Mitigating Quantization Errors Due to Activation Spikes in GLU-Based LLMs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Modern large language models (LLMs) have established state-of-the-art performance through architectural improvements, but still require significant computational cost for inference. In an effort to reduce the inference cost, post-training quantization (PTQ) has become a popular approach, quantizing weights and activations to lower precision, such as INT8. In this paper, we reveal the challenges of activation quantization in GLU variants [40], which are widely used in feed-forward network (FFN) of modern LLMs, such as LLaMA family. The problem is that severe local quantization errors, caused by excessive magnitudes of activation in GLU variants, significantly degrade the performance of the quantized LLM. We denote these activations as *activation spikes*. Our further observations provide a systematic pattern of activation spikes: 1) The activation spikes occur in the FFN of specific layers, particularly in the early and late layers, 2) The activation spikes are dedicated to a couple of tokens, rather than being shared across a sequence. Based on our observations, we propose two empirical methods, Quantization-free Module (QFeM) and Quantization-free Prefix (QFeP), to isolate the activation spikes during quantization. Our extensive experiments validate the effectiveness of the proposed methods for the activation quantization, especially with coarse-grained scheme, of latest LLMs with GLU variants, including LLaMA-2/3, Mistral, Mixtral, SOLAR, and Gemma. In particular, our methods enhance the current alleviation techniques (e.g., SmoothQuant) that fail to control the activation spikes.[1]

1 Introduction

Large language models (LLMs) have become a key paradigm in natural language processing, accelerating the release of variations within the community [49, 58]. Furthermore, latest LLMs establish state-of-the-art performance by training with increased scale, as well as by adopting architectural improvements such as GLU [40], RoPE [41], GQA [2], and MoE [21]. Especially, GLU (Gated Linear Unit) variants (e.g., SwiGLU, GeGLU) has been adopted in the most of modern LLM architectures (e.g., LLaMA family [46]), due to training efficiency [31, 40]. Although LLMs broaden foundational capabilities in natural language tasks and potential for various applications, billions of parameters in the large models impose considerable computational costs on end users in practice. To reduce GPU memory requirements and accelerate inference speed, post-training quantization (PTQ) offers an affordable solution by quantizing weights and activations into a lower precision (e.g., INT8) without a need for expensive retraining steps [17, 19, 30]. However, recent studies have revealed that large magnitude values at certain coordinates exist in the activations of LLMs, which are often called outliers, posing a key challenge in activation quantization [1, 12, 50, 51]. Another line of works attempts to explain the role of outlier values in the attention mechanism [9, 42]. Nevertheless, current research on the impact of evolving LLM architectures on the outliers remains insufficient.

[1] Code is available at `https://anonymous.4open.science/r/activation-spikes-EDF0`.

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

In this paper, we present our discovery that the GLU architecture in the feed-forward network (FFN) generates excessively large activation values, which are responsible for significant local quantization errors. Specifically, we observe that these problematic activation values occur in specific linear layers and are dedicated to a couple of tokens, which will be discussed in Section 3. To distinguish the excessive GLU activations from the outliers, we refer to them as *activation spikes*. In light of our observations, we propose two empirical methods to mitigate the impact of activation spikes on quantization: Quantization-free Module (QFeM) and Quantization-free Prefix (QFeP). QFeM aims to partially exclude quantization for linear layers (or modules) where large quantization errors occur, instead of quantizing the entire linear modules in the LLM. By scoring the extent of scale disparity, QFeM selects linear modules to exclude. On the other hand, QFeP identifies the prefix that triggers activation spikes and preserves its context as a key-value (KV) cache, thereby preventing the recurrence of activation spikes in subsequent tokens. It is noteworthy that both QFeM and QFeP rely on calibration results to capture activation spikes in advance, without any modifications to the target LLM. This indicates that our methods can be integrated into any existing quantization methods.

In our comprehensive experiments, we demonstrate that recently released LLMs incorporating GLU variants struggle with activation spikes when applying activation quantization. Consequently, the proposed methods, QFeM and QFeP, substantially enhance the performance of the primitive quantization method, the round-to-nearest (RTN) method. Furthermore, we observe that current outlier alleviation methods [50, 51] are exposed to the activation spikes and benefit from our proposed methods. Compared to the strong baseline of fine-grained activation quantization [55], our methods show competitive performance, achieving reduced latency and memory footprint.

In summary, the contributions of our work are as follows:

- We find that the GLU architecture in modern LLMs systematically generates excessive activation values, which are responsible for significant performance degradation in activation quantization.
- Based on our observations, we propose two empirical methods, QFeM and QFeP, which effectively exclude the activation spikes during quantization, with negligible computational overhead and compatibility with any existing quantization techniques.
- Our extensive experimental results validate the detrimental impact of the activation spikes on activation quantization, while our proposed methods consistently enhance the quantization performance.

2 Related Works

Outlier Values in LLMs. Previously, outlier values have been observed in the transformer-based language models such as BERT [14] and early GPT [36] models through numerous studies [8, 24, 27, 35, 45]. Since the advent of LLMs [10, 57] rooted in the GPT, recent studies by [1, 12, 51] have tackled the existence of outlier values in LLMs. According to them, these outliers exhibit a large magnitude of values at the shared dimensions of hidden states across tokens. More recently, [9, 42] explain that the outliers attribute to the vertical pattern in the attention mechanism [25, 52], which influences the performance of LLMs. In particular, [42] claims a different type of outlier existing in the hidden states of specific tokens. However, prior studies merely focus on the superficial hidden states between the decoder layers. Our work provides a module-level investigation where quantization is applied practically, focusing on different LLM architectures.

Post-training Quantization for LLMs. Post-training quantization (PTQ) refers to the quantization of a neural network model to low precision, such as INT8, without additional parameter updates [17, 19]. Especially for LLMs, this approach cost-effectively achieves inference with low memory usage and faster inference latency by quantizing the weights and activations used in matrix multiplication (e.g., linear layer). However, because of the challenges in activation quantization of LLMs, many recent works are mainly focused on the weight-only quantization [11, 13, 15, 23, 26, 39, 54]. Otherwise, the activation quantization faces inherent outliers, which hinder accurate quantization by reducing representation resolution. To address this challenge, [12] proposes a mixed-precision quantization method where the outlier dimensions are computed in high precision. [50, 51] approach migration of scale from activation to weights to alleviate the scale of outlier activations. Along this line of research, we propose to enhance the activation quantization based on our observations.

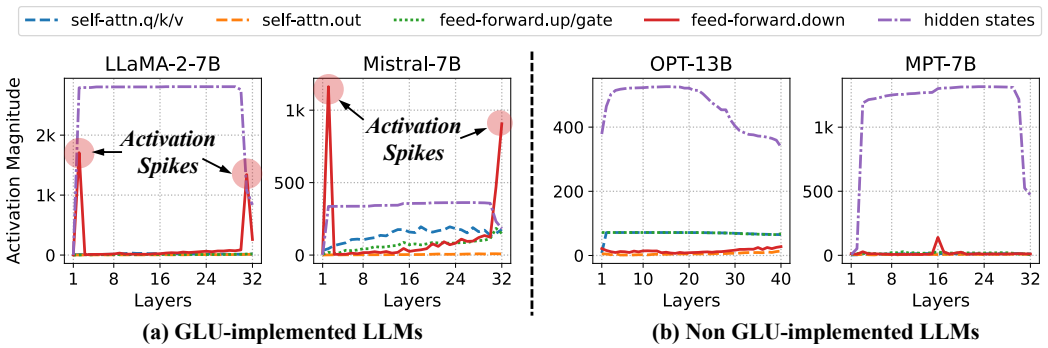

Figure 1: Calibration results on GLU-implemented and non GLU-implemented LLMs. We present the maximum magnitudes of input activations for each linear modules and layer-wise hidden states. For more results on different LLMs, see Appendix A.2, A.3.

3 Activation Spikes: Excessive Magnitude of GLU Activations

For clarity, "hidden states" refer to the output tensor of a transformer layer (or block), while "input activations" or "activations" denote the input tensor of a linear layer (or module) in the remain of this paper. Recent work [42] has investigated a novel type of outlier existing in the hidden states across modern LLMs. Although these outliers of hidden states play a crucial role in the attention mechanism [9, 42, 52], their relationship with input activations for quantization has not been fully explored. Importantly, because recent LLMs adopt Pre-LN [4, 53], which normalizes hidden states before self-attention and feed-forward network (FFN) blocks, the scale of hidden states does not reflect the scale of input activations within the transformer block. Therefore, we focus on the input activations fed into each linear module within the transformer block to connect to activation quantization. Specifically, we examine the four linear (projection) layers: `query` (parallel to `key` and `value`), `out`, `up` (parallel to `gate`), and `down` modules. For detailed illustration of Pre-LN transformer, please see Appendix D.1.

3.1 Existence of Activation Spikes in GLU Variants

To analyze the input activations, we employ a calibration method, which is used to estimate the quantization factors such as scale and zero-point. For the calibration data, we use 512 samples randomly collected from the C4 [37] training dataset. Afterwards, we feed each sample into the LLM and monitor each hidden state and input activation through the decoder layers. To estimate the scale factor, we use absolute maximum value. The tested LLMs are listed in Appendix A.1.

GLU-implemented LLMs exhibit activation spikes at specific layers. In Figure 1a, we display the calibrated scale factors for the LLMs that implement GLU variants (e.g., SwiGLU, GeGLU). Across models, we observe a shared pattern of scale from the results. Within the early and late layers, the `down` modules in the FFN show noticeable magnitudes of input activations. Note that these input activations are derived from the *Hadamard Product* within GLU. Thus, the GLU variants generate activation spikes at the specific layers. Interestingly, we notice a high correlation between the emergence of activation spikes and intermediate hidden states of large scale. This indicates that the FFN contributes to amplifying the hidden states via the addition operation in the residual connection [18]. Once the magnitude of the hidden states is exploded, it persists through layers until encounter the activation spikes at late layers.

Non GLU-implemented LLMs show modest scale distribution. Figure 1b illustrates the calibration results for LLMs with the original feed-forward implementation in Transformer [48]. We observe that the LLMs continue to generate the large-scale hidden states, regardless of the GLU implementation. This corresponds to the observations in [42]. More importantly, our module-level results elaborate that the scale of hidden states is not transferable to the input activations of inner linear modules. Instead, we reveal that GLU variants are associated with the hidden states and generate activation spikes. This clarifies the quantization challenge of the GLU-implemented LLMs concentrated in the early and late layers. Because excessive scales of activation spikes have the potential to hinder the accurate quantization, we conduct an in-depth analysis to better understand these activation spikes in the following sections.

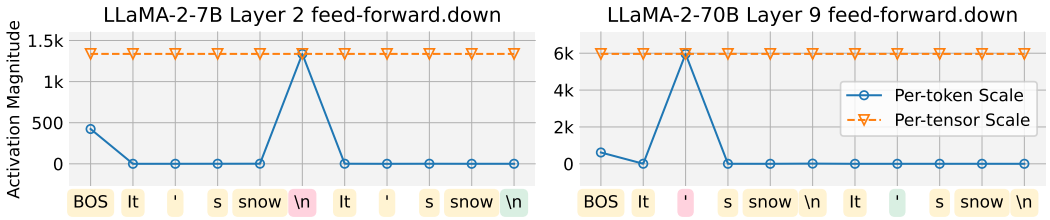

Figure 2: Token-wise scales in a specific layer with an activation spike. When quantizing the input activations using a per-tensor scale, the scale of the activation spike dominates the scales of the other tokens. For more examples, see Appendix D.2.

3.2 Token-level Scale Analysis within Activation Spikes

In the previous section, we observed the excessive scale of the input activations derived from GLU activation. When quantizing the input activations, the variance of input activation scales for each token affects the quantization performance [55]. To delve into the disparity between token-wise scales in the activation spikes, we unroll them through the sequence of tokens. Figure 2 illustrates the individual input activation scales where the activation spike appears. Given a token sequence, the large magnitudes of input activations are observed in a couple of tokens, such as the BOS token, newline (`\n`), and apostrophe (`'`). These specific tokens coincide with the observations of [42], which suggests that such tokens exhibit massive values in the hidden states. Thus, the activation spike is associated with the process of assigning a special role to these tokens in later transformer layers. However, the excessive scale of specific token hinders the estimation of scale factor for the other tokens, such as in per-tensor quantization. Additionally, the largest scale is dedicated to the first instance of the specified token, while the following usage exhibits a modest scale. This phenomenon makes the quantization more complicated, as the activation spikes dynamically occur depending on the current input sequence.

3.3 Effect of Quantization on Activation Spikes

We explore the impact of local quantization errors caused by activation spikes on LLM outputs. To identify the layers where activation spikes occur, we utilize a ratio between the maximum and median values of the token-wise input activation scales, instead of using the maximum scale value alone. The max-median ratio for linear layer m can be formulated as $r^{(m)} = \frac{\max(\mathbf{S}^{(m)})}{\text{median}(\mathbf{S}^{(m)})}$, where $S^{(m)}$ represents the token-wise input activation scales incoming to module $m \in M$. This max-median ratio captures the extent to which maximum scale dominate the other token scales. For comparison, we choose the activation quantization targets as the top-4, middle-4, and bottom-4 modules, based on the max-median ratio in descending order. Then, we evaluate the perplexity and mean-squared error (MSE) using the calibration dataset. Here, the MSE is calculated for the last hidden states between the original (FP16) and partially quantized LLM. As shown in Table 1, quantization on the top-4 rated modules solely degrades the LLM performance by significant margins, while the other cases exhibit negligible performance changes. We consider these quantization-sensitive input activations (*inter alia* activation spikes) to be the quantization bottleneck, which, in this paper, refers to the quantization error caused by outliers.

Furthermore, the activation spikes are conditioned on the specific context of the input sequence as discussed in Section 3.2. Altogether, such dynamic bottlenecks must be handled with caution to enhance the quantization performance of LLMs.

Table 1: Perplexity and MSE of partial activation quantization of LLMs

Model	Perplexity(↓)				MSE(↓)		
	FP16	Top 4	Middle 4	Bottom 4	Top 4	Middle 4	Bottom 4
LLaMA-2-7B	7.37	**11.77**	7.38	7.40	**1908.80**	1.03	12.90
LLaMA-2-13B	6.84	**15.09**	6.84	6.84	**4762.11**	0.91	10.38
Mistral-7B	8.35	**69.45**	8.35	8.36	**218.60**	0.02	0.18
Gemma-7B	10.85	**85.83**	10.94	10.87	**213.93**	1.60	1.07

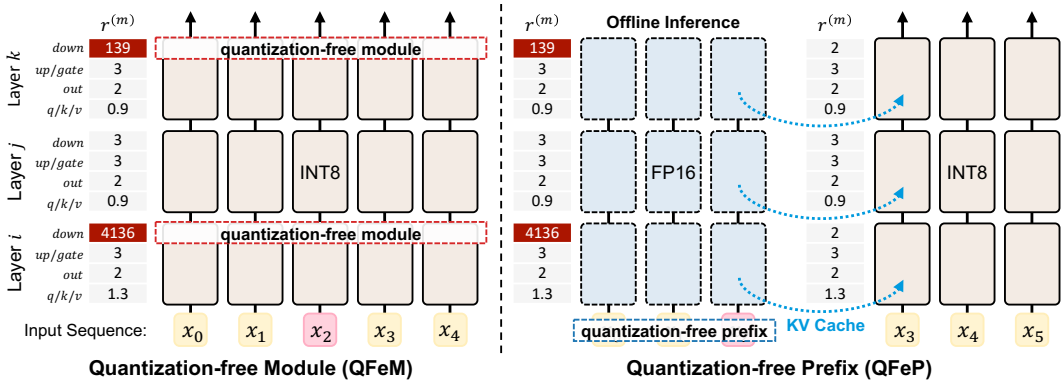

Figure 3: Overview of QFeM and QFeP. (Left): QFeM excludes the modules whose $r^{(m)}$ is larger than the hyperparameter α from quantization. (Right): QFeP computes in advance the prefix of activation spikes and utilizes solely their KV cache during the quantization phase, effectively preventing further activation spikes in subsequent sequences.

4 Mitigating Quantization Quality Degradation Based on the Observation

To address the quantization bottleneck, our approach is based on the deterministic occurrence patterns of activation spikes. First, we utilize the observation that bottlenecks occur at a few specific layers. This implies that naive full quantization of LLMs is affected by these bottlenecks. Second, we exploit the phenomenon that the activation spike is derived from the first occurrence of specific tokens. Thus, the planned occurrence prevents recurrence in the subsequent and possibly future tokens. In the following sections, we propose two methods inspired the above insights.

4.1 Quantization-free Module (QFeM)

In the full quantization of LLM, all linear layers within the LLM are quantized. Among these linear layers, we propose omitting the quantization of input activations for linear layers where significant quantization errors are caused by activation spikes. To be noted, increasing the number of unquantized modules exhibits a trade-off between the inference latency and the model performance. Thus, determining which module should be quantized (or left unquantized) is crucial to retain the efficacy of quantization. Here, we use the max-median ratio $r^{(m)}$ and define a set of unquantized modules, denoted as M_{unq}, where the ratio $r^{(m)}$ of each linear layer is larger than threshold α. For instance, all linear layers in M are quantized if $\alpha = \infty$. For clarity, we treat sibling linear layers, such as query-key-value, as a single linear layer. To control the impact of activation quantization only, we leave the weight parameters in unquantized linear layers as INT8 and dequantize them into FP16 during matrix multiplication with the incoming activations, operating as weight-only quantization.

Optimizing the threshold α. To calculate the activation scale ratio for each linear layer, we first gather token-wise input activation scales from the calibration examples discussed in Section 3.1. Exceptionally, for FFN experts in the mixture of experts (MoE) architectures like the Mixtral model [21], calibration is performed separately. After determining these ratios, we use binary search to set the threshold value α, balancing inference latency and performance degradation. As a metric, we assess performance through perplexity measured on the same calibration examples. For example, the relationship between threshold value α and its impact on performance is depicted in Figure 4, demonstrating how full quantization can degrade performance. Rather than fully quantizing, we identify an optimal threshold by finding the intersection of two performance curves; in Figure 4, this threshold is approximately 16. Details on the QFeM implementation are provided in Table 2.

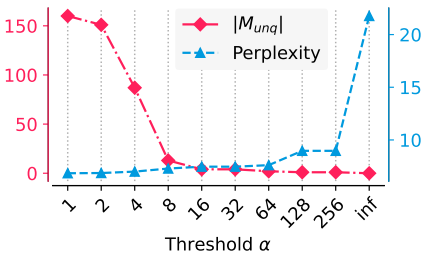

Figure 4: Trade-off between perplexity (stands for performance) and $|M_{unq}|$ (stands for latency) according to the threshold α for LLaMA-2-13B model.

4.2 Quantization-free Prefix (QFeP)

Orthogonal to the QFeM, we propose Quantization-free Prefix (QFeP) that mitigates the quantization errors by precomputing the prefix (or short prompt) corresponding to activation spikes. This method is based on the observations presented in Section 3.2, which indicate that significant quantization errors result from the overestimated scale factor of the *first instance* within the restricted token set. Inspired by this occurrence pattern of activation spikes, we aim to construct a prefix which stabilizes the quantization scale factor of the tokens that come after the prefix. In other words, once the prefix is fixed at the beginning, the activation spikes consistently occur within the prefix. Afterward, we employ key-value (KV) caching mechanism to process the activation spikes in advance. In practice, KV cache is utilized to optimize the decoding speed of causal language models by storing precomputed key and value states of the previous tokens [32, 34]. This approach provides a bypass of the quantization including activation spikes, while preserving the context of prefix through the KV cache. The KV cache for the prefix is precomputed once through the offline inference of LLM without quantization. Then, this KV cache is exploited in the quantization phases, such as calibration or dynamic quantization, even for quantized inference. The process of QFeP is illustrated in Figure 3.

Prefix Search. To form a prefix of explicit activation spike, we first identify candidate token that represent the activation spike at the linear layer with the highest max-median ratio $r^{(m)}$. For instance, the candidate token can be apostrophe (') token for LLaMA-2-70B model, as highlighted in red in Figure 2. Once the candidate token is identified, we search the middle context token for between the BOS token and the candidate token in the prefix. This middle context provides dummy context, which is required to activate the candidate token. To find the middle context, we design a template $[B, T_1, C_1, T_2, C_2]$ where B, T_i, and C_i denote the BOS token, context token, and candidate token in the vocabulary V, respectively. Then, we select the context token T where C_1 triggers an activation spikes, while later instance of the same token C_2 does not. When the context token for the activation spikes is varied, we choose the token that maximizes the activation scale ratio between the C_1 and C_2. Finally, we prepare the KV cache for searched prefix of $[B, T, C]$. Note that the latter sequence in the template can be replaced with sequences from dataset instead of repetition.

Implementation Details. During the prefix search phase, we exploit the calibration dataset used in Section 3.1. For the candidate tokens, we consider the tokens with the top three largest input activation magnitudes. Then, we search for the middle context token among top 200 most frequent tokens in the calibration dataset, which is the subset of the vocabulary V. Finally, with the search result, we prepare the KV cache for the target model in FP16 precision. Exceptionally, for the Mixtral [21] model, we use the scale of output hidden states instead of input activations, as the tokens are divided sparsely in a mixture of experts architecture. Table 2 presents the searched prefix.

Table 2: Specifications for QFeM and QFeP used in experiments. $|M|$ denotes the total number of linear layers in the LLM, and $|M_{unq}|$ represents the number of unquantized layers for QFeM.

Model	Prefix	α	$\lvert M_{unq}\rvert/\lvert M\rvert$
LLaMA-2-7B	[BOS] all .	6.68	17 / 128
LLaMA-2-13B	[BOS] then ,	12.91	6 / 160
LLaMA-2-70B	[BOS] I '	9.16	25 / 320
Mistral-7B	[BOS] how \n	49.00	3 / 128
Mixtral-8x7B	[BOS]). \n	4.03	191 / 608
SOLAR-10.7B	[BOS] a 1	6.48	11 / 192
Gemma-7B	[BOS] . Più	10.65	5 / 112
LLaMA-3-8B	[BOS] - nd	6.64	6 / 128
LLaMA-3-70B	[BOS] and ,	78.37	3 / 320

5 Experiments

5.1 Experimental Setup

Models. Our proposed methods, QFeM and QFeP, aim to mitigate the quantization bottleneck, which is discussed in Section 3.3, caused by the activation spikes, especially in the GLU variants. To validate the efficiency proposed methods, we tested publicly released LLMs that were implemented with GLU, according to their paper and source code. We recognize recent LLMs, including LLAMA-2-{7B, 13B, 70B} [47], LLaMA-3-{7B, 70B}, Mistral-7B [20], Mixtral-8x7B [21], SOLAR-10.7B [22], and Gemma-7B [43], utilize the GLU architecture. The LLMs with original FFN are not covered, as they suffer from the existing outliers rather than activation spikes. All models are sourced from the huggingface-hub[2] repository.

[2]https://huggingface.co/models

Table 3: Perplexity and zero-shot evaluation for the quantization on LLaMA-2 models. FP16 denotes the original model precision, and W8A8 denotes the model quantized to INT8 for both weights and activations.

Method	WikiText-2 (ppl↓)	PIQA (acc↑)	LAMBADA (acc↑)	HellaSwag (acc↑)	WinoGrande (acc↑)	Avg (acc↑)
LLaMA-2-7B						
FP16	5.268	78.18%	73.67%	57.13%	69.46%	69.61%
W8A8	8.634	72.80%	62.27%	49.57%	63.69%	62.08%
+QFeM	5.758[**-2.876**]	78.02%	73.86%	56.32%	68.35%	69.14%[**+7.06**]
+QFeP	5.758[**-2.876**]	76.44%	73.57%	55.55%	69.22%	68.69%[**+6.61**]
+QFeM+QFeP	5.573[**-3.061**]	77.86%	74.58%	56.05%	69.38%	69.47%[**+7.39**]
LLaMA-2-13B						
FP16	4.789	79.49%	76.54%	60.20%	72.38%	72.15%
W8A8	34.089	70.13%	49.66%	42.65%	58.72%	55.29%
+QFeM	5.241[**-28.848**]	77.58%	75.68%	59.13%	72.61%	71.25%[**+15.96**]
+QFeP	6.000[**-28.089**]	77.53%	73.94%	57.23%	70.96%	69.91%[**+14.62**]
+QFeM+QFeP	5.126[**-28.963**]	78.51%	75.86%	59.44%	72.61%	71.61%[**+16.32**]
LLaMA-2-70B						
FP16	3.218	81.45%	79.45%	65.29%	80.43%	76.65%
W8A8	8.055	74.05%	70.27%	55.21%	67.96%	66.87%
+QFeM	3.830[**-4.225**]	81.23%	77.66%	64.15%	78.14%	75.30%[**+8.43**]
+QFeP	6.007[**-2.048**]	77.64%	73.26%	63.40%	76.16%	72.62%[**+5.75**]
+QFeM+QFeP	3.708[**-4.347**]	81.23%	77.82%	64.65%	77.11%	75.20%[**+8.33**]

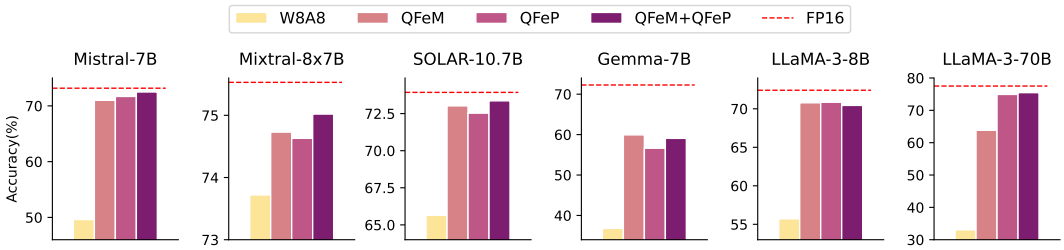

Figure 5: The average accuracy of zero-shot evaluation on other GLU-implemented LLMs. Most models recover significantly compared to W8A8, with performance close to FP16.

Quantization. In the experiments, we quantize both the input activations and the weights of linear layers for INT8 matrix multiplication operations. Note that in Table 2, $|M|$ denotes the total number of linear modules targeted for quantization. In these linear layers, we opt for dynamic per-tensor quantization as the quantization scheme of input activations, and per-channel quantization for weights, respectively. Regarding both input activations and weights, we symmetrically quantize the range using the absolute maximum value as the scale estimation function. For comparison, we use FP16 and per-token activation quantization [55] as baselines. We refer the reader to Appendix B for Batch Matrix-Multiplication (BMM) quantization, which involves quantizing tensors in the self-attention.

Evaluations. We evaluate the quantized LLMs with two metrics: zero-shot evaluation accuracy and perplexity. For zero-shot evaluation, we use the four datasets: PIQA [7], LAMBADA [33], HellaSwag [56], and WinoGrande [38]. We utilize the lm-evaluation-harness library [16] to evaluate zero-shot tasks. To measure perplexity, we use the WikiText-2 [28] dataset. In all cases, we use the [BOS] token as the starting token for each input sequence by default.

5.2 Main Results

LLaMA-2 Models. We report the evaluation results of quantization on LLaMA-2 models in Table 3. Compared to FP16 precision, quantizing both weights and activations (W8A8) degrades the overall performance. The results demonstrate that our proposed methods resolve the activation spikes and, surprisingly, restore the performance of the W8A8 close to that of FP16. For example, the LLaMA-2 7B model achieves less than a 1% performance drop from FP16. It is worth noting that the

Table 4: Evaluation of outlier alleviation methods with QFeM and QFeP. We report perplexity on WikiText-2 and averaged accuracy of four zero-shot tasks. The same quantization scheme for used on both SQ and OSP. Per-tensor weight quantization results are provided in Appendix C.1.

Method	LLaMA-2-7B		LLaMA-2-13B		LLaMA-2-70B	
	ppl(↓)	acc(↑)	ppl(↓)	acc(↑)	ppl(↓)	acc(↑)
SQ [51]	9.907	61.08%	34.869	59.45%	8.800	70.25%
+QFeM	**5.534**	**69.65%**	**5.118**	**71.23%**	**3.599**	**75.93%**
+QFeP	5.715	68.66%	6.551	69.33%	5.228	74.07%
OSP [50]	38.490	59.90%	5.148	71.29%	3.827	75.52%
+QFeM	**5.493**	**69.37%**	**5.099**	**71.37%**	**3.559**	**75.92%**
+QFeP	5.642	68.95%	5.144	71.05%	3.752	75.36%

proposed QFeM and QFeP improve at comparable levels. This indicates that the activation spikes present a direct cause of the significant decrease in quantization performance. Because the proposed methods are orthogonal, the performance slightly increases when incorporating both QFeM and QFeP compared to applying them individually.

Other GLU-implemented LLMs. For other LLMs that incorporate GLU, we investigated the effectiveness of our methods in mitigating the quantization bottleneck. As can be seen in Figure 5, our methods consistently remedy the performance drop caused by activation spikes. Noticeably, the Mixtral model demonstrates robustness towards the performance degradation. This indicates that the mixture of experts architecture, which divides the MLP experts by tokens, helps to alleviate the impact of the activation spikes. Meanwhile, addressing the activation spikes is not a sufficient complement for the Gemma model compared to other models. We attribute this to the choice of activation function among GLU variants; specifically, Gemma uses GeGLU, while other models employ SwiGLU.

5.3 Combining Outlier Alleviation Methods

While our method focuses on the activation spikes, the inherent outlier values in the input activations remain. Here, we combine the prior outlier alleviation methods, such as SmoothQuant (SQ) [51] and OutlierSuppressionPlus (OSP) [50], to further improve the quantization error. In practice, our methods are utilized during the scale calibration phase of alleviation methods to mitigate the impact of activation spikes on scale migration between activations and weights. Table 4 demonstrates the evaluation results of applying the outlier alleviation methods solely and combining them with our methods. We find that there are cases where the alleviation method fails to recover the performance when quantizing the activations with per-tensor scheme.[3] This indicates that alleviating the outlier scales, including the activation spikes, is challenging. With the QFeM, the activation spikes are excluded, and the accurate alleviation is enabled. In addition, the QFeP also benefits from the SQ method, as seen in the case of LLaMA-2 70B. Exceptionally, the OSP successfully addresses the activation spikes in the 13B and 70B cases.

5.4 Ablation Study

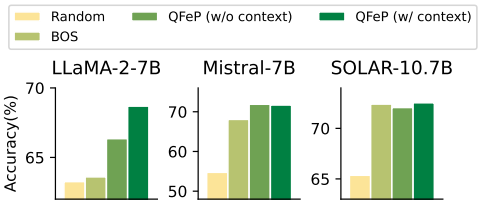

Figure 6: Prefix ablation. Y-axis represents averaged accuracy of four zero-shot tasks.

For the QFeP, we designed a length-three prefix for the KV cache, including the BOS token, context token, and extra token for activation spike. Because the KV cache consumes the capacity of the pretrained sequence position, it raises a question about the length of the prefix. Therefore, we conduct ablation study for different prefixes for the KV cache. For the prefixes, we prepare random, BOS only, and both QFeP without and with the context token. We illustrate the results of ablation study in Figure 6. In all cases, the random prefix showcases the lowest performance. While the KV cache with the BOS token demonstrates inconsistent performance, our QFeP

[3] In their papers, the activations of LLaMA models are quantized using only a per-token scheme.

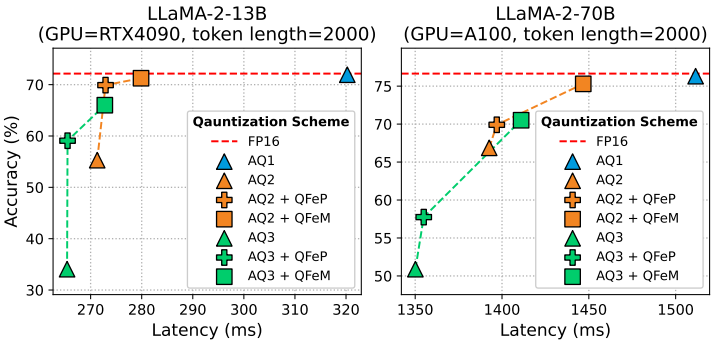

Figure 7: Accuracy-latency comparison of different activation quantization schemes: dynamic per-token (AQ1), dynamic per-tensor (AQ2), and static per-tensor (AQ3).

Table 5: Memory footprint.

Method	SeqLen 1K	SeqLen 2K
LLaMA-2-7B		
AQ1	8185MiB	9516MiB
AQ2	8148MiB	9474MiB
+QFeP	8149MiB	9478MiB
+QFeM	8148MiB	9474MiB
LLaMA-2-70B		
AQ1	67756MiB	69037MiB
AQ2	67648MiB	68820MiB
+QFeP	67651MiB	68822MiB
+QFeM	67838MiB	68819MiB

consistently shows significant improvement. Importantly, the results imply that the sufficient prefix for the models exhibits differences. However, we emphasize that our KV design for QFeP shows improvements by large margins across all models.

5.5 Computational Cost Analysis

The proposed methods require additional resources to evict the activation spikes. Therefore, we analyze the computational costs of the methods and compare them in various schemes. For comparison, we evaluate different activation quantization schemes: dynamic per-token, dynamic per-tensor, and static per-tensor, denoted as AQ1, AQ2, and AQ3, respectively. This distinction establishes strong baselines and demonstrates the potential of the methods. To calibrate the static scales, we estimate the absolute maximum value using the calibration dataset, which is used in Section 3.1.

Inference Latency. For each setting, we present the accuracy of the zero-shot tasks and inference latency of the fixed token sequence, as shown in Figure 7. While the fine-grained scheme (AQ1) shows a negligible accuracy drop, the counterparts (AQ2, AQ3) degrade with the quantization bottleneck. However, by applying our methods, the coarse-grained schemes achieve a competitive performance gain. For example, the combination of AQ2 and QFeM demonstrates the performance close to the AQ1 but with faster latency. The results signify that addressing the quantization bottleneck is important to accelerate the inference latency with coarser granularity. Specifically, the naive static quantization (AQ3), the fastest scheme, exhibits a significant decline. We hope that our work contributes to the future works, which address the remaining challenges in static quantization.

Memory Footprint. In Table 5, we record the maximum memory footprint of our methods. For QFeP, the additional memory is consistently required for the preserved KV cache. However, this memory overhead is much smaller than that used in the fine-grained quantization (AQ1), as QFeM utilizes only three tokens for the cache. Contrary to QFeP, QFeM shows inconsistent memory utilization. For example, the 7B model with QFeM exhibits memory usage similar to AQ2, while the 70B model with QFeM incur additional consumption for a sequence length of 1K. This is attributed to the use of W8A16 for the unquantization modules in QFeM. To tailor the memory usage or inference speed, an alternative strategy can be utilized for QFeM, such as applying fine-grained activation quantization to the unquantization modules instead of using W8A16.

6 Conclusion

We explore the quantization challenge of GLU activations for modern LLMs. We find that the GLU variants generates excessive activation scales, which cause significant quantization bottlenecks at the specific layers. Based on the systematic generation pattern of the activation spikes, we propose methods that address the spikes in a layer-wise (QFeM) and token-wise manner (QFeP). In the experiments, we confirm that the proposed methods effectively resolve the quantization bottlenecks and result in a large performance gain. We expect that our work sheds light on the potential challenges in future studies regarding quantization and facilitates the development of efficient LLM systems.

References

[1] Arash Ahmadian, Saurabh Dash, Hongyu Chen, Bharat Venkitesh, Zhen Stephen Gou, Phil Blunsom, Ahmet Üstün, and Sara Hooker. Intriguing properties of quantization at scale. *Advances in Neural Information Processing Systems*, 36:34278–34294, 2023.

[2] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

[3] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*, 2023.

[4] Alexei Baevski and Michael Auli. Adaptive input representations for neural language modeling. In *International Conference on Learning Representations*, 2018.

[5] Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, et al. Stable lm 2 1.6 b technical report. *arXiv preprint arXiv:2402.17834*, 2024.

[6] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

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

[8] Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Understanding and overcoming the challenges of efficient transformer quantization. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7947–7969, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[9] Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Quantizable transformers: Removing outliers by helping attention heads do nothing. *Advances in Neural Information Processing Systems*, 36, 2024.

[10] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[11] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M De Sa. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems*, 36, 2024.

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

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

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[15] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

[16] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 12 2023.

[17] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*, pages 291–326. Chapman and Hall/CRC, 2022.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016.

[19] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.

[20] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

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

[22] Dahyun Kim, Chanjun Park, Sanghoon Kim, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, et al. Solar 10.7 b: Scaling large language models with simple yet effective depth up-scaling. *arXiv preprint arXiv:2312.15166*, 2023.

[23] Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization. *arXiv preprint arXiv:2306.07629*, 2023.

[24] Olga Kovaleva, Saurabh Kulshreshtha, Anna Rogers, and Anna Rumshisky. BERT busters: Outlier dimensions that disrupt transformers. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3392–3405, Online, August 2021. Association for Computational Linguistics.

[25] Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. Revealing the dark secrets of BERT. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China, November 2019. Association for Computational Linguistics.

[26] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*, 2023.

[27] Ziyang Luo, Artur Kulmizev, and Xiaoxi Mao. Positional artefacts propagate through masked language model embeddings. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5312–5327, Online, August 2021. Association for Computational Linguistics.

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

[29] Javaheripi Mojan and Bubeck Sébastien. Phi-2: The surprising power of small language models, 2023.

[30] Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart Van Baalen, and Tijmen Blankevoort. A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295*, 2021.

[31] Sharan Narang, Hyung Won Chung, Yi Tay, Liam Fedus, Thibault Fevry, Michael Matena, Karishma Malkan, Noah Fiedel, Noam Shazeer, Zhenzhong Lan, Yanqi Zhou, Wei Li, Nan Ding, Jake Marcus, Adam Roberts, and Colin Raffel. Do transformer modifications transfer across implementations and applications? In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5758–5773, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[32] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*, 2019.

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

[34] Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems*, 5, 2023.

[35] Giovanni Puccetti, Anna Rogers, Aleksandr Drozd, and Felice Dell'Orletta. Outlier dimensions that disrupt transformers are driven by frequency. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1286–1304, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[36] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[37] 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. *Journal of machine learning research*, 21(140):1–67, 2020.

[38] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.

[39] Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*, 2023.

[40] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[41] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

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

[43] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[44] MosaicML NLP Team. Introducing mpt-7b: A new standard for open-source, commercially usable llms, 2023. Accessed: 2023-05-05.

[45] William Timkey and Marten van Schijndel. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4527–4546, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[46] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

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

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[49] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[50] Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier suppression+: Accurate quantization of large language models by equivalent and effective shifting and scaling. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1648–1665, Singapore, December 2023. Association for Computational Linguistics.

[51] 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 *Proceedings of the 40th International Conference on Machine Learning*, 2023.

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

[53] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pages 10524–10533. PMLR, 2020.

[54] Zhewei Yao, Cheng Li, Xiaoxia Wu, Stephen Youn, and Yuxiong He. A comprehensive study on post-training quantization for large language models. *arXiv preprint arXiv:2303.08302*, 2023.

[55] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183, 2022.

[56] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics.

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

[58] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

A Additional Calibration Results

In this section, we provide details of LLMs when performing calibration, which is the step during quantization where the FP16 ranges are computed (Appendix A.1), and additional calibration results (Appendix A.2, A.3).

A.1 Detailed Specification of LLMs

In Section 3.1, we have performed the calibration method on various LLMs. We observe the calibration results by categorizing based on the presence of GLU in the LLMs. Table 6 shows the detailed structures of the LLMs. We refer notations for feed-forward implementiation from [40]. In the case of GLU-implemented LLMs, which is LLaMA-2, LLaMA-3, Mistral, Mixtral, SOLAR, StableLM-2, and Gemma, most models have SwiGLU for FFN activation, while only Gemma has GeGLU. On the other hand, in non GLU-implemented LLMs, most of them utilize GeLU for FFN activation, with the exception of OPT, which uses ReLU.

Table 6: Architecture specification of LLMs. We categorize them into two groups depending on whether GLU is implemented in the FFN. All LLMs in the table use Pre-LN for the LayerNorm position.

Model	Size	FFN Activation	Normalization	PE	Vocabulary Size
GLU-implemented LLMs:					
LLaMA-2 [47]	7B, 13B, 70B	SwiGLU	RMSNorm	RoPE	32000
LLaMA-3	8B, 70B	SwiGLU	RMSNorm	RoPE	128256
Mistral [20]	7B	SwiGLU	RMSNorm	RoPE	32000
Mixtral [21]	8x7B	SwiGLU	RMSNorm	RoPE	32000
SOLAR [22]	10.7B	SwiGLU	RMSNorm	RoPE	32000
StableLM-2 [5]	12B	SwiGLU	LayerNorm	RoPE	100352
Gemma [43]	7B	GeGLU	RMSNorm	RoPE	256000
Non GLU-implemented LLMs:					
OPT [57]	6.7B, 13B, 30B, 66B	ReLU	LayerNorm	Learned	50272
MPT [44]	7B, 30B	GeLU	LayerNorm	ALiBi	50432
Pythia [6]	6.9B, 12B	GeLU	LayerNorm	RoPE	50432, 50688
Falcon [3]	7B, 40B	GeLU	LayerNorm	RoPE	65024
Phi-2 [29]	2.7B	GeLU	LayerNorm	RoPE	51200

A.2 Other Calibration Results on GLU-implementation

Figure 8, 9 show the calibration result examples for various GLU-implemented LLMs that are not shown in the models in Figure 1a. In most GLU-implemented LLMs, we observe that the input activations have large values near the first and last layers. Unlike the typical GLU-implemented LLM architecture, Mixtral is composed of 8 feed-forward blocks in the single FFN, containing multiple gate linear units [21]. According to this structure, we can observe that one of the gates spikes in value in Figure 8.

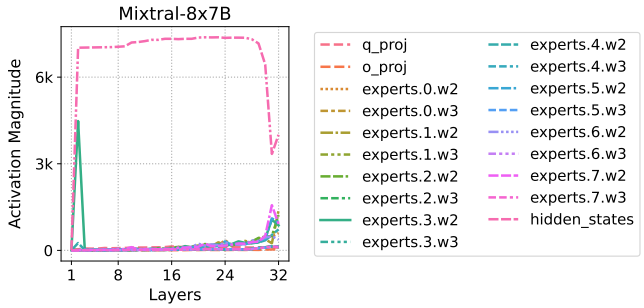

Figure 8: **Calibration results on GLU-implemented LLMs (Mixtral-8x7B).**

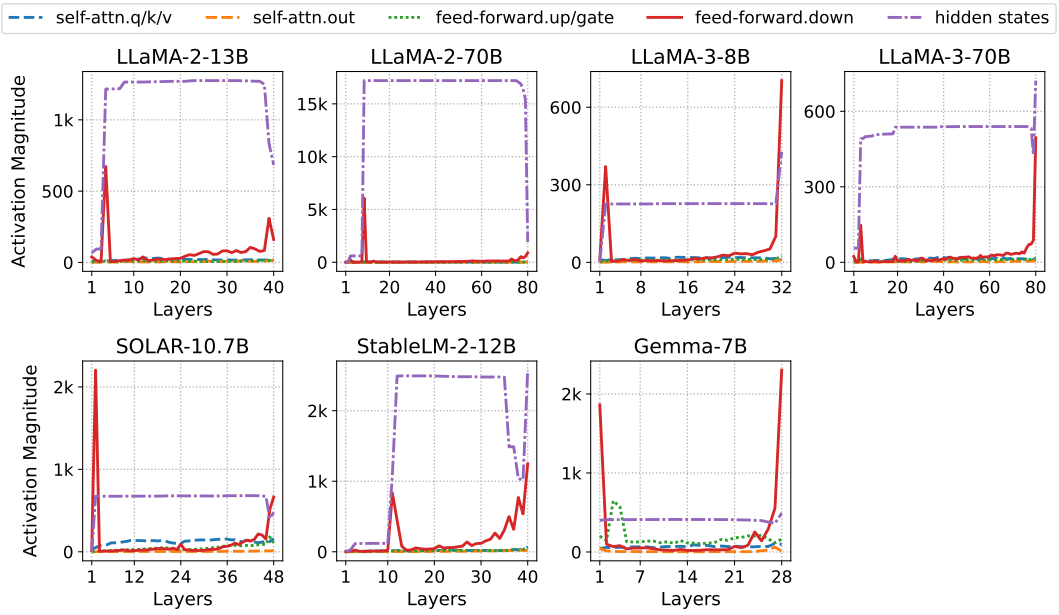

Figure 9: **Calibration results on GLU-implemented LLMs.**

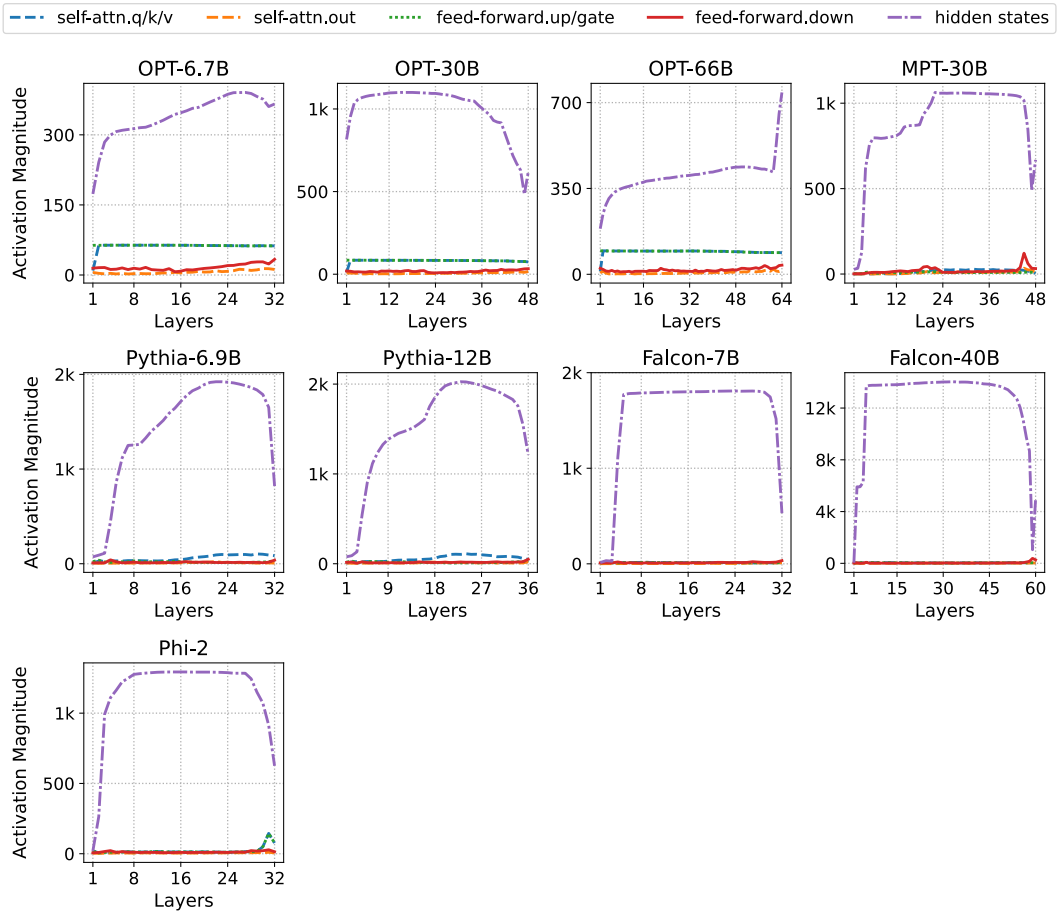

Figure 10: **Calibration results on Non GLU-implemented LLMs.**

A.3 Other Calibration Results on Non GLU-implementation

Figure 10 shows the calibration result examples for various non GLU-implemented LLMs that were not shown in the models in Figure 1b. There are no activation spikes on non GLU-implemented LLMs.

B BMM Quantization

To achieve faster inference latency, BMM operations in the self-attention also can be computed as INT8 operation [51]. This requires a quantization on the query, key, and value states including the cached context. Because activation spikes produce a large magnitude of latent values, it is important to confirm the extent of quantization errors from KV quantization. This confirmation is necessary to gain advantages from BMM quantization. In Table 7, we examine the impact of BMM quantization on the W8A8 and QFeM. Regardless of the BMM quantization, the QFeM method consistently improves the quantization bottleneck. For example, the 13B and 70B models maintain their performance, while the 7B model shows a slight decrease. However, this decrease appears to be due to inherent quantization errors rather than a quantization bottleneck from activation spikes. As a result, we confirm that our QFeM method effectively improves the overall performance even in the BMM quantization scenario.

Table 7: BMM quantization results.

Model	Method	BMM Quantization	
		No	Yes
7B	W8A8	62.08%	61.66%
	+QFeP	68.69%	68.30%
13B	W8A8	55.29%	55.43%
	+QFeP	69.91%	69.77%
70B	W8A8	66.87%	66.75%
	+QFeP	72.62%	72.69%

C Supplementary Experiment Results

C.1 Additional Results for Combining Outlier Alleviation Methods

In Table 8, we provide additional results for Section 5.3 with coarse-grained quantization (i.e., per-tensor quantization) scheme for weight quantization. Compared to the results obtained with per-channel weight quantization in Table 4, these results elucidate the negative impact of activation spikes on the performance of outlier alleviation methods. Furthermore, this suggests that the performance of OSP method resort to the weight quantization scheme. Nevertheless, the proposed methods, QFeM and QFeP, consistently improve the effectiveness of outlier alleviation methods by mitigating the impact of activation spikes.

Table 8: Evaluation of outlier alleviation methods with QFeM and QFeP. We report perplexity on WikiText-2 and averaged accuracy of four zero-shot tasks. Compared to Table 4, per-tensor weight quantization and dynamic per-tensor activation quantization are used.

Method	LLaMA-2-7B		LLaMA-2-13B		LLaMA-2-70B	
	ppl(↓)	acc(↑)	ppl(↓)	acc(↑)	ppl(↓)	acc(↑)
SQ [51]	24.661	56.87%	120.966	53.06%	8.435	67.08%
+QFeM	**6.016**	**67.74%**	**5.464**	**70.04%**	**4.015**	**74.18%**
+QFeP	6.122	67.22%	10.473	68.17%	5.998	72.54%
OSP [50]	9.131	63.61%	8.997	64.03%	6.492	71.13%
+QFeM	5.951	**68.65%**	**5.284**	**70.67%**	**4.434**	73.30%
+QFeP	**5.821**	68.25%	5.868	67.96%	4.976	**73.57%**

D Miscellaneous

D.1 Transformer Architecture.

In Figure 11, we illustrate the Pre-LN transformer architecture and each sub-modules. We highlight with the same color the linear modules that accept identical input activations. Note that the hidden states are normalized before forwarding into the `query` and `up` linear modules.

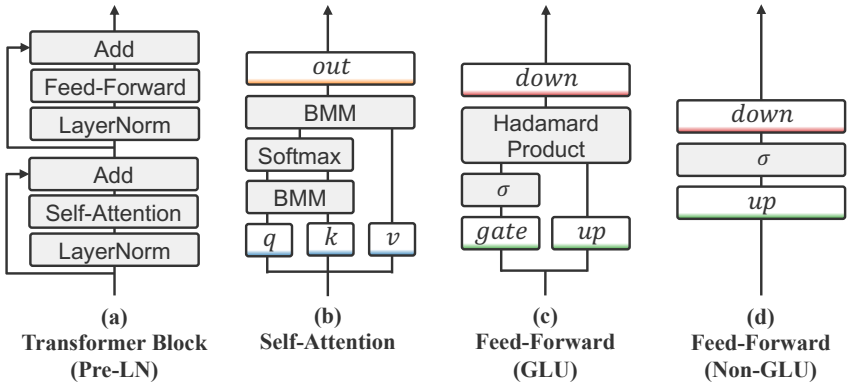

Figure 11: An illustration of Pre-LN transformer block and its sub-modules. Two feed-forward implementation, GLU and Non-GLU, are visualized in (c) and (d) respectively. In feed-forward network, σ denotes non-linear activation function, such as GeLU. We highlight the linear modules where input activations are quantized.

D.2 Additional Results for Token-level Scale Analysis

We provide additional results for token-level scale analysis (Section 3.2). In Figure 12 and Figure 13, the token for the activation spikes behind the BOS token does not exhibit the excessive activation scale.

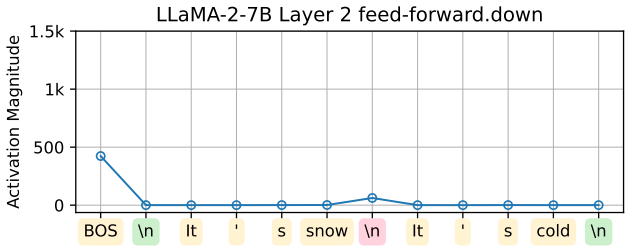

Figure 12: Token-wise scales analysis for LLaMA-2-7B. The newline token behind the BOS token does not exhibit the activation spikes.

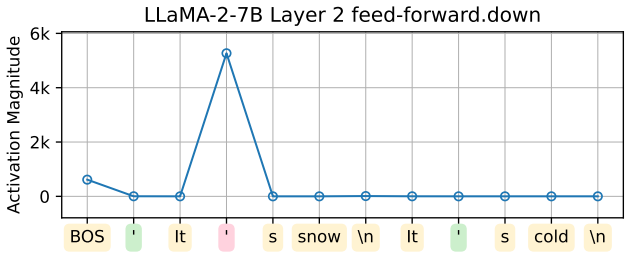

Figure 13: Token-wise scales from the unrolled activation spike of LLaMA-2-70B. The newline token behind the BOS token does not exhibit the activation spikes.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clarify our research scope and contributions in abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [No]

   Justification: The limitation of our work is that our methods are based on the observations without theoretical validation. However, our extensive experimental results validate the effectiveness of our methods.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We propose empirical methods based on our observation, rather than theoretical analysis.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We precisely describe the process of the proposed methods in their respective subsections. The models (LLMs) and datasets used in the experiments are publicly accessible.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide accessible URL in the abstract.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the hyperparameter settings in Table 2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The proposed methods rely on the sample size of the calibration dataset. Nevertheless, we are convinced that the sample size used in the experiments is sufficient for achieving reliable and consistent calibration results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide a computational cost analysis in Section 5.5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the models and dataset used in the experiments (see Section 5.1).

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.