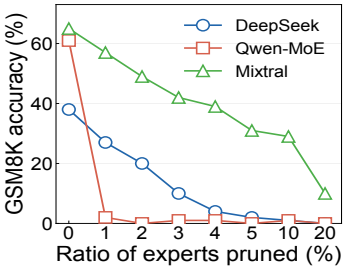


Figure 12: Pruning experts in descending order of the l_2 norm of expert outputs monotonically degrades the GSM8K accuracy of the models.

D Effect of the Second-Order Term on Output Quantization Error

In this section, we theoretically analyze why the contribution of the second-order term $\delta W \delta x$ in Eq. (3) is trivial. First, following Eq. (3), the complete representation of the output error is

$$\delta y := \hat{W}\hat{x} - Wx = \delta W x + W \delta x + \delta W \delta x. \tag{14}$$

The bounds for the first two terms have been deduced in Section 3.2:

$$\|\delta W x\|_2 \leqslant \epsilon_w I_e, \qquad \|W \delta x\|_2 \leqslant \epsilon_a I_e. \tag{15}$$

From the definition of $\epsilon_a := \frac{\|W\|_2 \|x\|_2}{\|Wx\|_2} \cdot \frac{\|\delta x\|_2}{\|x\|_2}$ and $\|Wx\|_2 \leqslant \|W\|_2 \|x\|_2$, we have

$$\frac{\|\delta x\|_2}{\|x\|_2} \leqslant \epsilon_a.$$

Given the definition of $\epsilon_w := \frac{\|\delta W\|_2 \|x\|_2}{\|Wx\|_2}$, the second-order term can therefore be bounded as

$$\|\delta W \delta x\|_2 \leqslant \|\delta W\|_2 \|\delta x\|_2 = \epsilon_w I_e \frac{\|\delta x\|_2}{\|x\|_2} \leqslant \epsilon_w \epsilon_a I_e. \tag{16}$$

Combining Eq. (14), Eq. (15) and Eq. (16), we have

$$\|\delta y\|_2 \leqslant (\epsilon_w + \epsilon_a + \epsilon_w \epsilon_a) I_e. \tag{17}$$

Let $\eta := \max(\epsilon_w, \epsilon_a)$. Since $\epsilon_w \epsilon_a \leqslant \frac{(\epsilon_w + \epsilon_a)^2}{4} \leqslant \frac{\eta}{2}(\epsilon_w + \epsilon_a)$, Eq. (17) becomes

$$\|\delta y\|_2 \leqslant \left(1 + \frac{\eta}{2}\right)(\epsilon_w + \epsilon_a) I_e. \tag{18}$$

Accordingly,

$$\Delta_e := \|\delta y\|_2^2 \leqslant \left(1 + \frac{\eta}{2}\right)^2 (\epsilon_w + \epsilon_a)^2 I_e^2. \tag{19}$$

Thus, the upper bound including the second-order term is at most $(1 + \eta/2)^2$ times the first-order bound in Eq. (4), providing a worst-case characterization of the effect of the second-order term on quantization error. Since the quantization errors (δW and δx) are typically small perturbations relative to W and x, respectively (Lin et al., 2016; Meller et al., 2019; Federici et al., 2026), we have $\eta \to 0$ in most cases. Therefore, the second-order term $\delta W \delta x$ contributes negligibly to the output quantization error δy, and tends to quadratically decrease along with the improvement of quantization strategies.

E Validation of Online Activation Bit-Width Adaptation

To validate that the input activation norm serves as a reliable proxy for the corresponding output activation, and can thus be used to adjust the default activation bit-width $b^a_{e,\text{off}}$ at inference time as described in Section 3.3.3, we conduct an experimental analysis on the consistency between experts' input and output activations.

Specifically, on the one hand, as in prior mixed-precision quantization methods for MoE-LLMs (Duanmu et al., 2025; Yin et al., 2026), each expert e in MxQBound receives multiple input hidden states and their corresponding outputs during calibration. Thus, the average baseline of the input in Eq. (13) (i.e., $\overline{\|x_e\|}_2$) is calculated as the mean over all the received inputs. Likewise, we analogously calculate the corresponding average baseline of the output (denoted as $\overline{\|Y_e(x_e)\|}_2$). On the other hand, following the general MoE-LLM inference paradigm, the inputs routed to expert e are grouped into windows of N inputs each (Song et al., 2025). Accordingly, the input norm at the current (i.e., t-th) window, $\|x_e^{(t)}\|_2$, can be calculated as the mean of the inputs within the window, and the

corresponding output norm at current window (denoted as $\|Y_e(x_e)^{(t)}\|_2$) is calculated in the same manner. Then, we respectively calculate the current-to-baseline ratios of the inputs and outputs as:

$$r^x_{e,t} = \frac{\|x_e^{(t)}\|_2}{\overline{\|x_e\|_2}}, \qquad r^y_{e,t} = \frac{\|Y_e(x_e)^{(t)}\|_2}{\overline{\|Y_e(x_e)\|_2}}.$$

Finally, for the inference of DeepSeek-V2-Lite, Qwen1.5-MoE-A2.7B and Mixtral-8×7B on GSM8K, we compute $r^x_{e,t}/r^y_{e,t}$ of each expert throughout the inference. Fig. 13 shows the distributions of these ratios across experts for each model. Each box indicates the median, the 25th and the 75th percentiles of the results. The line extending upward represents the 95th percentile value, and the line extending downward represents the 5th percentile value of the measured results. Across all three models, most ratios fall within [0.97, 1.04], and the standard deviation is below 0.083. This means that $r^x_{e,t}/r^y_{e,t}$ remains close to 1 throughout inference with trivial fluctuation. We further extend the computation of $r^x_{e,t}/r^y_{e,t}$ to other datasets, of which results are shown in Table 6. Across the nine datasets and three models, the median of $r^x_{e,t}/r^y_{e,t}$ concentrates within [0.9748, 1.0232], and the standard deviation concentrates within [0.0679, 0.1719]. These results indicate that $r^x_{e,t}$ and $r^y_{e,t}$ are generally consistent across tasks and models. Since $r^x_{e,t}$ is available before the expert output is computed, this consistency supports its use as the online activation bit-width adjustment signal in Stage III.

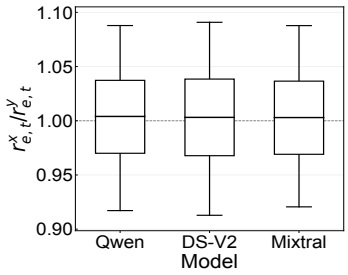


Figure 13: Distribution of $r^x_{e,t}/r^y_{e,t}$ across Qwen, DeepSeek, and Mixtral on GSM8K.

Table 6: Medians and standard deviations of $r^x_{e,t}/r^y_{e,t}$ across models and datasets.

Dataset	Qwen1.5-MoE-A2.7B	DeepSeek-V2-Lite	Mixtral-8×7B
WikiText-2	1.0085±0.1450	1.0124±0.1352	1.0065±0.1039
C4	1.0117±0.1719	1.0148±0.1607	1.0066±0.0931
GSM8K	1.0088±0.0938	1.0091±0.0943	1.0060±0.0679
MMLU	1.0112±0.1528	1.0027±0.1182	0.9827±0.0784
HumanEval	1.0117±0.1030	1.0125±0.1160	0.9748±0.0963
BoolQ	1.0089±0.1036	1.0133±0.1257	1.0232±0.0952
HellaSwag	1.0080±0.1130	1.0134±0.1168	1.0059±0.0857
OpenBookQA	1.0043±0.0744	1.0045±0.0822	1.0021±0.0820
MathQA	1.0046±0.0761	1.0080±0.0910	0.9998±0.0906

F Algorithm of Three-Stage Quantization Pipeline

Algorithm 1 summarizes the three-stage pipeline proposed in Sections 3.3.1, 3.3.2 and 3.3.3. Given the target MoE-LLM for quantization M, a calibration set $\mathcal{D}_{\text{calib}}$, candidate activation and weight bit-width sets $\mathcal{B}_a$ and $\mathcal{B}_w$, activation and weight budgets B_a and B_w, and candidate clipping-threshold sets $\{\mathcal{T}_e \mid e \in \mathcal{E}\}$, the algorithm outputs the quantization strategy for M. For each expert, the algorithm specifies the offline activation configuration $(T^\star_e, b^a_{e,\text{off}})$, the offline weight bit-width $b^w_{e,\text{off}}$, and determines the input-dependent activation bit-width $b^a_e(t)$ during inference.

Specifically, at Lines 1-3, the algorithm performs a single forward pass to collect each expert's input x_e, product activation z, and output $Y_e(x_e)$. The resulting statistics are used to compute the expert importance I_e and the mean input norm $\overline{\|x_e\|_2}$. At Lines 4-8, the algorithm determines the offline activation configuration in stage I. For each expert, the algorithm computes the channel-wise scaling matrix S_e and rewrites its output as Eq. (5). For each expert e, the algorithm evaluates every clipping threshold $T_{e,n} \in \mathcal{T}_e$ and activation bit-width $b^a_{e,k} \in \mathcal{B}_a$ using activation quantization error $\epsilon^2_a(T_{e,n}, b^a_{e,k}; S_e)I^2_e$. After evaluating all candidates, the algorithm solves the activation bit allocation ILP in Eq. (7) to (9) under the activation budget B_a to jointly determine $\{(T^\star_e, b^a_{e,\text{off}}) \mid e \in \mathcal{E}\}$. At Lines 9-14, the algorithm determines the offline weight bit-widths in stage II. For each expert and each candidate $b^w_{e,k} \in \mathcal{B}_w$, the algorithm evaluates the weight quantization error with expert importance $\epsilon^2_w(b^w_{e,k}; S_e)I^2_e$. The weight bit-allocation ILP is then solved jointly under the weight

Algorithm 1: Three-stage quantization pipeline in MxQBound.

Input: Target model M, calibration set $\mathcal{D}_{\text{calib}}$, candidate activation and weight bit-width sets $\mathcal{B}_a$ and $\mathcal{B}_w$, activation and weight budgets B_a and B_w , candidate clipping-thresholds $\{\mathcal{T}_e \mid e \in \mathcal{E}\}$.

Output: Quantized model $\widehat{M}$.

`// Calibration and expert-level importance assessment (Sections 3.1 and 3.2)`

1 **for** $x \in \mathcal{D}_{\text{calib}}$ **do**

2 Run one forward pass and collect x_e, z, and $Y_e(x_e)$;

3 Accumulate $I_e = \|Y_e(x_e)\|_2$ and $\overline{\|x_e\|}_2$ for expert e;

`// Stage I: offline activation bit allocation (Section 3.3.1)`

4 **for** *each expert* $e \in \mathcal{E}$ **do**

5 Compute the scaling matrix S_e and rewrite $Y_e(x_e) = (W_{\text{down},e}S_e)(S_e^{-1}z)$;

6 **for** $T_{e,n} \in \mathcal{T}_e$ *and* $b^a_{e,k} \in \mathcal{B}_a$ **do**

7 Evaluate $\epsilon_a^2(T_{e,n}, b^a_{e,k}; S_e)I_e^2$;

8 Solve the activation bit allocation ILP in Eq. (7) to (9) to obtain $\{(T_e^\star, b^a_{e,\text{off}}) \mid e \in \mathcal{E}\}$;

`// Stage II: offline weight bit allocation (Section 3.3.2)`

9 **for** *each expert* $e \in \mathcal{E}$ **do**

10 **for** $b^w_{e,k} \in \mathcal{B}_w$ **do**

11 Evaluate $\epsilon_w^2(b^w_{e,k}; S_e)I_e^2$;

12 Solve the weight bit allocation ILP in Eq. (10) to (12) to obtain $b^w_{e,\text{off}}$;

13 **for** *each expert* $e \in \mathcal{E}$ **do**

14 Quantize its gate, up and down projection matrices using $b^w_{e,\text{off}}$;

`// Stage III: online activation bit adaptation (Section 3.3.3)`

15 **for** *each inference step* t **do**

16 **for** *each active expert* e **do**

17 Compute the current input norm $\|x_e^{(t)}\|_2$;

18 Set $b_e^a(t) \leftarrow \dfrac{\|x_e^{(t)}\|_2}{\overline{\|x_e\|}_2} b^a_{e,\text{off}}$;

19 Map $b_e^a(t)$ to a feasible bit-width in $\mathcal{B}_a$;

20 **return** $\widehat{M}$;

budget B_w in Eq. (10) to (12). This produces the offline weight bit-widths $\{b^w_{e,\text{off}} \mid e \in \mathcal{E}\}$. The gate, up-projection, and down-projection weight matrices of each expert are then quantized using the selected bit-width. At Lines 15-19, the algorithm adapts the activation bit-width at inference time as in stage III. At each inference window t, the algorithm computes the current input norm of every active expert. It adjusts the offline activation bit-width according to the ratio between the current input norm and its calibration mean as in Eq. (13). The resulting continuous target is mapped to a supported bit-width in $\mathcal{B}_a$, yielding the input-dependent bit-width $b_e^a(t)$. Finally, Line 20 returns the quantized model $\widehat{M}$ with the selected offline configurations and the online adaptation strategy.

G Experimental and Implementation Details

G.1 Models and Calibration

In addition to the MoE-LLMs mentioned in Section 4.1, Phi-mini-MoE (Li et al., 2025), Qwen3-30B-A3B (Qwen3 in short) (Yang et al., 2025) and Qwen3-Next-80B-A3B (Qwen3-Next in short) (Qwen Team, 2025) are also used for evaluation, of which architectural details are summarized in Table 7. All models are loaded in FP16, except Qwen3, which is loaded in BF16. In our experiments, MxQBound, MxMoE, and CodeQuant are evaluated under the same quantization scope, covering all expert FFNs, including shared experts. The remaining model components retain their original precision. For MxQBound, the gate, up, and down projections within each expert share the same weight bit-width. Following (Frantar et al., 2023; Frantar & Alistarh, 2023; Sun et al., 2024), for each

Table 7: Architectural configurations and loading precision of the evaluated MoE-LLMs.

Model	Layers	Experts/ layer	Top-k	Hidden dim	Expert dim	Shared expert	Precision
DeepSeek-V2-Lite	27	64	6	2048	1408	Yes	FP16
Qwen1.5-MoE-A2.7B	24	60	4	2048	1408	Yes	FP16
Mixtral-8×7B	32	8	2	4096	14336	No	FP16
Phi-mini-MoE	32	16	2	4096	960	No	FP16
Qwen3-30B-A3B	48	128	8	2048	768	No	FP16
Qwen3-Next-80B-A3B	48	512	10	2048	512	Yes	BF16

Table 8: Evaluation protocols used in the experiments.

Dataset	Evaluation setting	Metric
WikiText-2	Sliding-window evaluation	Perplexity
C4	Sliding-window evaluation	Perplexity
GSM8K	8-shot chain-of-thought, greedy decoding	Exact match
MMLU	5-shot multiple-choice evaluation	Accuracy
HumanEval	Greedy decoding	Pass@1
BoolQ	0-shot evaluation	Accuracy
HellaSwag	0-shot evaluation	Normalized accuracy
OpenBookQA	0-shot evaluation	Normalized accuracy
MathQA	0-shot evaluation	Accuracy

model, we sample 128 sequences of 2,048 tokens from the C4 training set (Raffel et al., 2020). The same calibration set is reused across all quantization budgets and all baselines that require calibration.

G.2 Quantization Configurations

Scaling Matrix. Following SmoothQuant (Xiao et al., 2023), we compute an expert-specific channel-wise scaling matrix S_e from the calibration statistics of the product activation z and the corresponding down projection matrix (i.e., $W_{\text{down},e}$), which mitigates activation outliers by shifting quantization difficulty from activations to weights. The hyperparameter α controls the extent of this shift, and we set $\alpha = 0.5$ in all experiments, consistent with SmoothQuant.

Activation Quantization. Product activations z and gate/up inputs x are quantized in groups of 64 after a randomized Hadamard transform with a fixed seed of 42. For each expert, the clipping configuration T_e for z is selected from candidate percentiles of {90,93,95,97,99,99.5,99.9}. Each candidate is converted into a clipping ratio by normalizing the corresponding percentile by the maximum magnitude of each calibration vector and taking the median across the vectors assigned to that expert. The ratio is bounded to [0.01,1] and applied to the maximum magnitude of each activation group. The gate/up inputs x use the nominal activation precision with a fixed 99.5th-percentile clipping threshold. We set the candidate set for activation precision as $\mathcal{B}_a = \{2, 3, 4, 5, 6, 8, 12, 16\}$.

Weight Quantization. All expert weights are quantized symmetrically in groups of 64 values. Before quantization, the smoothing matrix S_e is applied to the down projection matrices $W_{\text{down},e}$ as in Eq. (5). The gate, up and down projection matrices within each expert share the same bit-width. For each projection, group and candidate bit-width, the quantization scale is selected from a 20-point shrinkage grid by minimizing the weight reconstruction error. The candidate set of weight precision $\mathcal{B}_w$ is the same as $\mathcal{B}_a$.

G.3 Evaluation Protocols

Table 8 summarizes the evaluation protocols. Greedy decoding is used for all generation tasks. Perplexity is computed with a stride of 512 on WikiText-2 and C4, and a stride of 1,024 for Qwen3-Next-80B-A3B. GSM8K and HumanEval use a maximum generation length of 512 tokens. For MMLU, predictions are obtained by comparing the model scores assigned to the four answer choices at the final position.

Table 9: Comparison of impact on larger MoE-LLM performance. Results under Full Precision (FP) are provided for reference. ↓ means lower perplexity is better, and ↑ means higher accuracy is better.

Conf	Model	Method	PPL (↓)		Accuracy score (↑)							
			WT2	C4	GSM8K	MMLU	HEval	BoolQ	HSwag	OBQA	MathQA	Avg
FP16	Phi-mini-MoE	FP	6.21	11.38	84.89	70.68	55.49	83.49	76.08	46.20	44.72	65.93
	Qwen3	FP	6.45	11.28	91.58	80.29	69.51	89.40	79.73	45.00	63.15	74.09
	Qwen3-Next	FP	5.34	9.96	96.13	83.96	95.12	90.76	82.69	46.80	63.79	79.89
W4A4	Phi-mini-MoE	RTN	9.16	17.22	22.21	57.28	25.00	72.84	67.43	39.20	33.60	45.37
		GPTQ	9.09	17.25	30.25	61.52	26.22	70.15	67.34	41.40	33.63	47.21
		MxMoE	**7.47**	14.81	51.78	**62.93**	33.54	78.81	70.59	41.60	35.81	53.58
		MxQBound	7.56	**12.16**	**76.65**	57.86	**34.76**	**80.95**	**71.40**	**42.40**	**39.20**	**57.60**
	Qwen3	RTN	NaN	NaN	0.00	22.95	0.00	37.83	25.04	27.80	20.57	19.17
		GPTQ	NaN	NaN	0.00	22.95	0.00	37.86	25.04	27.60	20.57	19.15
		MxMoE	8.66	16.25	76.72	76.27	54.88	86.24	73.55	41.80	47.91	65.34
		CodeQuant	7.74	13.32	19.03	76.52	59.15	87.40	77.16	44.40	56.55	60.03
		MxQBound	**6.81**	**12.59**	**90.30**	**77.73**	**66.46**	**89.08**	**78.39**	**44.60**	**61.04**	**72.51**
	Qwen3-Next	RTN	8.30	19.33	0.00	22.95	0.00	37.83	25.04	27.60	20.57	19.14
		GPTQ	8.31	19.41	57.32	80.35	42.68	90.15	76.80	43.20	52.86	63.34
		MxMoE	6.17	14.08	76.95	81.42	30.49	90.52	80.89	43.60	58.66	66.08
		CodeQuant	5.97	11.42	58.00	81.57	53.04	**90.64**	81.85	44.00	62.61	67.39
		MxQBound	**5.63**	**10.26**	**95.15**	**82.27**	**61.58**	90.58	**81.93**	**45.40**	**62.78**	**74.24**
W4A8	Phi-mini-MoE	RTN	7.04	13.84	51.40	64.58	39.63	80.18	73.75	44.00	40.17	56.24
		GPTQ	**6.33**	13.07	77.33	66.54	49.39	81.83	75.27	45.20	40.57	62.30
		MxMoE	6.74	13.65	74.53	67.26	50.61	81.47	74.18	44.00	41.78	61.98
		MxQBound	6.85	**11.44**	**81.88**	**68.03**	**52.44**	**83.58**	**75.34**	**45.80**	**43.32**	**64.34**
	Qwen3	RTN	NaN	NaN	0.00	72.95	52.43	37.83	25.04	27.60	20.57	33.77
		GPTQ	6.88	12.99	82.49	79.36	54.26	88.44	78.25	44.40	56.58	69.11
		MxMoE	7.35	13.69	82.64	79.95	57.32	88.32	78.05	44.40	58.16	69.83
		CodeQuant	7.48	12.87	19.18	79.84	67.07	87.68	78.77	**45.00**	59.93	62.50
		MxQBound	**6.58**	**11.57**	**91.36**	**80.11**	**68.90**	**89.17**	**78.99**	44.80	**61.68**	**73.57**
	Qwen3-Next	RTN	5.85	13.40	80.29	83.44	32.93	90.73	81.99	45.00	62.04	68.06
		GPTQ	5.83	13.40	75.59	83.63	40.85	90.83	82.11	43.80	62.48	68.47
		MxMoE	5.80	13.24	82.56	83.52	82.32	90.70	81.78	44.00	61.98	75.27
		CodeQuant	5.79	11.26	83.55	83.31	89.63	90.55	82.15	45.60	62.81	76.80
		MxQBound	**5.43**	**11.22**	**95.53**	**83.68**	**93.29**	**90.89**	**82.47**	**46.40**	**63.08**	**79.33**
W5A5	Phi-mini-MoE	RTN	7.17	14.19	42.99	60.68	28.05	80.43	72.45	44.60	40.20	52.77
		GPTQ	7.10	14.19	43.67	62.87	33.54	80.89	72.06	44.40	39.80	53.89
		MxMoE	7.47	14.81	49.05	63.22	34.15	78.96	70.59	41.60	35.81	53.34
		MxQBound	**7.09**	**11.78**	**82.49**	**65.18**	**45.73**	**83.03**	**73.88**	**45.60**	**41.68**	**62.51**
	Qwen3	RTN	7.56	14.84	84.31	70.75	0.00	88.41	78.10	36.80	59.43	59.69
		GPTQ	7.52	NaN	84.00	73.64	0.00	88.35	77.89	38.00	59.60	60.21
		MxMoE	7.23	13.87	83.70	75.58	65.85	88.96	78.38	44.00	60.03	70.93
		CodeQuant	7.68	13.12	18.04	79.09	66.46	87.95	77.34	43.40	58.39	61.52
		MxQBound	**6.59**	**11.80**	**90.37**	**79.75**	**68.29**	**89.30**	**79.26**	**44.60**	**62.41**	**73.43**
	Qwen3-Next	RTN	6.09	13.93	79.98	83.34	56.71	90.70	81.73	45.60	60.10	71.17
		GPTQ	6.08	13.92	76.04	**83.55**	59.15	90.34	81.53	44.60	61.81	71.00
		MxMoE	5.79	13.31	70.28	82.60	73.17	90.52	82.08	45.00	62.51	72.31
		CodeQuant	5.80	11.23	67.70	83.18	76.21	90.55	**82.45**	45.00	63.18	72.61
		MxQBound	**5.50**	**10.01**	**95.30**	83.36	**83.54**	**90.83**	82.33	**46.00**	**63.69**	**77.86**

G.4 Hardware and Software

Experiments are conducted on a single node equipped with eight NVIDIA RTX A6000 GPUs, each with 48 GB of memory, using NVIDIA driver 535.230.02. DeepSeek-V2-Lite and Qwen1.5-MoE-A2.7B use one GPU, Mixtral-8×7B and Qwen3-30B-A3B use two GPUs, and Qwen3-Next-80B-A3B uses all eight GPUs. Our software environment consists of Python 3.10.20, PyTorch 2.10.0+cu128, Transformers 5.14.1, SciPy 1.15.3, `lm-evaluation-harness` 0.4.10, `fast-hadamard-transform` 1.0.4.post1, and Datasets 4.5.0.

H Performance Results on More Settings

H.1 Impact on Model Performance across More MoE-LLMs

To assess the scalability of MxQBound across a broader range of MoE-LLM architectures, we further extend our evaluation of quantization methods' impact on model performance to Phi-mini-

Table 10: Performance under weight-only quantization configurations. Results under Full Precision (FP) are provided for reference. ↓ means lower perplexity is better, and ↑ means higher accuracy is better.

Conf	Model	Method	PPL (↓)		Accuracy score (↑)							
			WT2	C4	GSM8K	MMLU	HEval	BoolQ	HSwag	OBQA	MathQA	Avg
FP16	DS-V2	FP	5.72	8.74	41.12	58.09	29.90	79.88	77.99	46.00	41.21	53.46
	Qwen	FP	6.22	9.30	63.91	59.81	34.15	79.82	77.24	44.00	37.45	56.63
	Mixtral	FP	3.45	6.63	61.26	70.42	40.85	86.20	83.89	48.40	44.00	62.15
W4A16	DS-V2	RTN	6.44	9.97	**40.41**	56.15	26.22	79.08	57.30	44.00	39.20	48.91
		GPTQ	6.84	10.57	35.33	54.86	25.00	77.34	56.42	**44.20**	38.09	47.32
		MoEQuant	6.42	**9.10**	35.50	57.20	25.60	78.80	77.10	**44.20**	38.70	51.01
		MxQBound	**5.93**	9.14	40.18	**57.51**	**28.05**	**80.37**	**77.67**	43.00	**41.21**	**52.57**
	Qwen	RTN	7.42	10.54	59.89	56.15	26.83	78.65	55.84	42.20	35.78	50.76
		GPTQ	8.35	12.05	47.31	54.29	17.07	77.31	53.53	41.20	32.96	46.24
		MoEQuant	7.55	**9.62**	58.38	58.38	29.87	78.04	76.73	30.20	35.50	52.44
		MxQBound	**6.38**	9.65	**61.94**	**60.27**	**33.54**	**78.78**	**76.34**	**42.40**	**36.08**	**55.62**
	Mixtral	RTN	4.52	8.08	50.72	67.34	34.15	81.90	81.03	**46.20**	39.73	57.30
		GPTQ	4.51	8.00	50.19	67.40	34.27	80.85	81.26	43.60	39.43	56.71
		MoEQuant	4.12	7.74	60.80	69.60	32.20	85.00	80.26	44.60	43.00	59.35
		MxQBound	**3.69**	**6.67**	**61.50**	**70.72**	**40.24**	**85.54**	**82.07**	45.00	**44.50**	**61.37**

MoE, Qwen3 and Qwen3-Next. CodeQuant is not reported for Phi-mini-MoE because its released implementation does not support this architecture.

Table 9 summarizes the quantization performance of different methods on the three MoE-LLMs under varying W+A quantization configurations. The results are generally consistent with those in Table 1. MxQBound achieves the highest average accuracy in all nine model-configuration combinations. Under the more challenging W4A4 regime, it achieves average accuracy scores of 57.60, 72.51 and 74.24 on Phi-mini-MoE, Qwen3, and Qwen3-Next, respectively. Compared with the best competing method on each model (CodeQuant), these results correspond to relative improvements of 7.50%, 20.79% and 10.16%. Similar to Table 1, the advantage of MxQBound is conspicuous on both Qwen3 and Qwen3-Next, where most competing methods collapse to near-zero accuracy or lose roughly half their performance on GSM8K and HEval, whereas MxQBound exhibits little performance degradation (an average accuracy drop of 14.85% relative to the FP16 results across the three models). MxQBound also achieves the highest average accuracy scores under W4A8 and W5A5, with a relative accuracy drop of only 0.70%-5.19% from the corresponding FP16 results.

For text generation quality, MxQBound achieves the lowest C4 perplexity across all nine model-configuration combinations and the lowest WT-2 perplexity in seven of them. The only exceptions occur on Phi-mini-MoE under W4A4 and W4A8, where its WT-2 perplexity exceeds the best results by 0.09 and 0.52, respectively.

Overall, these results validate the scalability of MxQBound across diverse MoE-LLMs, with parameter scales spanning 7.6B (Phi-mini-MoE), 14.3B (Qwen1.5-MoE-A2.7B), 15.7B (DeepSeek-V2-Lite), 30.5B (Qwen3-30B-A3B), 46.7B (Mixtral-8×7B) and 80B (Qwen3-Next-80B-A3B). Despite differences in architecture and expert configuration, MxQBound consistently achieves the best model performance across most benchmarks under varying quantization settings, further confirming its generalizability and robustness irrespective of model architecture and quantization configuration.

H.2 Performance under Weight-Only Quantization Configurations

To isolate the effect of weight-bit allocation from activation-induced coupling, we evaluate all methods under W4A16, where the MoE weights are quantized under an average 4-bit budget while activations remain at 16 bits. We compare MxQBound with uniform RTN, GPTQ (Frantar et al., 2023) and MoEQuant (Chen et al., 2025). The evaluated models, calibration data, downstream tasks, and quantized modules follow the setup described in Section 4.1.

As shown in Table 10, the results are generally consistent with those in Table 1. MxQBound achieves the highest average accuracy scores among all the methods, incurring only a 1.25-1.85% accuracy drop relative to the FP16 baseline. As for text generation quality, MxQBound achieves the lowest PPL in most cases on both WT2 and C4, incurring only a 0.6-6.96% PPL increase over the FP16 baseline. These results confirm that the proposed expert-level bit allocation remains effective even without activation quantization, primarily because the proposed assessment method of expert-level importance reliably identifies the weights that warrant higher-precision quantization.

I LIMITATIONS

As described in Sections 3.1 and 3.3, MxQBound allocates bit-widths according to expert-level importance and quantization error. Although this expert-level design enables efficient bit allocation, it does not explicitly account for dependencies among experts and model components. A promising direction for future work is to extend the current bit allocation strategy to account for interactions among experts and across model components.

As described in Section 3.3.3, MxQBound adapts activation bit-widths using the current input strength of each expert. Although this signal enables lightweight online adaptation, it does not reflect the expert importance of individual inputs to the overall reasoning process. Future work could explore adapting activation bit-widths based on expert importance at different steps of reasoning strategies such as Chain-of-Thought and Tree of Thoughts.

J BROADER IMPACT

MxQBound demonstrates that combining a theoretically grounded expert-importance metric, error-aware bit allocation, and input-strength-adaptive quantization enables robust mixed-precision bit allocation for MoE-LLMs across extremely low to moderate bit budgets. Beyond quantization, the proposed output-norm-based importance metric and quantization error bound suggest broader applicability. Since the ℓ_2 norm of expert outputs more directly reflects each expert's contribution to the MoE output and remains more consistent across calibration datasets than routing frequency or weight, it could facilitate training routers to estimate expert importance prior to expert selection, thereby reducing sensitivity to calibration data and enabling more reliable and efficient expert selection. Moreover, because our error bound relates expert importance and relative quantization error to each expert's output error, it could be extended to other compression techniques, such as expert pruning or dropping, to estimate their impact on model outputs and performance. With dedicated evaluation, these extensions could further support systematic studies of how MoE-LLM compression affects safety, fairness and privacy in deployment.