Beyond Fixed-Length Calibration for Post-Training Compression of LLMs

Jaehoon Oh and Dokwan Oh
Samsung Advanced Institute of Technology
Suwon, South Korea
{jh0104.oh, dokwan.oh}@samsung.com

Abstract

As large language models (LLMs) continue to grow in size, their practical deployment increasingly relies on a range of compression techniques, such as quantization, pruning, and low-rank approximation. Especially, post-training compression methods–which do not require re-training–have drawn considerable interest. Many recent methods leverage calibration data to capture magnitude or second-order characteristics of input activations. However, the role and significance of calibration data remain underexplored. In this study, we demonstrate that the sequence length of calibration data plays a crucial role in the effectiveness of post-training compression methods for LLMs. We then analyze input activations and find that, within the normalized hidden states, the embedding of the first token exhibits characteristics opposite to those of subsequent tokens. Building on this insight, we introduce *state-aware length calibration*, a technique that applies masking along the sequence axis, specifically targeting normalized hidden states. Experimental results show that our approach improves perplexity and zero-shot downstream tasks performance.

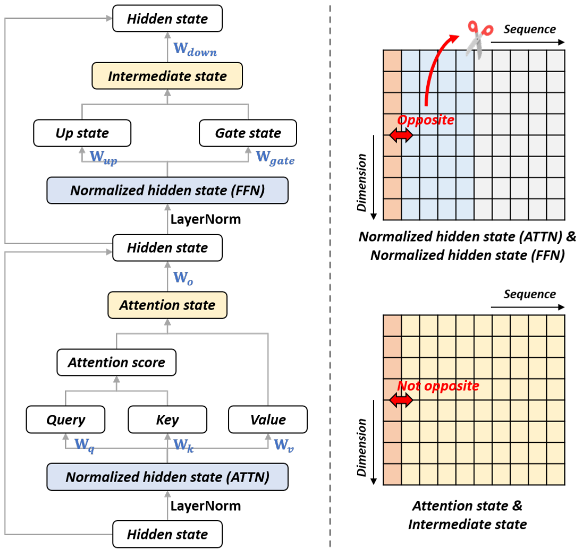


Figure 1: (Left) Illustration of a single layer with seven distinct linear weights $\mathbf{W}$ and their corresponding input activations (i.e., states). (Right) Two normalized hidden states display a tendency in which the embeddings of the first token and the subsequent tokens are oriented in opposite directions. As a result, we leverage only a partial representation of these states when calibrating LLMs, called *state-aware length calibration*. Conversely, because the attention state and intermediate state do not show this opposing orientation, we leverage them directly without modification. ATTN and FFN indicate attention and feed-forward network modules, respectively.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language processing tasks, including text generation, translation, and question answering (Achiam et al., 2023; Touvron et al., 2023; Dubey et al., 2024; Abdin et al., 2024; Jiang et al., 2023). Nevertheless, their real-world deployment remains challenging due to their substantial computational and memory overhead (Gholami et al., 2024). To overcome these limitations, researchers have developed various compression techniques designed to reduce model size and enhance inference efficiency, such as quantization, pruning, and low-rank approximation (Wan et al., 2024).

Notably, post-training compression methods have gained significant attention due to their ability to optimize model efficiency without requiring costly and time-consuming retraining. These techniques enable the direct compression of trained models, enhancing their practicality for deployment on resource-constrained devices, such as mobile devices and edge computing platforms. For instance, quantization methods, such as SmoothQuant (Xiao et al., 2023) and AWQ (Lin et al., 2024a), reduce numerical precision to lower computational costs. Pruning methods, such as

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 19355–19366
November 4-9, 2025 ©2025 Association for Computational Linguistics

SparseGPT (Frantar and Alistarh, 2023) and Wanda (Sun et al., 2024b), focus on identifying and removing redundant or less important weights from the model weights. Additionally, low-rank approximation methods, such as ASVD (Yuan et al., 2025) and SVD-LLM (Wang et al., 2025), decompose weight matrices into lower-dimensional forms.

In the context of post-training compression framework, Frantar and Alistarh (2022) introduced a layer-wise optimization approach, where each layer is optimized independently to minimize performance degradation. Specifically, the global compression task is decomposed into a series of layer-wise subproblems, with each layer being optimized separately. This process leverages input activations derived from calibration data. In particular, calibration data is used to estimate input activation magnitudes or Hessian (i.e., second-order) information, which is essential for achieving a more advanced and accurate compression process.

Despite its crucial role, calibration data remains an underexplored aspect of post-training compression. For example, calibration data is typically sourced from randomly selected web text or pre-training data, and constrained to a fixed length (e.g., 2048 tokens), under the observations that post-training compression methods are resilient to variations in calibration data distribution (Sun et al., 2024b; Li et al., 2023). However, Williams and Aletras (2024) conducted a systematic analysis of the influence of calibration data on compression performance. They revealed that the source and seed of the calibration data can largely influence on the results of both post-training quantization and pruning. Additionally, Lee et al. (2023) stated that it is necessary to align the sequence length of the calibration data with that of the target task.

In this paper, we present a systematic analysis of the impact of the *sequence length of calibration data* on post-training compression. Our analysis involves four different LLMs, six post-training compression algorithms, and two calibration datasets to thoroughly evaluate how varying the sequence length influences the compressed model performance. Our main contributions are as follows:

- (Section 4) Contrary to common intuition, we demonstrate that using shorter (< 2048) calibration data generally yields better compression performance, for most existing post-training compression algorithms.
- (Section 5) We reveal that within both the attention and feed-forward modules, the normalized hidden states exhibit an opposite relationship between the embedding of the first token and those of subsequent tokens.
- (Section 6) Building on this key observation, we propose *state-aware length calibration*, as illustrated in Figure 1. This approach leverages only a short sequence of normalized hidden states to enhance the effectiveness of post-training compression.

2 Related Work

2.1 LLM Compression

Model compression techniques for LLMs have been extensively studied to reduce computational costs and memory usage while maintaining performance. Quantization is a widely used method that reduces the precision of weights, activations, and KV caches (Xiao et al., 2023; Lin et al., 2024a; Frantar et al., 2023; Lin et al., 2024b; Dettmers et al., 2022; Wang et al., 2023). Pruning identifies and removes less significant weights to create a sparser model (Frantar and Alistarh, 2023; Sun et al., 2024b; Ma et al., 2023; Xia et al., 2024). Low-rank approximation leverages matrix factorization techniques to reduce the parameter count while preserving model expressiveness (Yuan et al., 2025; Wang et al., 2025; Li et al., 2023; Hsu et al., 2022). For a broader perspective on model compression, readers are encouraged to explore surveys such as Wan et al. (2024).

Post-training compression does not require retraining; instead, it relies on calibration data. For example, input activations of calibration data can be used to derive first-order (i.e., $\mathbf{X}$) or second-order (i.e., $\mathbf{X}\mathbf{X}^\top$) statistics. While quantization, pruning, and low-rank approximation use different strategies to minimize compression loss, they fundamentally share a common reliance on calibration to either (1) identify weights sensitive to compression (Lin et al., 2024a; Sun et al., 2024b) or (2) adjust those weights accordingly (Lin et al., 2024a; Xiao et al., 2023; Frantar and Alistarh, 2023; Yuan et al., 2025; Wang et al., 2025).

2.2 Calibration Data for Compression

Most previous studies use a small amount of calibration data, typically 128 examples, with a fixed sequence length (e.g., 2048 tokens). Williams and Aletras (2024) investigated the impact of calibration data on the performance of LLMs when applying post-training quantization and pruning methods. They found that the effectiveness of these compression techniques can vary significantly depending on the source and randomness of the calibration dataset.

Similar to ours, Lee et al. (2023) introduced sequence-length-aware calibration, which ensures that the sequence length of the target application task matches that of the post-training calibration dataset. Specifically, in zero-shot CommonSenseQA, where sequence lengths range from tens to hundreds, OPTQ (Frantar et al., 2023) achieves better performance when the sequence lengths are aligned (e.g., 128) compared to when they are mismatched (e.g., 2048). We examine whether this trend holds across various post-training techniques with different sequence lengths.

3 Experiments Setup

LLMs. We utilize four LLMs for our experiments: Llama2-7B (Touvron et al., 2023), Llama3-8B (Dubey et al., 2024), Mistral-7B-v0.3 (Jiang et al., 2023), and Phi3.5-mini-instruct (3.8B) (Abdin et al., 2024).

Post-training Compressions. We examine six recent post-training compression methods. Most algorithms use calibration data to make weights more amenable to compression, such as scaling.

- SmoothQuant[1] (Xiao et al., 2023) is a representative weight-activation co-quantization algorithm that mitigates the quantization difficulty from activations to weights by using per-channel scaling. For channel j, scaling factor s_j is defined as $\max(|\mathbf{X}_j|)^\alpha / \max(|\mathbf{W}_j|)^{1-\alpha}$. We quantize both weights and activations in 8-bits, and set α to 0.7.
- AWQ[2] (Lin et al., 2024a) is a representative weight-only quantization algorithm that preserves salient weights. Scaling factor s is simply defined as $\text{mean}(|\mathbf{X}|)^\alpha$ and then α is searched by fast grid search to minimize quantization error in a layer-wise manner. We only quantize weights in 4-bits.
- SparseGPT[3] (Frantar and Alistarh, 2023) is a second-order (un)structured pruning algorithm based on inverse Hessian, where the weight importance S_{ij} is defined as $[|\mathbf{W}|^2/\text{diag}[(\mathbf{X}\mathbf{X}^\top + \lambda\mathbf{I})^{-1}]]_{ij}$. Then, this information is used to choose a pruning mask and optimize the unpruned weights.
- Wanda[4] (Sun et al., 2024b) is a simple (un)structured pruning algorithm without weight update, unlike SparseGPT. In this algorithm, the weight importance S_{ij} is defined as $|\mathbf{W}_{ij}| \cdot \|\mathbf{X}_j\|_2$, then low-scored weights are pruned.
- ASVD[5] (Yuan et al., 2025) is a simple low-rank approximation algorithm, where a scaling matrix S is used to transform the weights more decomposition-friendly. S is defined as the diagonal matrix of the averaged magnitude, similar to AWQ. However, α is not searched; rather, it is determined as a hyper-parameter. We set α to 0.5, following the original paper. Furthermore, ASVD uses a sensitivity-based rank search method that accounts for variations in singular values across different layers.
- ASVD+[5] (Yuan et al., 2025) extends ASVD by changing the transform matrix S into Hessian-based. In detail, S is defined as a lower triangular matrix of Cholesky decomposition of $\mathbf{X}\mathbf{X}^\top$. This transform matrix guarantees a lower output error, that is explained in SVD-LLM (Wang et al., 2025).

Calibration Data. We use two calibration datasets: WikiText-2 (Merity et al., 2016) train set and Pile (Gao et al., 2020) validation set. We set the number of samples to 256 and do not use the *bos* token as the first token (i.e., at position 0). We adjust the sequence length, spanning from an extreme minimum (e.g., 1) to a standard setting (e.g., 2048).

Evaluation. We calculate the perplexity on the WikiText-2 (Merity et al., 2016) validation set using a sequence length of 2048. The results are averaged over 3 runs. The original perplexities on WikiText-2 of Llama2-7B, Llama-3-8B, Mistral-7B-v0.3, and Phi3.5-mini-instruct are 5.472, 6.138, 5.318, and 6.195, respectively.

[1]https://github.com/mit-han-lab/smoothquant
[2]https://github.com/mit-han-lab/llm-awq
[3]https://github.com/IST-DASLab/sparsegpt
[4]https://github.com/locuslab/wanda
[5]https://github.com/hahnyuan/ASVD4LLM

Method	Dataset	Model	Sequence Length							
			1	4	16	64	256	512	1024	2048
SmoothQuant (W8A8)	Pile	Llama2-7B	7.897	**5.508**	5.511	5.515	5.520	5.523	5.525	5.528
		Llama3-8B	8.753	**6.252**	6.260	6.263	6.259	6.266	6.263	6.261
		Mistral-7B	5.358	**5.345**	**5.345**	5.348	5.350	5.350	5.350	5.349
		Phi3.5-mini	6.626	**6.420**	6.436	6.469	6.469	6.475	6.472	6.488
	WikiText	Llama2-7B	7.920	**5.507**	5.511	5.515	5.519	5.521	5.524	5.525
		Llama3-8B	9.368	**6.254**	6.257	6.262	6.258	6.259	6.263	6.261
		Mistral-7B	5.354	5.347	**5.345**	5.348	5.350	5.348	5.350	5.349
		Phi3.5-mini	8.508	**6.414**	6.433	6.466	6.466	6.464	6.476	6.476
AWQ (W4A16)	Pile	Llama2-7B	6.988	5.653	5.661	5.637	5.633	5.634	**5.628**	5.634
		Llama3-8B	11.057	6.841	**6.816**	6.900	6.876	6.847	6.836	6.920
		Mistral-7B	5.589	5.469	5.461	5.457	**5.450**	5.458	5.458	5.451
		Phi3.5-mini	7.771	6.772	6.764	6.753	6.762	6.749	6.606	**6.579**
	WikiText	Llama2-7B	7.174	5.659	**5.641**	5.647	5.644	5.643	5.648	5.657
		Llama3-8B	11.421	**6.846**	6.884	6.896	6.947	7.014	7.025	6.979
		Mistral-7B	5.598	5.464	5.461	5.464	5.454	5.463	5.461	**5.435**
		Phi3.5-mini	10.498	6.764	6.773	6.772	6.789	6.788	**6.573**	6.581

Table 1: Perplexity of post-training quantization methods.

Method	Dataset	Model	Sequence Length							
			1	4	16	64	256	512	1024	2048
SparseGPT	Pile	Llama2-7B	10.818	8.154	8.139	8.030	**8.023**	8.045	8.041	8.049
		Llama3-8B	81.680	14.877	**12.956**	13.257	13.413	13.547	13.711	13.777
		Mistral-7B	8.006	7.579	**7.482**	7.657	7.697	7.655	7.640	7.659
		Phi3.5-mini	19.698	**15.401**	17.707	17.999	18.245	18.337	18.656	19.175
	WikiText	Llama2-7B	10.366	8.065	7.991	7.983	**7.959**	7.962	7.971	7.963
		Llama3-8B	82.689	14.144	12.522	12.457	12.450	12.447	**12.428**	12.570
		Mistral-7B	7.928	7.423	**7.209**	7.350	7.300	7.291	7.265	7.264
		Phi3.5-mini	19.260	**15.467**	18.537	18.586	19.189	19.512	20.058	20.350
Wanda	Pile	Llama2-7B	90.982	9.200	8.798	8.559	8.521	**8.481**	8.556	8.578
		Llama3-8B	238.360	14.934	14.375	**14.317**	14.649	14.618	14.586	14.837
		Mistral-7B	17.986	8.616	8.331	**8.304**	8.356	8.381	8.399	8.448
		Phi3.5-mini	92.533	12.623	12.152	**12.115**	12.258	12.320	12.412	12.495
	WikiText	Llama2-7B	88.706	8.819	8.377	8.222	8.162	**8.152**	8.176	8.198
		Llama3-8B	229.448	14.018	13.171	**12.985**	13.005	12.990	13.071	13.377
		Mistral-7B	16.899	8.104	7.785	7.693	**7.664**	7.679	7.673	7.668
		Phi3.5-mini	182.069	12.142	11.541	**11.480**	11.630	11.685	11.690	11.815

Table 2: Perplexity of post-training structured pruning (4:8) methods.

Method	Dataset	Model	Sequence Length							
			1	4	16	64	256	512	1024	2048
ASVD	Pile	Llama2-7B	12.086	9.622	8.924	8.878	8.692	**8.638**	8.715	8.645
		Llama3-8B	240.019	**61.525**	80.316	70.074	75.634	72.202	82.125	71.280
		Mistral-7B	17.666	**12.133**	12.346	13.458	14.152	13.382	12.770	12.471
		Phi3.5-mini	34.428	20.545	19.336	18.979	19.895	19.499	**18.212**	18.599
	WikiText	Llama2-7B	13.593	10.256	9.344	9.427	**8.949**	9.060	9.187	9.022
		Llama3-8B	379.238	**88.395**	105.591	103.276	121.171	114.070	102.240	131.133
		Mistral-7B	59.074	17.181	17.211	15.521	15.871	14.673	15.174	**14.286**
		Phi3.5-mini	46.119	19.583	18.637	18.880	19.925	19.511	**17.629**	18.484
ASVD+	Pile	Llama2-7B	11.474	13.003	12.822	9.134	7.896	**7.674**	7.785	7.878
		Llama3-8B	90.929	115.969	153.634	74.189	37.271	**27.435**	31.299	28.774
		Mistral-7B	21.636	23.358	47.590	17.565	11.476	10.922	10.395	**10.060**
		Phi3.5-mini	92.534	46.586	42.126	19.623	16.535	15.463	**15.332**	17.070
	WikiText	Llama2-7B	12.554	13.424	11.763	8.141	7.011	**6.797**	6.987	7.171
		Llama3-8B	145.772	102.865	100.822	40.614	27.756	**24.235**	25.644	28.322
		Mistral-7B	22.698	22.460	36.398	10.910	8.782	7.884	7.853	**7.726**
		Phi3.5-mini	177.627	32.863	19.115	11.245	9.323	**9.286**	9.599	9.596

Table 3: Perplexity of post-training low rank decomposition (ratio 0.8) methods.

4 Sequence Length of Calibration Data

We investigate the influence of the sequence length of calibration data on the performance of the post-training compression algorithms. Tables 1, 2, and 3 present the perplexity results for post-training quantization, pruning, and low-rank approximation methods, respectively, according to the sequence length of calibration data. For post-training pruning, we evaluate the 4:8 structured pruning method, which is regarded as hardware-friendly.

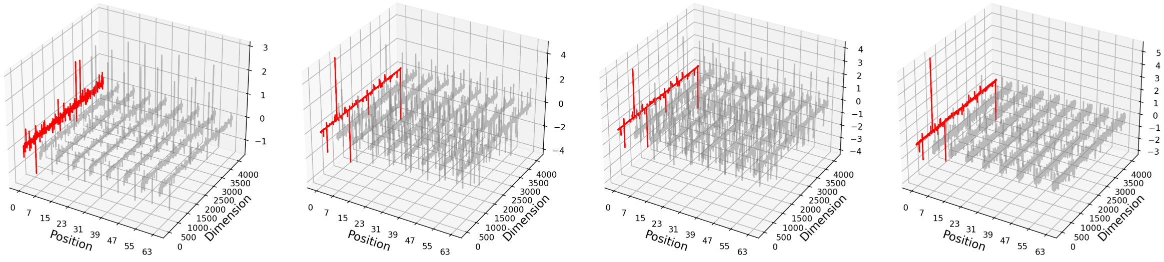
Shorter calibration data provides unexpected benefits. When focusing on calibration sequences of 4 tokens and above, the experimental results reveal that very short sequences can yield low perplexity values, which directly correlate to improved performance. For example, Table 1 demonstrates that the Phi3.5-mini calibrated on the Pile dataset reaches an optimal perplexity of 6.420 at 4 tokens, with no significant improvements observed when extending the sequence length further. This suggests that the essential activation statistics needed for effective post-training quantization are largely captured within this minimal token window. Such a finding is unexpected, as one might initially assume that a longer sequence would be necessary to fully capture the model's dynamic behavior. Furthermore, although pruning and low-rank approximation typically require a longer sequence length than quantization, our results indicate that they do not necessarily require the commonly used length of 2048 tokens.

The optimal sequence length has little correlation with the calibration dataset. Tables 1, 2, and 3 indicate that the optimal sequence length is largely independent of the calibration dataset used, whether it is Pile or WikiText. Instead, it primarily depends on the methods and models employed. Specifically, quantization generally exhibits superior perplexity when applied to shorter sequence lengths. In contrast, pruning tends to demonstrate optimal performance at mid-range sequence lengths. Meanwhile, low-rank approximation often achieves the best results when dealing with longer sequences. However, it is important to note that these observed performance trends are not necessarily dictated by the inherent characteristics of the calibration dataset. Instead, they are influenced by other factors, such as the model architecture and the specific compression methods. Thus, these findings suggest that selecting an appropriate sequence length should be guided by the specific compression technique and LLM rather than the specific calibration dataset.

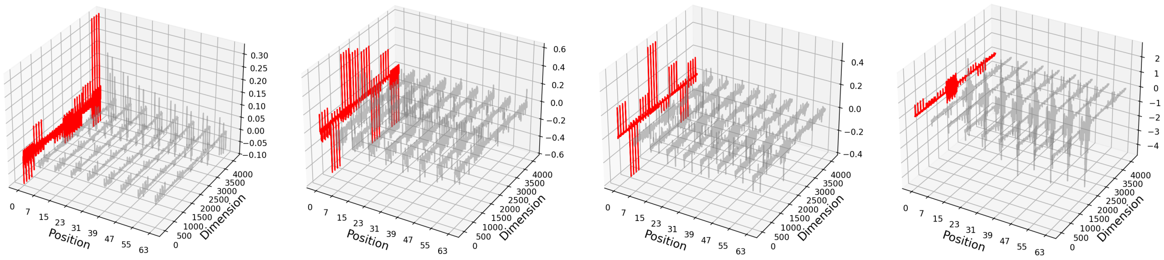
Robustness hierarchy: quantization, pruning, and low-rank approximation. The experimental results clearly establish a hierarchy in terms of robustness when different compression methods are subjected to variations in calibration sequence length. Specifically, the quantization method (Table 1) consistently achieves low perplexity values across a wide range of sequence lengths, from minimal to typical lengths. This consistent performance indicates that quantization techniques are highly resilient to changes in sequence length and remain efficient in capturing the necessary activation statistics even when provided with limited input data. In contrast, structured pruning methods (Table 2) exhibit moderate sensitivity to variations in sequence length. Their perplexity values tend to fluctuate more noticeably compared to those of the quantization method, suggesting a higher dependency on precise activation information for maintaining robust performance. Among the three categories of compression methods, low-rank approximation techniques (Table 3) demonstrate the highest level of sensitivity to sequence length variations. Their performance deteriorates more significantly when shorter sequences are used, indicating a greater reliance on an extended context to achieve optimal calibration. ASVD+ algorithm shows a marked improvement in performance as the sequence length increases, highlighting its strong dependence on a longer calibration sequence for achieving effective compression and maintaining low perplexity.

When sequence length is reduced to 1, the performance dramatically declines. Although the discussion for the previous observations mainly focuses on sequences of 4 tokens or more, it is crucial to note the dramatic performance degradation that occurs when the calibration sequence is reduced to a single token. The results from Table 1 clearly indicate that using only 1 token for the Llama2-7B model on the Pile dataset results in a perplexity of 7.897, which is substantially higher than the optimal 5.508 observed at 4 tokens. This significant increase in perplexity, reflective of poorer model performance, is also evident in the structured pruning and low-rank approximation results in Tables 2 and 3. The decline in performance can be attributed to the insufficiency of a single token to capture the activation patterns necessary for effective calibration. However, observing the significant recovery in performance at token 4 implies that the first token exhibits entirely different statistics compared to the subsequent tokens.

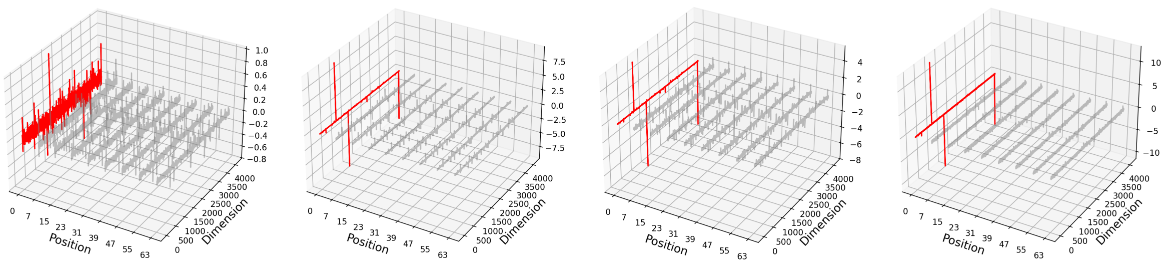
Collectively, these observations deepen our understanding of the calibration process in post-training compression. It is important to select an optimal sequence length to ensure high performance across various compression methods. The results on zero-shot downstream tasks are described in Appendix B.



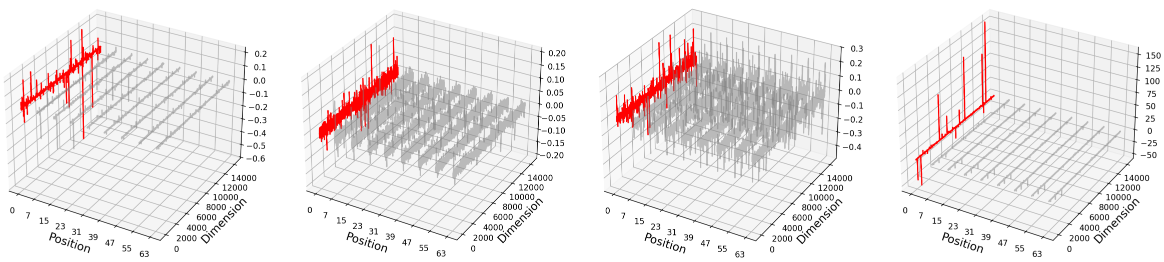
(a) Input activations of $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$ (i.e., normalized hidden state (ATTN)).



(b) Input activations of $\mathbf{W}_o$ (i.e., attention state).



(c) Input activations of $\mathbf{W}_{gate}$ and $\mathbf{W}_{up}$ (i.e., normalized hidden state (FFN)).



(d) Input activations of $\mathbf{W}_{down}$ (i.e., intermediate state).

Figure 2: Input activations of Llama3-8B, represented in increments of 8, between positions 0 and 63. Red color indicates activations at position 0. We randomly extracted 64 samples from the Pile dataset and then average them along batch axis. From left to right, figures correspond to the layer 0, 8, 17, and 31.

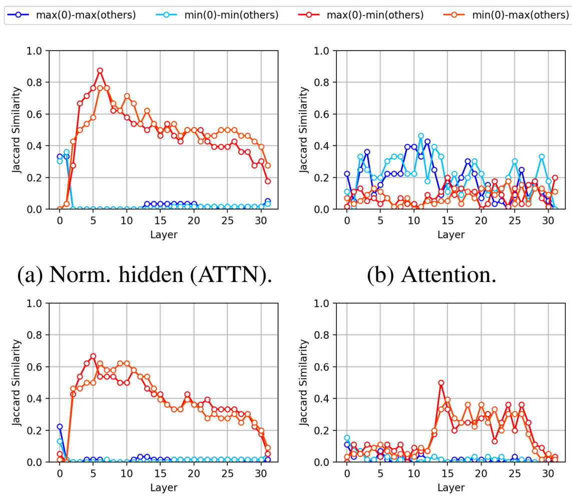
5 Analysis on Input Activation

We conduct a detailed investigation of the input activations directly, with a particular emphasis on the first token (i.e., position 0). As depicted in Figure 1, a single layer typically contains four types of input activations: normalized hidden state in the attention module (ATTN) for $\mathbf{W}_{q/k/v}$, attention state for $\mathbf{W}_o$, normalized hidden state in the feed-forward network module (FFN) $\mathbf{W}_{gate/up}$, and intermediate state for $\mathbf{W}_{down}$.

Figure 2 illustrates the input activations of Llama3-8B from position 0 to 63, with a period of 8, at four different layers: 0, 8, 17, and 31. Specifically, (a) depicts the normalized hidden state feeding into the attention projections ($\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$), (b) depicts the attention state feeding into the attention output projection ($\mathbf{W}_o$), (c) depicts the normalized hidden state feeding into the feed-forward gate and up projections ($\mathbf{W}_{gate}$ and $\mathbf{W}_{up}$), and (d) depicts the intermediate state feeding into the feed-forward down projection ($\mathbf{W}_{down}$). Each sub-figure visualizes activation values (not their absolute values) across dimensions and token positions. We randomly select 64 samples from the Pile dataset and average their activations.

Interestingly, position 0 consistently shows a distinct activation across most states and layers, even though no dedicated *bos* token is used in our setting. This implies that LLMs seem to have learned to process the first token in a distinct manner, no matter what it is. In fact, a similar phenomenon, referred to as "massive activations," has been observed (Sun et al., 2024a; Oh et al., 2024). After passing through the initial few layers, a phenomenon occurs in which only specific dimensions of the hidden state have extremely large magnitudes, and this is observed at the first position. This can also be inferred from Figures 2(a) and (c). Although we have provided the normalized hidden state, the dimensions with large magnitudes are being propagated through the residual connection. Moreover, Sun et al. (2024a) demonstrated that when massive activations are set to zero, LLMs completely fail to function. Based on this finding, we believe that the statistics of the first token should be adequately considered during calibration.

In the leftmost of Figures 2(a) and (b), it can be observed that the activations of position 0 behave similarly to those of other positions. To quantify this, we utilize the Jaccard similarity, which is a metric for measuring the similarity between two sets by dividing the size of their intersection by the size of their union. We first calculate the average of the activations for all positions except the first token. Next, we extract the indices of the top 30 maximum values and the top 30 minimum values from both the position 0 activations and the averaged activations. Figure 3 describes the Jaccard similarity between the extracted indices, where max(0) and min(0) refer to the indices of the top 30 maximum values and the top 30 minimum values from position 0, respectively. Similarly, max(others) and min(others) are defined in the same manner. A high Jaccard similarity between max(0) and max(others) (i.e., blue line) or between min(0) and min(others) (i.e., sky-blue line) means that the first token and the subsequent tokens share similar activation patterns. On the other hand, a high Jaccard similarity between max(0) and min(others) (i.e., red line) or between min(0) and max(others) (i.e., orange line) means that the first token and the subsequent tokens exhibit opposite activation patterns.



(a) Norm. hidden (ATTN). (b) Attention.

(c) Norm. hidden (FFN). (d) Intermediate.

Figure 3: Jaccard similarity between the first token and the average of the subsequent tokens across layers. The greater the red and orange lines, the more the first token and the subsequent tokens are opposite. Conversely, the greater the blue and sky-blue lines, the more the first token and the subsequent tokens are similar.

The most notable observation is that, in both the ATTN and FFN modules, a high Jaccard similarity is observed between max(0) and min(others) or between min(0) and max(others) after the layer when massive activations occur in the normalized hidden state. That is, the first token and subsequent tokens are opposite. For Llama3-8B, a similar phenomenon is observed in the intermediate state from layer 13. However, this is not a phenomenon commonly found in other LLMs, as detailed in Appendix C.

6 State-Aware Length Calibration

Building on prior observations and evidence indicating that the first token plays a pivotal role in the performance of LLMs (Sun et al., 2024a; Oh et al., 2024; Xiao et al., 2024; Hooper et al., 2024), we propose a simple calibration technique termed *state-aware length calibration*. Specifically, when the sequence length of calibration data is L, only the initial Lr tokens of the normalized hidden state in both modules are employed for calibration, where r denotes a predetermined ratio. In contrast, for other states, the full sequence length L is retained, as illustrated in Figure 1. By concentrating calibration efforts on the initial critical tokens, our method aims to capture the most salient features, thereby improving the effectiveness of the calibration process.

Method	Dataset	Model	Sequence Length=512				Sequence Length=1024			
			r=1/8	r=1/4	r=1/2	r=1	r=1/8	r=1/4	r=1/2	r=1
SmoothQuant (W8A8)	Pile	Llama2-7B	**5.514**	5.516	5.521	5.523	5.517	5.519	5.521	5.525
		Llama3-8B	6.264	**6.261**	6.262	6.266	6.262	6.262	**6.261**	6.263
		Mistral-7B	5.349	5.349	5.351	5.350	5.349	**5.348**	5.349	5.350
		Phi3.5-mini	6.459	**6.453**	6.468	6.475	6.465	6.464	6.475	6.472
AWQ (W4A16)	Pile	Llama2-7B	5.635	5.632	5.637	5.634	5.632	5.633	5.630	**5.628**
		Llama3-8B	6.846	6.847	6.845	6.847	6.835	**6.832**	6.839	6.836
		Mistral-7B	5.452	5.459	5.452	5.458	**5.451**	5.453	5.456	5.458
		Phi3.5-mini	6.751	6.758	6.738	6.749	6.611	6.586	**6.603**	6.606

Table 4: Perplexity of post-training quantization methods using our calibration data, according to the r. r represents the ratio of the sequence length used in the input activations of $\mathbf{W}_{q/k/v}$ and $\mathbf{W}_{gate/up}$. The results for r=1 correspond to the results in Table 1.

Method	Dataset	Model	Sequence Length=512				Sequence Length=1024			
			r=1/8	r=1/4	r=1/2	r=1	r=1/8	r=1/4	r=1/2	r=1
SparseGPT	Pile	Llama2-7B	8.034	8.030	8.028	8.045	8.032	**8.025**	8.039	8.041
		Llama3-8B	**13.227**	13.367	13.487	13.547	13.425	13.446	13.562	13.711
		Mistral-7B	**7.595**	7.623	7.641	7.655	7.621	7.632	7.637	7.640
		Phi3.5-mini	**17.555**	17.793	18.095	18.337	18.014	18.243	18.513	18.656
Wanda	Pile	Llama2-7B	8.574	8.496	8.492	**8.481**	8.490	8.505	8.494	8.556
		Llama3-8B	**14.297**	14.397	14.596	14.618	14.505	14.527	14.506	14.586
		Mistral-7B	**8.281**	8.307	8.359	8.381	8.309	8.361	8.388	8.399
		Phi3.5-mini	12.124	12.160	12.233	12.320	**12.117**	12.281	12.401	12.412

Table 5: Perplexity of post-training structured pruning (4:8) methods using our calibration data, according to the r. The results for r=1 correspond to the results in Table 2.

Method	Dataset	Model	Sequence Length=512				Sequence Length=1024			
			r=1/8	r=1/4	r=1/2	r=1	r=1/8	r=1/4	r=1/2	r=1
ASVD	Pile	Llama2-7B	8.729	8.691	8.648	8.638	8.628	8.570	**8.531**	8.715
		Llama3-8B	76.799	**61.172**	71.763	72.202	63.768	91.292	79.422	82.125
		Mistral-7B	**12.530**	12.607	13.307	13.382	13.130	13.188	13.122	12.770
		Phi3.5-mini	**18.117**	19.662	19.030	19.499	18.706	19.768	20.403	18.212
ASVD+	Pile	Llama2-7B	8.694	8.278	7.900	**7.674**	8.478	7.976	7.873	7.785
		Llama3-8B	52.930	43.066	37.280	**27.435**	41.889	36.771	30.548	31.299
		Mistral-7B	12.977	11.813	11.180	10.922	12.321	10.892	10.725	**10.395**
		Phi3.5-mini	20.299	17.574	**14.988**	15.463	16.966	15.806	15.152	15.332

Table 6: Perplexity of post-training low rank decomposition (ratio 0.8) methods using our calibration data, according to the r. The results for r=1 correspond to the results in Table 3.

Tables 4, 5, and 6 present the perplexity results obtained from applying state-aware length calibration to post-training quantization, pruning, and low-rank approximation methods. Our experimental findings indicate that leveraging partial sequence lengths (i.e., $r < 1$) of normalized hidden states for calibration generally results in lower perplexities across various post-training compression methods and model architectures. Although there exist certain cases where the observed performance gains are relatively modest, we would like to emphasize the significance of state-aware length calibration as a promising and effective strategy. This is because our proposed approach incurs no additional computational cost, making it a practical and efficient approach for improving the performance of compressed models.

7 Conclusion

In this study, we investigate the critical role of calibration data sequence length in improving the effectiveness of post-training compression methods for LLMs. Our analysis reveals that the normalized hidden states of the first token behave oppositely to those of subsequent tokens. Motivated by this observation, we introduce *state-aware length calibration*, a novel technique that uses a subset of initial tokens for calibration. Experimental results demonstrate that this approach generally enhances perplexity across various post-training compression methods and LLMs. Notably, as state-aware length calibration incurs no additional computational cost, we consider it a practical and simple strategy for optimizing post-training compressed LLMs.

Limitations

While our proposed state-aware length calibration technique demonstrates consistent improvements across various post-training compression methods, it currently lacks a principled strategy for selecting the optimal sequence ratio r. Future work should explore adaptive or learnable calibration length schemes to generalize our algorithm across broader settings.

Ethical Considerations

This work does not involve the collection of new datasets, human participants, or the deployment of models in sensitive applications. All experiments are conducted using publicly available models (e.g., LLaMA, Mistral) and datasets (e.g., WikiText-2, Pile), which are commonly used for academic research.

Use of AI Assistants

We used ChatGPT exclusively to improve the clarity and quality of the writing. We did not use ChatGPT for other purposes (e.g., experiment design, or analysis).

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, and 1 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. 2024. Streamlining redundant layers to compress large language models. *arXiv preprint arXiv:2403.19135*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. GPT3.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Elias Frantar and Dan Alistarh. 2022. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35:4475–4488.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, and 1 others. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Amir Gholami, Zhewei Yao, Sehoon Kim, Coleman Hooper, Michael W. Mahoney, and Kurt Keutzer. 2024. Ai and memory wall. *IEEE Micro*, 44(3):33–39.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. 2024. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *arXiv preprint arXiv:2401.18079*.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. 2022. Language model compression with weighted low-rank factorization. In *International Conference on Learning Representations*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Janghwan Lee, Minsoo Kim, Seungcheol Baek, Seok Hwang, Wonyong Sung, and Jungwook Choi. 2023. Enhancing computation efficiency in large language models through weight and activation quantization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14726–14739, Singapore. Association for Computational Linguistics.

Yixiao Li, Yifan Yu, Qingru Zhang, Chen Liang, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2023. Losparse: Structured compression of large language models based on low-rank and sparse approximation. In *International Conference on Machine Learning*, pages 20336–20350. PMLR.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024a.

Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. 2024b. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *arXiv preprint arXiv:2405.04532*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

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

Jaehoon Oh, Seungjun Shin, and Dokwan Oh. 2024. House of cards: Massive weights in llms. *arXiv preprint arXiv:2410.01866*.

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. 2024a. Massive activations in large language models. In *First Conference on Language Modeling*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024b. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, and Mi Zhang. 2024. Efficient large language models: A survey. *Transactions on Machine Learning Research*. Survey Certification.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. 2023. Bitnet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453*.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2025. SVD-LLM: Truncation-aware singular value decomposition for large language model compression. In *The Thirteenth International Conference on Learning Representations*.

Miles Williams and Nikolaos Aletras. 2024. On the impact of calibration data in post-training quantization and pruning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10100–10118.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*.

Zhihang Yuan, Yuzhang Shang, Yue Song, Dawei Yang, Qiang Wu, Yan Yan, and Guangyu Sun. 2025. ASVD: Activation-aware singular value decomposition for compressing large language models.

A Experiments Setup Details

We use eight A100 80GB GPU cards for out experiments, although we do not use multi-GPUs. For each algorithm, our state-aware calibration length technique only involves indexing, thereby adding only a negligible amount of time. It is worth noting that the most time-consuming algorithm is ASVD, with each evaluation taking around three hours.

B Zero-shot Downstream Tasks

Table 7 provides the performance of SmoothQuant (Xiao et al., 2023) and Wanda (Sun et al., 2024b) according to the sequence length of calibration dataset. The results are averaged over 5 zero-shot downstream tasks; arc easy, arc challenge, boolq, piqa, winogrande.

Method	Model	Sequence Length							
		1	4	16	64	256	512	1024	2048
SmoothQuant	Llama2-7B	63.7	69.1	69.3	**69.4**	69.3	69.2	69.2	69.1
	Llama3-8B	66.3	**73.0**	72.8	**73.0**	**73.0**	**73.0**	72.8	72.9
	Mistral-7B	73.9	**74.1**	74.0	73.6	73.7	73.6	73.8	73.8
	Phi3.5-mini	76.0	**77.0**	76.9	76.8	76.8	76.8	76.8	76.8
Wanda	Llama2-7B	44.2	60.8	61.9	62.8	63.2	**63.7**	63.5	63.5
	Llama3-8B	39.6	60.0	60.8	**61.0**	60.7	60.8	60.8	60.9
	Mistral-7B	54.8	64.3	65.3	65.6	**65.7**	65.5	65.5	65.5
	Phi3.5-mini	45.8	65.4	66.8	**67.0**	66.9	66.6	66.8	66.2

Table 7: Performance on 5 zero-shot tasks according to the sequence length of calibration dataset.

Tables 8 and 9 provide the zero-shot downstream task performance of SmoothQuant and Wanda, when our state-aware length calibration algorithm is applied, respectively.

Model	Sequence Length=512				Sequence Length=1024			
	r=1/8	r=1/4	r=1/2	r=1	r=1/8	r=1/4	r=1/2	r=1
Llama2-7B	**69.3**	69.2	69.2	69.2	69.2	**69.3**	69.2	69.2
Llama3-8B	73.0	**73.1**	**73.1**	73.0	72.9	73.0	73.0	72.8
Mistral-7B	73.6	73.6	73.6	73.6	73.7	**73.8**	73.6	**73.8**
Phi3.5-mini	**76.9**	76.8	76.7	76.8	**76.9**	76.7	76.7	76.8

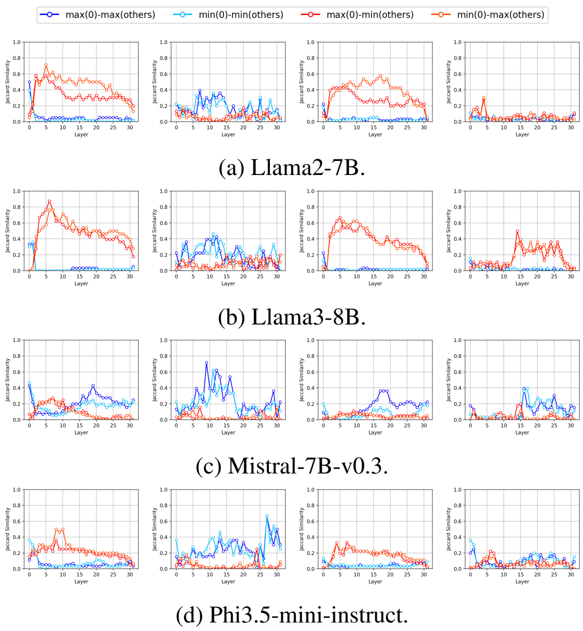
Table 8: Performance on 5 zero-shot tasks of SmoothQuant, which is the same model in Table 4.

Model	Sequence Length=512				Sequence Length=1024			
	r=1/8	r=1/4	r=1/2	r=1	r=1/8	r=1/4	r=1/2	r=1
Llama2-7B	63.0	63.4	63.6	**63.7**	63.3	63.6	63.4	63.5
Llama3-8B	60.9	60.6	60.7	60.8	**61.2**	61.1	**61.2**	60.8
Mistral-7B	65.5	65.7	**65.8**	65.5	**65.8**	65.7	**65.8**	65.5
Phi3.5-mini	66.6	**67.1**	66.6	66.6	66.6	**67.1**	66.6	66.8

Table 9: Performance on 5 zero-shot tasks of Wanda, which is the same model in Table 5.

C Jaccard Similarity of Various LLMs

Figure 4 illustrates the layer-wise Jaccard similarity of various LLMs (Llama2-7B, Llama2-8B, Mistral-7B-v0.3, and Phi3.5-mini-instruct) across four different states: (1) normalized hidden state (ATTN), (2) attention state, (3) normalized hidden state (FFN), and (4) intermediate state. Each row corresponds to a different model, and each column represents one of the four states. Across all models, a clear pattern emerges: the Jaccard similarity of the normalized hidden states in both the ATTN and FFN modules remains relatively high and stable in the lower layers but decreases as depth increases.



(a) Llama2-7B.

(b) Llama3-8B.

(c) Mistral-7B-v0.3.

(d) Phi3.5-mini-instruct.

Figure 4: Jaccard similarity of other LLMs. From left to right, figures correspond to the normalized hidden state (ATTN), attention state, normalized hidden state (FFN), and intermediate state.

D Structured Pruning

In this section, we address structured pruning methods using Llama3-8B, especially focusing on layer pruning. In the context of layer-wise pruning, analyzing the calibration length would be a valuable extension.

LLM-Streamline (Chen et al., 2024) is the most recent work among them, demonstrating strong performance. This algorithm prunes consecutive layers based on the similarity between the two hidden states and trains a lightweight network for performance recovery. In order to isolate and accurately evaluate the effect of calibration length on pruning, we deliberately omitted the subsequent training steps used for performance recovery.

Table 10 provides perplexity when some layers pruned using LLM-Streamline. When the perplexity remains unchanged, it indicates that the same layers are pruned. For instance, in the case of one-layer pruning, layer26 is pruned except for the case where sequence length is 1. For sequence length of 1, layer3 is pruned. And, for instance, in the case of eight-layers pruning, layer21-layer28 are pruned except for the case where sequence length is 1. For sequence length of 1, layer3-layer10 are pruned.

According to our findings, in the context of layer pruning, the key distinction appears to lie in whether the calibration data has a length of 1 or not. However, no strict or consistent rule was identified.

#	Sequence Length							
	1	4	16	64	256	512	1024	2048
1	7.281	6.739	6.739	6.739	6.739	6.739	6.739	6.739
2	12.508	7.503	8.389	8.389	7.503	7.503	8.389	8.389
4	37.292	15.082	15.082	15.082	15.082	15.082	15.082	15.082
8	110.330	125.956	125.956	125.956	125.956	125.956	125.956	125.956

Table 10: Perplexity according to the sequence length of calibration dataset, when some layers are pruned by LLM-Streamline. # represents the number of pruned layers.

E Model Scale

Table 11 provides the results on WikiText of Llama2-13B calibrated on Pile (similar to Tables 1, 2, and 3) when using SmoothQuant and Wanda, according to the sequence length. This shows that the key factor affecting the effectiveness of sequence length is not the model size, but rather how strongly the model's performance depends on the first token. In other words, even for large-scale models like 13B, if the model is highly sensitive to the first token, using a truncated length is likely to remain beneficial.

Method	Sequence Length							
	1	4	16	64	256	512	1024	2048
SmoothQuant	5.479	**4.922**	4.926	4.927	4.926	4.927	4.928	4.927
Wanda	44.767	7.084	6.863	**6.832**	6.876	6.885	6.922	6.960

Table 11: Perplexity of Llama2-13B according to the sequence length of calibration dataset.

F State-Aware Length Calibration

We evaluated the effectiveness of our algorithm at two sequence lengths (512 and 1024) in Tables 4, 5, and 6. We have extended this analysis to different calibration lengths (32 or 128).

Additional results also follow that using a shorter portion of the normalized hidden states ($r < 1$) yields better or comparable performance than using the full length across all states ($r = 1$) in most cases. Furthermore, the minimum perplexity values in additional tables is lower than those in Tables 1, 2, and 3.

Model	Sequence Length=32				Sequence Length=128			
	r=1/8	r=1/4	r=1/2	r=1	r=1/8	r=1/4	r=1/2	r=1
Llama2-7B	5.510	5.511	5.511	5.513	5.509	5.517	5.515	5.517
Llama3-8B	6.254	6.255	6.260	6.261	6.260	6.258	6.261	6.265
Mistral-7B	5.344	5.343	5.346	5.348	5.345	5.347	5.346	5.350
Phi3.5-mini	6.420	6.430	6.435	6.463	6.433	6.453	6.469	6.461

Table 12: Perplexity of SmoothQuant, which is the same model in Table 4.

Model	Sequence Length=32				Sequence Length=128			
	r=1/8	r=1/4	r=1/2	r=1	r=1/8	r=1/4	r=1/2	r=1
Llama2-7B	8.969	8.900	8.767	8.704	8.679	8.597	8.533	**8.479**
Llama3-8B	14.455	14.276	14.257	14.343	14.157	**14.103**	14.221	14.518
Mistral-7B	8.337	8.297	8.285	8.305	**8.277**	8.308	8.329	8.341
Phi3.5-mini	12.242	12.139	12.085	12.141	**12.041**	12.055	12.072	12.202

Table 13: Perplexity of Wanda, which is the same model in Table 5.