Rethinking Post-Training Quantization: Introducing a Statistical Pre-Calibration Approach

Alireza Ghaffari[1], Sharareh Younesian[2], Boxing Chen[2], Vahid Partovi Nia[2] and Masoud Asgharian[1]
[1]*Department of Mathematics and Statistics, McGill University, Montreal, Canada*
[2]*Huawei Noah's Ark Lab, Montreal, Canada*

Keywords: Post-Training Quantization (PTQ), Model Compression, Adaptive LASSO.

Abstract: As Large Language Models (LLMs) become increasingly computationally complex, developing efficient deployment strategies, such as quantization, becomes crucial. State-of-the-art Post-training Quantization (PTQ) techniques often rely on calibration processes to maintain the accuracy of these models. However, while these calibration techniques can enhance performance in certain domains, they may not be as effective in others. This paper aims to draw attention to robust statistical approaches that can mitigate such issues. We propose a *weight-adaptive* PTQ method that can be considered a precursor to calibration-based PTQ methods, guiding the quantization process to preserve the distribution of weights by minimizing the Kullback-Leibler divergence between the quantized weights and the originally trained weights. This minimization ensures that the quantized model retains the Shannon information content of the original model to a great extent, guaranteeing robust and efficient deployment across many tasks. As such, our proposed approach can perform on par with most common calibration-based PTQ methods, establishing a new pre-calibration step for further adjusting the quantized weights with calibration. We show that our pre-calibration results achieve the same accuracy as some existing calibration-based PTQ methods on various LLMs.

1 INTRODUCTION

Large Language Models (LLMs) have rapidly evolved, demonstrating unprecedented capabilities in natural language processing tasks. However, the immense computational resources required for their deployment pose significant challenges, particularly in resource-constrained environments. As these models become more complex, the need for efficient deployment strategies becomes increasingly critical. Quantization, a technique that reduces the precision of the model, has emerged as a promising solution to this problem by significantly reducing the computational and memory demands of LLMs while striving to maintain their performance.

Post-training quantization (PTQ) is a widely adopted approach for implementing quantization after a model has been fully trained. Traditionally, PTQ methods rely heavily on calibration processes to fine-tune the quantized model, ensuring that it retains a high degree of accuracy. These calibration techniques have proven effective in various domains, particularly when the target deployment environment closely resembles the conditions under which the model was calibrated. However, their efficacy may diminish in scenarios where the deployment environment diverges from the calibration conditions, leading to suboptimal performance.

For instance, Table 1 shows this limitation when quantizing a Code-Llama model using mainstream PTQ methods such as SpQR (Dettmers et al., 2024b). To showcase the robustness issue of calibration-based PTQ method, we evaluated the coding performance of quantized Code-Llama model (Roziere et al., 2023) on HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) datasets. HumanEval includes 164 human handwritten programming problems with a function signature, docstring, body, and several unit tests, and MBPP consists of around 1,000 crowd-sourced Python programming problems. Table 1 shows that a robust pre-calibration method outperforms SpQR(Dettmers et al., 2024b), demonstrating that if calibration data does not have the same nature as the task, using calibration data decreases the performance.

Given these limitations, there is a growing interest in exploring mathematical approaches to enhance the robustness of PTQ methods. In particular, sta-

Ghaffari, A., Younesian, S., Chen, B., Nia, V. P. and Asgharian, M.
Rethinking Post-Training Quantization: Introducing a Statistical Pre-Calibration Approach.
DOI: 10.5220/0013348800003905

In *Proceedings of the 14th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2025)*, pages 159-169
ISBN: 978-989-758-730-6; ISSN: 2184-4313

Table 1: Comparison of *weight-adaptive* pre-calibration results for Code-Llama models on HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021).

Model	Method	Avg Bits	Human Eval		MBPP	
			pass@1	pass@10	pass@1	pass@10
Code-Llama-7B	FP16	16.00	29.63	59.84	25.87	63.52
	RTN (g128)	4.25	30.13	57.97	28.26	62.42
	SpQR*	4.63	29.94	57.40	27.59	61.78
	Pre-calibration (g128, α=5%)	4.60	**30.34**	**58.60**	28.03	**62.55**
Code-Llama-13B	FP16	16.00	34.79	66.50	30.17	67.51
	RTN (g128)	4.25	33.70	65.88	29.63	66.00
	SpQR*	4.63	34.19	65.69	29.74	66.20
	Pre-calibration (g128, α=6%)	4.67	**34.79**	**66.02**	**31.36**	**66.82**

tistical methods that guide the quantization process itself —prior to any calibration— offer a promising avenue for improvement. By focusing on preserving the underlying distribution of model weights, these approaches can potentially ensure a more consistent performance across diverse deployment scenarios.

In this paper, we introduce a novel *weight-adaptive pre-calibration* quantization method that functions as a precursor to traditional calibration-based techniques. Our method is grounded in a statistical framework that minimizes the Kullback-Leibler divergence between the original weights and quantized weights, thereby preserving the Shannon information content of the model. This pre-calibration step ensures that the quantized model remains robust across various tasks, even before any further calibration is applied.

Note that our approach not only preserves the accuracy of the quantized model but also sets a new initial point for subsequent calibration processes. Through extensive experiments on various LLMs, we show that our pre-calibration method achieves performance on par with existing calibration-based PTQ techniques, offering a more reliable and efficient deployment strategy for LLMs in diverse environments.

To summarize, we make the following contributions

- We introduce a weight-adaptive pre-calibration method that as a precursor to traditional calibration-based methods guides the quantization process to better preserve model information. To the best of our knowledge, this is the first time a statistical pre-calibration method has been proposed to improve the quantization process.
- Our proposed pre-calibration method *classifies* weights and does not adjust them as opposed to traditional PTQ methods. The proposed method then uses pseudo activations (i.e. identity matrix) to identify and isolate important weights simplifying the algorithm to soft-thresholding which makes the pre-calibration computationally efficient.
- The proposed pre-calibration approach ensures that the quantized model performs consistently across a variety of deployment environments, addressing the limitations of calibration-based methods in domain-specific scenarios.
- Our work introduces a new pre-calibration step that can be integrated with existing PTQ calibration methods, offering a new initial point for the calibration optimization procedure, enhancing the overall effectiveness of the PTQ proces.
- We provide a theoretical foundation for our proposed pre-calibration method using information theory and techniques from statistical machine learning.

The rest of the paper is organized as follows. In Section 2 we provide a detailed problem statement and clarify our proposed weight-adaptive pre-calibration. Section 3 reviews recent works in the field of PTQ and specifies the differences to our proposed weight-adaptive pre-calibration method. Section 4 discusses the proposed pre-calibration algorithm in detail. Section 5 delves deeper into the theoretical analysis of the algorithm and shows how pre-calibration can control information loss in quantization. Finally, experimental results supporting our proposed methodology and theoretical findings are presented in Section 6.

2 PROBLEM STATEMENT

Recently proposed PTQ methods such as (Frantar et al., 2023; Chee et al., 2023) often use $\arg\min_{\hat{\mathbf{W}}} \|\mathbf{W}\mathbf{X} - \hat{\mathbf{W}}\mathbf{X}\|_2^2$ to adjust the quantized model weights $\hat{\mathbf{W}}$ with respect to original weights $\mathbf{W}$, ensuring that the reduction in precision does not significantly degrade performance. In contrast, we propose a fundamentally different approach to PTQ notable as *pre-calibration*, which re-frames the quantization process as a classification problem on the model's weights. This approach does not rely on any calibration data, setting it apart from the conventional PTQ methods. Instead of using calibration for post-hoc adjustments, our method classifies the model's weights into quantization bins in a manner that inherently preserves the underlying distribution of the weights.

2.1 Weight Adaptive Penalization

Let us consider the following optimization problem

$$\arg\min_{\hat{\mathbf{W}}} \|\mathbf{W}\mathbf{X} - \hat{\mathbf{W}}\mathbf{X}\|_2^2 + \lambda \mathbb{D}_{\mathrm{KL}}(f_{\mathbf{W}} \| f_{\hat{\mathbf{W}}}), \qquad (1)$$

where $\mathbf{W}$ denotes original weights with $f_{\mathbf{W}}$ distribution and $\hat{\mathbf{W}}$ denotes quantized weights with $f_{\hat{\mathbf{W}}}$ distribution.

Note that problem (1) is *only* used for classification, *not* shrinkage, of weights and the penalty term, $\lambda\mathbb{D}_{\mathrm{KL}}(f_{\mathbf{W}}\|f_{\hat{\mathbf{W}}})$, is used to guide the classification in a way that the distribution of quantized weights closely follows that of the original weights.

By viewing pre-calibration as a classification problem, we can ensure that the quantization process itself is robust, reducing the need for extensive calibration afterward. This method fundamentally shifts the focus from calibration after quantization to optimizing the quantization process from the outset, thereby enhancing the robustness and generalizability of the quantized model across various tasks.

2.2 Weight Classification, Penalization, and Saliency Detection

Unlike traditional calibration-based methods that adjust quantized weights by solving $\arg\min_{\hat{\mathbf{W}}} \|\mathbf{W}\mathbf{X} - \hat{\mathbf{W}}\mathbf{X}\|_2^2$, our proposed pre-calibration method does not modify the quantized weight tensor. Instead, the optimization problem (1) and its penalty term are employed solely to classify weights into two categories: salient weights and non-salient weights.

It is important to clarify that in this context, salient weights are not simply large values. Rather, our optimization framework defines salient weights as those that cause the distribution of quantized weights to deviate significantly from the original distribution. The penalty term $\lambda\mathbb{D}_{\mathrm{KL}}(f_{\mathbf{W}}\|f_{\hat{\mathbf{W}}})$ is used specifically to ensure that the classification of weights is conducted in a way that it preserves the overall weight distribution after quantization.

2.3 Pre-Calibration and Pseudo Activations

An inherent challenge that emerges from the optimization problem (1) is that activations $\mathbf{X}$ are inherently tied to the input of a layer, implying a need for calibration. To address this issue, we remove the necessity for calibration by utilizing pseudo activations. For example, when $\mathbf{X}\mathbf{X}^\top = b\mathbf{I}$, we can leverage specific mathematical properties to simplify the KL-divergence using a straightforward soft-thresholding approach.

3 RELATED WORKS

In the field of low-precision deep learning, three existing notable categories are (i) low-precision or quantized training, (ii) quantization-aware training (QAT), and (iii) post-training quantization (PTQ). While our proposed method can be applied to both low-precision training (e.g. (Banner et al., 2018; Zhang et al., 2020; Zhu et al., 2020; Zhao et al., 2021; Ghaffari et al., 2022)) and QAT (e.g. (Zhu et al., 2023; Dettmers et al., 2024a; Liu et al., 2023)), our primary focus is PTQ of LLMs which is found to be more challenging in the literature. As such, we confine our attention to PTQ of LLMs in this section.

Historically, PTQ methods were common for computer vision models with small number of parameters, some notable methods are AdaRound (Nagel et al., 2020), OBQ (Frantar and Alistarh, 2022), AdaQuant (Hubara et al., 2021), and BRECQ (Li et al., 2021). However, these methods were found to be either compute-intensive or inaccurate for large language models.

LLM.int8() (Dettmers et al., 2022) and ZeroQuant (Yao et al., 2022) are among the first PTQ techniques that were designed for LLMs. LLM.int8() separates the outlier activations and keeps them in floating-point number format while quantizing weights and non-outlier activations to 8-bit integers. LLM.int8() separates the outlier activations based on their magnitude. On the other hand, ZeroQuant uses a fine-grained hardware-friendly quantization scheme as well as layer-by-layer knowledge distillation for quantizing both weight and activations. However, both LLM.int8() and ZeroQuant are not efficient for quantizing LLMs to extreme low-precision number formats such as 3-bit integers.

OPTQ (Frantar et al., 2023) is a PTQ algorithm for LLMs that can quantize weights to 3- or 4-bit integers. OPTQ adapted a calibration algorithm inspired by (Hassibi and Stork, 1992) that minimizes the ℓ_2 loss of the quantized layer output with the original output. SpQR (Dettmers et al., 2024b) uses OPTQ algorithm while separating the salient weights and keeping them in FP16 format and further uses double quantization to reduce the memory. Both SpQR and OPTQ algorithms require calibration data for quantization.

SmoothQuant (Xiao et al., 2023) performs 8-bit integer quantization of weights and activation by offline migration of the quantization difficulty from activations to weights. Likewise, AWQ (Lin et al., 2023), quantized weights by applying per-channel scales that protect the salient weights by observing the activation. SmoothQuant and AWQ algorithms also require calibration data to perform quantization.

QuarRot (Ashkboos et al., 2024), AQLM (Egiazarian et al., 2024) and QServe (Lin et al., 2024) are among the most recent PTQ approaches. Quarot tackles the outlier problem by rotating LLMs in a way that removes outliers from the hidden state without changing the output. AQLM generalizes the classic Additive Quantization (AQ) approach for LLMs. QServe introduces a quantization algorithm with 4-bit weight, 8-bit activation, and 4-bit KV cache. Moreover, QServe introduces SmoothAttention to effectively mitigate the accuracy degradation incurred by 4-bit KV quantization.

The key feature of our proposed pre-calibration algorithm lies in its ability to classify weights to improve quantization accuracy without performing any calibration. Furthermore, our proposed method uniquely classifies and isolates outlier weights solely through analysis of the model weight tensors ensuring more robust quantization that does not depend on the calibration dataset. While our proposed methodology is a precursor to PTQ algorithms, it can also be combined with calibration based method to improve the accuracy.

4 METHODOLOGY

The core intuition behind our approach is rooted in the goal of matching the distribution of quantized weights to that of the original weights as explained by optimization problem (1). A straightforward way to achieve this is by ensuring that each quantized weight is as close as possible to its corresponding original weight i.e. $\hat{w}_i = w_i$ or $\frac{\hat{w}_i}{w_i} - 1 = 0$ s.t. $w_i \neq 0$. This proximity naturally preserves the overall distribution, minimizing the divergence between the original and quantized weight distributions.

However, directly matching each quantized weight to its original counterpart may not always be possible, especially when the quantization process introduces significant changes in the weight values. To achieve parsimony, we may decide, according to the ”importance” of each weight, how close a quantized weight should be matched with its original counterpart. Given this basic intuition, we are led to the classification of weights. The question then arises as to how such classification should be done. To this end, we consider penalization methods for classifications where the penalty on each quantized weight is gauged and guided by its original weight, the available gold standard. One penalty that serves such purpose well is called Adaptive LASSO (Zou, 2006) in statistical machine learning literature. Adpative Lasso is the penalty of choice when a gold standard exists.

$$\arg\min_{\hat{\mathbf{W}}} \|\mathbf{W}\mathbf{X} - \hat{\mathbf{W}}\mathbf{X}\|_2^2 + \lambda \sum_i \left| \frac{\hat{w}_i}{w_i} \right|, \tag{2}$$

The mathematical proof of how problem (2) is a proxy solution to problem (1) is presented in Section 5. We emphasize that the penalization method is *only* used for classification of weights into salient and non-salient, in statistics language active and inactive, weights, *not* for shrinkage.

4.1 Pseudo Activations

A key challenge arising from the optimization problem (2) is that activations $\mathbf{X}$ are intrinsically linked to the input of a layer, suggesting a requirement for calibration. To overcome this, we eliminate the need for calibration by employing pseudo activations. Let us assume $\mathbf{X}$ is an orthogonal matrix i.e. $\mathbf{X}\mathbf{X}^\top = b\mathbf{I}$, where b is a constant and $\mathbf{I}$ is the identity matrix. By expanding (2) we have

$$\begin{aligned} \mathcal{L} &= \big(\mathbf{W}\mathbf{X} - \hat{\mathbf{W}}\mathbf{X}\big)\big(\mathbf{W}\mathbf{X} - \hat{\mathbf{W}}\mathbf{X}\big)^\top + \lambda \sum_i \left| \frac{\hat{w}_i}{w_i} \right| \qquad (3) \\ &= b\|\mathbf{W}\|_2^2 - 2b\hat{\mathbf{W}}\mathbf{W}^\top + b\|\hat{\mathbf{W}}\|_2^2 + \lambda \sum_i \left| \frac{\hat{w}_i}{w_i} \right|, \end{aligned}$$

and since $\mathbf{W}$, weights of the original model are constant, the Adaptive LASSO loss becomes

$$\begin{aligned} \mathcal{L} &= -2b\hat{\mathbf{W}}\mathbf{W}^\top + b\|\hat{\mathbf{W}}\|_2^2 + \lambda \sum_i \left| \frac{\hat{w}_i}{w_i} \right| \qquad (4) \\ &= \sum_i \Big(-2bw_i\hat{w}_i + b\hat{w}_i^2 + \lambda \left| \frac{\hat{w}_i}{w_i} \right| \Big). \end{aligned}$$

By taking the derivative with respect to $\hat{w}_i$, and setting it equal to zero, it is easy to see

$$\hat{w}_i = \mathrm{sign}(w_i)\mathrm{ReLU}(|w_i| - \frac{\lambda'}{|w_i|}), \tag{5}$$

where $\lambda' = \lambda/2b$ and ReLU() denotes the positive part i.e. $\mathrm{ReLU}(x) = \max(x, 0)$. Equation (5) shows that Adaptive LASSO is a simple soft-thresholding method that is very efficient to be implemented in the currently available commodity hardware.

4.2 Proposed Pre-Calibration Algorithm

To match the distributions of original and quantized weights using KL divergence, we employ adaptive lasso penalty term, and the combination of Adaptive LASSO with pseudo activations naturally leads to a soft-thresholding approach as shown in (5). Through using soft-thresholding for classifications of weights, we achieve a quantized model that not only retains the key characteristics of the original but also ensures robust performance across diverse tasks. Algorithm

1 shows this classification procedure. Note that in Algorithm 1, none of the weights are shrunk to zero and equation (5) is only used to classify weights to salient and non-salient weights. Here, salient weights are defined as those weights that cause the distribution of quantized weights to deviate significantly from the original distribution.

Algorithm 1: Pre-Calibration algorithm.

Input: Layer weight tensor $\mathbf{W}$, Outlier percentage α

Step 1. Start from a large $\lambda' >> 0$ in (5) then reduce it until α percent of weights are selected as outlier. (Class 1)

Step 2. Classify all other weights as common weights. (Class 2)

Step 3. Quantize Class 1 weights and Class 2 weights using minmax quantization

5 THEORETICAL CONSIDERATIONS

In this section, we establish the theoretical foundation that underpins our approach, specifically demonstrating that the Adaptive LASSO serves as a proxy solution to minimizing the KL divergence between the original and quantized weight distributions. By rigorously analyzing the relationship between adaptive lasso regularization and KL divergence, we show that the adaptive lasso effectively guides the quantization process toward preserving the original model's weight distribution.

Suppose $f_{\mathbf{W}}$ is twice continuously differentiable. Let $f'_{\mathbf{W}}$ and $f''_{\mathbf{W}}$ denote the first and second derivatives of $f_{\mathbf{W}}$. Suppose the mean μ_δ and variance σ_δ^2 of the quantization error δ are small. Then

Claim 1:

$\mathbb{D}_{\text{KL}}(f_{\mathbf{W}} \| f_{\hat{\mathbf{W}}}) \approx \mu_\delta \sum_i f'_{\mathbf{W}}(w_i) + \mu_\delta \sum_i f''_{\mathbf{W}}(w_i)(\hat{w}_i - w_i)$

Claim 2:

$\left|\mu_\delta \sum_i f''_{\mathbf{W}}(w_i)(\hat{w}_i - w_i)\right| \le C\left(\sum_i \left|\frac{\hat{w}_i}{w_i}\right| + 1\right)$ where C is a constant.

Proof: Assuming a quantization error δ_i, each original weight relates to the quantized weight such that $\hat{w}_i = w_i + \delta_i$. Let us also assume errors δ are independent of the weights values. Therefore, quantized weight distribution is a convolution of original weights distribution and quantization error distribution such that

$$f_{\hat{\mathbf{W}}}(\hat{w}) = (f_{\mathbf{W}} * f_\delta)(\hat{w}) = \int_{-\infty}^{\infty} f_{\mathbf{W}}(\hat{w} - x) f_\delta(x) dx \tag{6}$$
$$= f_{\mathbf{W}}(\hat{w}) + \int_{-\infty}^{\infty} (f_{\mathbf{W}}(\hat{w} - x) - f_{\mathbf{W}}(\hat{w})) f_\delta(x) dx.$$

Using the mean value theorem for $f_{\mathbf{W}}(\hat{w} - x) - f_{\mathbf{W}}(\hat{w})$, we have

$$f_{\hat{\mathbf{W}}}(\hat{w}) = f_{\mathbf{W}}(\hat{w}) + \int_{-\infty}^{\infty} (-x) f'_{\mathbf{W}}(\xi_{\hat{w}}(x)) f_\delta(x) dx$$
$$\overset{\sigma_\delta^2 \text{ is small}}{\approx} f_{\mathbf{W}}(\hat{w}) - \int_{-\infty}^{\infty} x f'_{\mathbf{W}}(\hat{w}) f_\delta(x) dx, \tag{7}$$

and thus,

$$\frac{f_{\hat{\mathbf{W}}}(\hat{w})}{f_{\mathbf{W}}(\hat{w})} \approx 1 - \int_{-\infty}^{\infty} \frac{x f'_{\mathbf{W}}(\hat{w})}{f_{\mathbf{W}}(\hat{w})} f_\delta(x) dx \tag{8}$$
$$= 1 - \frac{f'_{\mathbf{W}}(\hat{w})}{f_{\mathbf{W}}(\hat{w})} \int_{-\infty}^{\infty} x f_\delta(x) dx = 1 - \mu_\delta \frac{f'_{\mathbf{W}}(\hat{w})}{f_{\mathbf{W}}(\hat{w})},$$

where μ_δ is the mean of the quantization error δ. Then

$$\ln\left(\frac{f_{\hat{\mathbf{W}}}(\hat{w})}{f_{\mathbf{W}}(\hat{w})}\right) \approx \ln\left(1 - \mu_\delta \frac{f'_{\mathbf{W}}(\hat{w})}{f_{\mathbf{W}}(\hat{w})}\right) \tag{9}$$
$$\overset{|\mu_\delta| \text{ is small}}{\approx} -\mu_\delta \frac{f'_{\mathbf{W}}(\hat{w})}{f_{\mathbf{W}}(\hat{w})}$$

By plugging the equation (10) in KL divergence, we have

$$\mathbb{D}_{\text{KL}}(f_{\mathbf{W}} \| f_{\hat{\mathbf{W}}}) = -\sum_i f_{\mathbf{W}}(\hat{w}_i) \ln\left(\frac{f_{\hat{\mathbf{W}}}(\hat{w}_i)}{f_{\mathbf{W}}(\hat{w}_i)}\right) \tag{10}$$
$$\approx \mu_\delta \sum_i f'_{\mathbf{W}}(\hat{w}_i).$$

Then, it follows from Taylor's expansion around the original weight w_i, i.e. $f'_{\mathbf{W}}(\hat{w}_i) \approx f'_{\mathbf{W}}(w_i) + f''_{\mathbf{W}}(w_i)(\hat{w}_i - w_i)$ that

$$\mathbb{D}_{\text{KL}}(f_{\mathbf{W}} \| f_{\hat{\mathbf{W}}}) \approx \mu_\delta \sum_i f'_{\mathbf{W}}(w_i) + \mu_\delta \sum_i f''_{\mathbf{W}}(w_i)(\hat{w}_i - w_i), \tag{11}$$

which proves **Claim 1**.

To prove **Claim 2**, since w_i and $f''(w_i)$ are bounded, i.e. $|w_i| \le A$ and $|f''(w_i)| \le B$ in which A and B are constants, using triangular inequality

$$|\mu_\delta \sum_i f''_{\mathbf{W}}(w_i)(\hat{w}_i - w_i)| =$$
$$|\mu_\delta| \sum_i |w_i| |f''_{\mathbf{W}}(w_i)| \left|\left(\frac{\hat{w}_i}{w_i} - 1\right)\right| \le C\left(\sum_i \left|\frac{\hat{w}_i}{w_i}\right| + 1\right), \tag{12}$$

in which $C = |\mu_\delta| AB$. This completes the proof.

Since in PTQ, original weights, and their distribution are known, the first term in equation (11) is constant. Therefore, minimizing $\mathbb{D}_{\text{KL}}(f_{\mathbf{W}} \| f_{\hat{\mathbf{W}}})$ is almost like minimizing $\mu_\delta \sum_i f''_{\mathbf{W}}(w_i)(\hat{w}_i - w_i)$. Thus, following inequality (12), we may replace $\mathbb{D}_{\text{KL}}(f_{\mathbf{W}} \| f_{\hat{\mathbf{W}}})$ with $\sum_i |\frac{\hat{w}_i}{w_i}|$ in minimization problem (1) which shows Adaptive LASSO is a proxy solution to minimization problem (1).

6 EXPERIMENTAL RESULTS

This section provides experimental results supporting our proposed methodology for pre-calibration quantized LLMs. Note that in our results, we use a row-wise group quantization technique in conjunction with the soft-thresholding method as explained in Algorithm 1.

Average Bits: The average bit presented in the results of our proposed pre-calibration is calculated based on three factors, (i) the number of non-salaient weights and their bit-width, (ii) the number of salient weights and their bitwidth and (iii) location index of the salient weights. Since pre-calibration classifies the salient weights in an unstructured manner, tracking the location index is essential to deal with quantized salient and non-salient weights separately. While maintaining a mask is straightforward, it would add an extra bit per weight, which is inefficient in terms of memory consumption. To tackle this issue, we chose to retain the location index of salient weights within each group when using group quantization. Retaining the index of salient weights leads to a lower average bit since in our method the salient weight ratio α, is at most 10%. This approach results in fewer bits compared to using a mask, i.e. it requires $\log_2 g$ bits only for each salient weight where g is the group size. Moreover, we store scales and zero-points in 16-bit floating-point format. In summary, the average number of bits per weight is computed as

$$b_{\text{avg}} = \left(b_{\text{C}} + \frac{2 \times 16}{g}\right) \times (1 - \alpha) + \left(b_{\text{O}} + \log_2 g + \frac{2 \times 16}{g}\right) \times \alpha \tag{13}$$

where g is the group size, α is the percentage of outlier weights, and b_{O} and b_{C} are the bit-widths of outlier and non-outlier weights respectively.

Clipping Non-Outlier Weights: We also used clipping to further reduce the b_{avg} while maintaining the accuracy in our 3-bit results. The clipping is done because in 3-bit quantization, maintaining quantization accuracy requires a higher ratio of salient weights. On the other hand, increasing the ratio would increase b_{avg} due to index tracking of outliers. We observed that applying a clipping range of 90-95% to non-salient weights yields similar accuracy compared to increasing the salient ratio. This confirms that our proposed pre-calibration method can also be combined with other known quantization techniques to achieve better results.

Perplexity: We evaluated perplexity of quantized LLaMA models on WikiText2 (Merity et al., 2016) and C4 (Raffel et al., 2020) datasets when sequence length is 2048. Table 3 shows the results comparing perplexity scores for FP16, Round to Nearest (RTN), GPTQ (Frantar et al., 2023), AWQ(Lin et al., 2023), SpQR,(Dettmers et al., 2024b) and our proposed pre-calibration method. Pre-calibration outperforms AWQ and RTN consistently in terms of perplexity scores. Furthermore, pre-calibration exhibits perplexity scores that closely follow those of SpQR and OmniQuant, particularly for larger models. These results highlight the pre-calibration ability to achieve competitive accuracy while offering a significant advantage in terms of quantization time efficiency and robustness. These results can be used as an initial point to further optimize the model using calibration. Refer to Appendix 8 for more perplexity results on Falcon (Almazrouei et al., 2023) and OPT (Zhang et al., 2022) models.

Quantization Time: Benefiting from our simple soft-thresholding technique, our proposed pre-calibration method significantly reduces the quantization time compared to existing methods. The proposed method achieves at least 10× faster quantization speed than AWQ (Lin et al., 2023) and surpasses SpQR (Dettmers et al., 2024b) quantization time by a factor of 100× as shown in Table 2.

Zero-Shot Task Evaluation: We also evaluated the accuracy of LLaMA 1 (Touvron et al., 2023a) and LLaMA 2 (Touvron et al., 2023b) models on 5 zero-shot common-sense reasoning tasks including ARC(easy and challenge) (Clark et al., 2018), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021) and PIQA (Bisk et al., 2020) using LM Evaluation Harness (Gao et al., 2021). As shown in Table 4, our proposed pre-calibration outperforms SpQR (Dettmers et al., 2024b) in both 4-bit and 3-bit quantization, showing correctly classifying salient weight in pre-calibration step can improve the quality of PTQ to a great extent.

Table 2: Quantization time comparison.

Model	Method	Avg Bits	Quantization Time (s) ↓
LLaMA-7B	AWQ (g128)	4.25	838
	SpQR	4.63	10901
	Pre-calibration (g128, α=8%)	4.81	**57**
LLaMA-13B	AWQ (g128)	4.25	1608
	SpQR	4.63	20502
	Pre-calibration (g128, α=6%)	4.67	**116**
LLaMA-30B	AWQ (g128)	4.25	3740
	SpQR	4.63	24069
	Pre-calibration (g128, α=5%)	4.60	**470**

Table 3: Comparison of the pre-calibration perplexity results of 4-bit & 3-bit on WikiText2.

		4-bit			**3-bit**		
Model	Method	Quantization setting	Avg Bits	Wiki2 ↓	Quantization setting	Avg Bits	Wiki2 ↓
LLaMA-7B	FP16	-	16.00	5.67	-	16.00	5.67
	RTN	4bit-g128	4.25	5.96	3bit-g128	3.25	7.01
	OPTQ	4bit-g128	4.25	5.83	3bit-g128	3.25	6.58
	AWQ	4bit-g128	4.25	5.78	3bit-g128	3.25	6.35
	OmniQuant	4bit-g128	4.25	5.77	3bit-g128	3.25	6.15
	SpQR	Refer to Appendix 8	4.63	**5.73**	Refer to Appendix 8	3.98	**5.87**
	Pre-calibration	(4bit-g128, $\alpha = 8\%$)	4.81	5.78	(3bit-g128, $\alpha = 9\%$, b_O=4)	3.97	6.07
LLaMA-13B	FP16	-	16.00	5.09	-	16.00	5.09
	RTN	4bit-g128	4.25	5.25	3bit-g128	3.25	5.88
	OPTQ	4bit-g128	4.25	5.20	3bit-g128	3.25	5.70
	AWQ	4bit-g128	4.25	5.18	3bit-g128	3.25	5.52
	OmniQuant	4bit-g128	4.25	5.17	3bit-g128	3.25	5.44
	SpQR	Refer to Appendix 8	4.63	**5.13**	Refer to Appendix 8	3.97	**5.22**
	Pre-calibration	(4bit-g128, $\alpha = 6\%$)	4.67	**5.15**	(3bit-g128, $\alpha = 9\%$, b_O=4)	3.97	5.32
LLaMA-30B	FP16	-	16.00	4.10	-	16.00	4.10
	RTN	4bit-g128	4.25	4.23	3bit-g128	3.25	4.88
	OPTQ	4bit-g128	4.25	4.22	3bit-g128	3.25	4.74
	AWQ	4bit-g128	4.25	4.21	3bit-g128	3.25	4.61
	OmniQuant	4bit-g128	4.25	4.19	3bit-g128	3.25	4.56
	SpQR	Refer to Appendix 8	4.63	**4.14**	Refer to Appendix 8	3.90	**4.25**
	Pre-calibration	(4bit-g128, $\alpha = 5\%$)	4.60	**4.16**	(3bit-g128, $\alpha = 8\%$, b_O=4)	3.89	**4.31**
LLaMA2-7B	FP16	-	16.00	5.47	-	16.00	5.47
	RTN	4bit-g128	4.25	5.72	3bit-g128	3.25	6.66
	OPTQ	4bit-g128	4.25	5.61	3bit-g128	3.25	6.38
	AWQ	4bit-g128	4.25	5.60	3bit-g128	3.25	6.24
	OmniQuant	4bit-g128	4.25	5.58	3bit-g128	3.25	6.03
	SpQR	Refer to Appendix 8	4.63	**5.52**	Refer to Appendix 8	3.98	**5.66**
	Pre-calibration	(4bit-g128, $\alpha = 8\%$)	4.81	5.60	(3bit-g128, $\alpha = 9\%$, b_O=4)	3.97	5.83
LLaMA2-13B	FP16	-	16.00	4.88	-	16.00	4.88
	RTN	4bit-g128	4.25	4.98	3bit-g128	3.25	5.52
	OPTQ	4bit-g128	4.25	4.99	3bit-g128	3.25	5.42
	AWQ	4bit-g128	4.25	4.97	3bit-g128	3.25	5.32
	OmniQuant	4bit-g128	4.25	4.95	3bit-g128	3.25	5.28
	SpQR	Refer to Appendix 8	4.63	**4.92**	Refer to Appendix 8	3.96	**5.01**
	Pre-calibration	(4bit-g128, $\alpha = 6\%$)	4.67	**4.93**	(3bit-g128, $\alpha = 9\%$, b_O=4)	3.97	**5.05**

Table 4: Comparison of the pre-calibration results on zero-shot tasks using LM Evaluation Harness (Gao et al., 2021).

Model	Method	Avg Bit	ARC-c	ARC-e	HellaSwag	Winogrande	PIQA	Avg
LLaMA-7B	FP16	16	41.89	75.25	56.95	69.93	78.67	64.54
	RTN (g128)	4.25	**42.92**	74.54	56.29	70.01	78.18	64.39
	OPTQ (g128)	4.25	40.78	74.62	56.59	69.22	78.51	63.94
	AWQ (g128)	4.25	41.13	75.00	56.44	69.14	77.86	63.91
	SpQR*	4.63	41.72	75.21	56.65	69.61	**79.05**	64.45
	Pre-calibration (g128, $\alpha = 8\%$)	4.81	42.15	**75.34**	**56.72**	**70.17**	78.56	**64.59**
LLaMA-13B	FP16	16	46.42	77.36	59.88	72.69	79.16	67.19
	RTN (g128)	4.25	45.82	76.77	59.37	72.45	**79.71**	66.82
	OPTQ (g128)	4.25	**45.99**	**77.06**	59.22	**73.32**	78.94	66.91
	AWQ (g128)	4.25	**45.99**	76.89	59.42	72.53	78.78	66.72
	SpQR*	4.63	45.73	76.85	**59.70**	73.09	79.22	**66.92**
	Pre-calibration (g128, $\alpha = 6\%$)	4.67	**45.99**	76.85	59.41	73.01	78.94	66.84
LLaMA-30B	FP16	16	52.90	80.43	63.37	75.85	81.12	70.73
	RTN (g128)	4.25	52.05	**80.77**	62.89	74.19	80.58	70.10
	OPTQ (g128)	4.25	51.37	80.47	63.12	75.30	**80.79**	70.21
	AWQ (g128)	4.25	**53.41**	80.72	**63.16**	**75.45**	80.69	**70.69**
	SpQR*	4.63	51.45	80.47	63.08	74.74	80.74	70.10
	Pre-calibration (g128, $\alpha = 5\%$)	4.60	51.88	**80.77**	63.07	74.19	80.74	70.13
LLaMA-2-7B	FP16	16	43.43	76.35	57.16	69.14	78.07	64.83
	RTN (g128)	4.25	43.09	76.18	56.90	68.67	77.48	64.46
	OPTQ (g128)	4.25	41.89	74.96	56.33	69.30	**77.97**	64.09
	AWQ (g128)	4.25	42.58	75.67	56.39	68.35	77.53	64.10
	SpQR*	4.63	**44.28**	76.14	56.95	68.51	77.42	64.66
	Pre-calibration (g128, $\alpha = 8\%$)	4.81	43.17	**76.39**	**57.12**	**69.77**	**77.97**	**64.88**
LLaMA-2-13B	FP16	16	48.46	79.42	60.05	72.38	79.11	67.88
	RTN (g128)	4.25	48.12	78.83	59.74	72.69	78.67	67.61
	OPTQ (g128)	4.25	47.95	78.79	59.81	72.85	78.56	67.60
	AWQ (g128)	4.25	46.59	79.46	59.85	**73.32**	**79.05**	67.65
	SpQR*	4.63	**48.46**	**79.76**	**59.97**	71.90	78.84	67.79
	Pre-calibration (g128, $\alpha = 6\%$)	4.67	48.38	79.63	59.89	72.53	78.94	**67.87**

* Refer to Appendix 8 for quantization settings.

7 DIVERGING VIEW ON IMPROVING POST-TRAINING QUANTIZATION

Traditional post-training quantization (PTQ) methods typically follow a two-step process: quantization followed by calibration. While this approach has been effective, the calibration step often poses significant challenges, as it involves adjusting the quantized weights to recover as much of the original model's accuracy as possible. The difficulty of this task is compounded by the fact that calibration is inherently a complex optimization problem, and based on optimization theory, having a better initial point can greatly influence the outcome.

In this paper, we propose a shift in perspective by introducing a pre-calibration step prior to the traditional calibration process. We have demonstrated that pre-calibration can provide a significantly improved starting point for calibration, enhancing the overall effectiveness of the PTQ process. In fact, in some cases, this pre-calibration starting point outperforms even the final results of previously introduced calibration methods.

Another critical aspect of our approach is the *classification* of weights during quantization. While we utilized KL divergence in conjunction with Adaptive LASSO, where the Adaptive LASSO serves as a proxy solution for minimizing KL divergence, this framework is flexible. The choice of divergence measure or regularization penalty is not fixed; one could employ other f-divergence measures or adapt different penalties based on the specific needs of the PTQ method. Our proposal is not prescriptive in this regard but rather encourages the exploration of the best tools for achieving optimal *weight classification* before the PTQ procedure.

8 CONCLUSION

We presented a weight-adaptive pre-calibration approach for PTQ methods. Traditional PTQ techniques typically rely on a two-step process of quantization followed by calibration. However, the calibration step often proves challenging, as it requires careful adjustments to quantized weights to regain the model's original accuracy. We have demonstrated that pre-calibration can provide a significantly improved starting point for calibration, enhancing the overall effectiveness of the PTQ process and, in some cases, even surpassing the final performance of previously introduced calibration methods. This highlights the importance of starting with a well-prepared initial point, which can significantly impact the success of the quantization process. Our work rethinks the traditional PTQ pipeline, advocating for the integration of a pre-calibration step that enhances the starting conditions for calibration. This shift not only improves the robustness and effectiveness of the quantization process but also opens the door to further innovations in model efficiency and deployment strategies. As the demand for efficient deployment of Large Language Models (LLMs) continues to grow, our approach provides a new perspective on optimizing PTQ for diverse applications.

REFERENCES

Almazrouei, E., Alobeidli, H., Alshamsi, A., Cappelli, A., Cojocaru, R., Debbah, M., Goffinet, E., Heslow, D., Launay, J., Malartic, Q., Noune, B., Pannier, B., and Penedo, G. (2023). Falcon-40B: an open large language model with state-of-the-art performance.

Ashkboos, S., Mohtashami, A., Croci, M. L., Li, B., Jaggi, M., Alistarh, D., Hoefler, T., and Hensman, J. (2024). Quarot: Outlier-free 4-bit inference in rotated llms. *arXiv preprint arXiv:2404.00456*.

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., et al. (2021). Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Banner, R., Hubara, I., Hoffer, E., and Soudry, D. (2018). Scalable methods for 8-bit training of neural networks. *Advances in neural information processing systems*, 31.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. (2020). Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Chee, J., Cai, Y., Kuleshov, V., and De Sa, C. M. (2023). Quip: 2-bit quantization of large language models with guarantees. In Oh, A., Neumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 4396–4429. Curran Associates, Inc.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. (2021). Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. (2018). Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Computer, T. (2023). Redpajama: An open source recipe to reproduce llama training dataset.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. (2022). Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 30318–30332. Curran Associates, Inc.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. (2024a). Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Dettmers, T., Svirschevski, R. A., Egiazarian, V., Kuznedelev, D., Frantar, E., Ashkboos, S., Borzunov, A., Hoefler, T., and Alistarh, D. (2024b). SpQR: A sparse-quantized representation for near-lossless LLM weight compression. In *The Twelfth International Conference on Learning Representations*.

Egiazarian, V., Panferov, A., Kuznedelev, D., Frantar, E., Babenko, A., and Alistarh, D. (2024). Extreme compression of large language models via additive quantization. *arXiv preprint arXiv:2401.06118*.

Frantar, E. and Alistarh, D. (2022). Optimal brain compression: A framework for accurate post-training quantization and pruning. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 4475–4488. Curran Associates, Inc.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. (2023). OPTQ: accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. (2020). The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Gao, L., Tow, J., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., McDonell, K., Muennighoff, N., et al. (2021). A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*, page 8.

Ghaffari, A., Tahaei, M. S., Tayaranian, M., Asgharian, M., and Partovi Nia, V. (2022). Is integer arithmetic enough for deep learning training? *Advances in Neural Information Processing Systems*, 35:27402–27413.

Hassibi, B. and Stork, D. (1992). Second order derivatives for network pruning: Optimal brain surgeon. In Hanson, S., Cowan, J., and Giles, C., editors, *Advances in Neural Information Processing Systems*, volume 5. Morgan-Kaufmann.

Hubara, I., Nahshan, Y., Hanani, Y., Banner, R., and Soudry, D. (2021). Accurate post training quantization with small calibration sets. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4466–4475. PMLR.

Li, Y., Gong, R., Tan, X., Yang, Y., Hu, P., Zhang, Q., Yu, F., Wang, W., and Gu, S. (2021). BRECQ: pushing the limit of post-training quantization by block reconstruction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Lin, J., Tang, J., Tang, H., Yang, S., Dang, X., and Han, S. (2023). Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*.

Lin, Y., Tang, H., Yang, S., Zhang, Z., Xiao, G., Gan, C., and Han, S. (2024). Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *arXiv preprint arXiv:2405.04532*.

Liu, Z., Oguz, B., Zhao, C., Chang, E., Stock, P., Mehdad, Y., Shi, Y., Krishnamoorthi, R., and Chandra, V. (2023). Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. (2016). Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Nagel, M., Amjad, R. A., Van Baalen, M., Louizos, C., and Blankevoort, T. (2020). Up or down? Adaptive rounding for post-training quantization. In III, H. D. and Singh, A., editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7197–7206. PMLR.

Penedo, G., Malartic, Q., Hesslow, D., Cojocaru, R., Cappelli, A., Alobeidli, H., Pannier, B., Almazrouei, E., and Launay, J. (2023). The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Remez, T., Rapin, J., et al. (2023). Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. (2021). Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023a). Llama: Open and efficient foundation language models (2023). *arXiv preprint arXiv:2302.13971*.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023b). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. (2023). SmoothQuant: Accurate and efficient post-training quantization for large language models. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38087–38099. PMLR.

Yao, Z., Yazdani Aminabadi, R., Zhang, M., Wu, X., Li, C., and He, Y. (2022). Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27168–27183. Curran Associates, Inc.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. (2019). Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. (2022). Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zhang, X., Liu, S., Zhang, R., Liu, C., Huang, D., Zhou, S., Guo, J., Guo, Q., Du, Z., Zhi, T., et al. (2020). Fixed-point back-propagation training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2330–2338.

Zhao, K., Huang, S., Pan, P., Li, Y., Zhang, Y., Gu, Z., and Xu, Y. (2021). Distribution adaptive int8 quantization for training cnns. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*.

Zhu, F., Gong, R., Yu, F., Liu, X., Wang, Y., Li, Z., Yang, X., and Yan, J. (2020). Towards unified int8 training for convolutional neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1969–1979.

Zhu, X., Li, J., Liu, Y., Ma, C., and Wang, W. (2023). A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633*.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429.

APPENDIX

Experimental Settings

Seed Sensitivity

Since our proposed method, only uses deterministic pre-trained weights of the model and performs a soft-thresholding to identify α percent of outlier weights, it does not exhibit any stochastic behavior during the quantization. Furthermore, we do not use any data for calibration and thus, our algorithm is robust toward randomness in data selection. We believe this is the main advantage of our proposed algorithm.

Calibration Datasets and Parameters

We follow the pipelines used in SpQR[1] and AWQ[2] official implementation to generate calibration datasets. Random selection of 128 samples of length 2048 form RedPajama (Computer, 2023), C4 and Refined-Web (Penedo et al., 2023) is used for quantization of LLaMA 1, LLaMa 2, OPT (Zhang et al., 2022) and Falcon (Almazrouei et al., 2023) using SpQR. For AWQ experiments 128 samples from a small subset of Pile (Gao et al., 2020) dataset is used following the AWQ's implementation.

Hyper-Parameters and Configs

RTN: We implemented RTN quantization method based on the implementation of AWQ which supports weight reshaping for group quantization.
AWQ: We used AWQ's official implementation for quantizing LLaMA and OPT models.
SpQR: We use SpQR's official implementation for quantizing LLaMA, Code-Llama and OPT models. Table 5 shows the hyper-parameters used for SpQR quantization.

Table 5: Quantization configuration of SpQR.

Model	Calibration Set	Group Size	Weight Bits	Scales & Zeros Bits	Outlier Threshold
LLaMA	RedPajama	16	4	3	0.2
	RedPajama	16	3	3	0.25-0.28
Code-Llama	RedPajama	16	4	3	0.2
OPT	C4	16	4	3	0.2
Falcon	RefinedWeb	16	4	3	0.2

Hardware Settings

We perform quantization on single NVIDIA V100-32G GPU. For evaluation using LM Evaluation Harness we use 8×V100-32G GPUs for 30B models.

Extra Experimental Results

Table 6 shows the perplexity results on OPT (Zhang et al., 2022) and Falcon(Almazrouei et al., 2023) models.

[1] See https://github.com/Vahe1994/SpQR

[2] See https://github.com/mit-han-lab/llm-awq

Table 6: Perplexity of 4-bit OPT and Falcon models on WikiText2 and C4.

Model	Method	Quantization setting	Avg Bits	Wiki2 ↓	C4↓
OPT-6.7B	FP16	-	16.00	10.86	11.74
	RTN	4bit-g128	4.25	11.15	12.31
	AWQ	4bit-g128	4.25	10.95	11.86
	SpQR	Refer to Appendix 8	4.63	10.91	11.78
	Pre-calibration	(4bit-g128, $\alpha = 6\%$)	4.67	**10.86**	11.99
OPT-13B	FP16	-	16.00	10.13	11.20
	RTN	4bit-g128	4.25	10.30	11.51
	AWQ	4bit-g128	4.25	10.29	11.28
	SpQR	Refer to Appendix 8	4.27	**10.22**	**11.27**
	Pre-calibration	(4bit-g128, $\alpha = 6\%$)	4.67	**10.20**	**11.31**
OPT-30B	FP16	-	16.00	9.55	10.69
	RTN	4bit-g128	4.25	9.94	10.94
	AWQ	4bit-g128	4.25	9.61	10.74
	SpQR	Refer to Appendix 8	4.63	9.55	10.71
	Pre-calibration	(4bit-g128, $\alpha = 5\%$)	4.60	9.64	10.79
Falcon-7B	FP16	-	16.00	6.59	9.50
	RTN	4bit-g128	4.25	6.79	9.79
	SpQR	Refer to Appendix 8	4.44	**6.64**	**9.58**
	Pre-calibration	(4bit-g128, $\alpha = 4\%$)	4.53	**6.69**	**9.63**
Falcon-40B	FP16	-	16.00	5.23	7.76
	RTN	4bit-g128	4.25	5.31	7.88
	SpQR	Refer to Appendix 8	4.46	**5.26**	**7.79**
	Pre-calibration	(4bit-g128, $\alpha = 5\%$)	4.60	**5.27**	**7.81**