Mitigate Position Bias in Large Language Models via Scaling a Single Dimension

Yijiong Yu [1] Huiqiang Jiang [2] Xufang Luo [2] Qianhui Wu [2] Chin-Yew Lin [2] Dongsheng Li [2] Yuqing Yang [2]
Yongfeng Huang [1] Lili Qiu [2]

Abstract

Large Language Models (LLMs) are increasingly applied in various real-world scenarios due to their excellent generalization capabilities and robust generative abilities. However, they exhibit position bias, also known as "lost in the middle", a phenomenon that is especially pronounced in long-context scenarios, which indicates the placement of the key information in different positions of a prompt can significantly affect accuracy. This paper first explores the micro-level manifestations of position bias, concluding that attention weights are a micro-level expression of position bias. It further identifies that, in addition to position embeddings, causal attention mask also contributes to position bias by creating position-specific hidden states. Based on these insights, we propose a method to mitigate position bias by scaling this positional hidden states. Experiments on the NaturalQuestions Multi-document QA, KV retrieval, LongBench and timeline reorder tasks, using various models including RoPE models, context window-extended models, and Alibi models, demonstrate the effectiveness and generalizability of our approach. Our method can improve performance by up to 15.2% by modifying just one dimension of hidden states. Our code is available at https://aka.ms/PositionalHidden.

1. Introduction

Long-context large language models (LLMs) (Reid et al., 2024; Liu et al., 2024a; Young et al., 2024; DeepSeek-AI, 2024; Gradient, 2024; Abdin et al., 2024) have recently garnered significant attention for handling complex tasks like long-context question-answering (Caciularu et al., 2023; Li et al., 2024) and repository-level code understanding (Bairi et al., 2023). However, recent research (Liu et al., 2024b; Li et al., 2023; Shi et al., 2023; Tang et al., 2023; Li et al., 2024) indicates that these models struggle with "lost in the middle" position bias, where information in the middle of the prompt is often ignored. This issue affects nearly all LLMs (Liu et al., 2024b; Junqing et al., 2023; Zhang et al., 2024), regardless of architecture or size. For example, GPT-3.5-Turbo shows a 22-point performance drop in the NaturalQuestion QA task when ground-truth information is placed in the middle versus the ends of a 2.3k token prompt (Liu et al., 2024b). This bias worsens with longer contexts, significantly impacting LLMs' practical application (Kamradt, 2023; Zhao et al., 2024).

Previous works have analyzed this issue from the perspectives of data distribution (Junqing et al., 2023; Yu, 2023; An et al., 2024) and position embeddings (Chen et al., 2023b; Zhang et al., 2024). FILM (An et al., 2024) addresses position bias by distributing key information across various positions for supervised fine-tuning (SFT). Ms-PoE (Zhang et al., 2024) mitigates position bias by interpolating RoPE (Su et al., 2024) with head-wise scaling factors. However, these methods require additional overhead for training or online estimation of scaling coefficients and are applicable to only a few models, limiting their generalizability.

To understand and alleviate position bias in LLMs, we first explored its micro-level manifestations and observed consistent patterns in attention weights. We then investigated the underlying causes, finding that both position embeddings and the causal mask significantly affect position bias. Our analysis revealed that the causal mask introduces "positional hidden states", which are correlated with absolute positions and convey positional information to LLMs. These positional hidden appear regardless of the position encoding method used, including RoPE (Su et al., 2024), Alibi (Press et al., 2022), and even NoPE (Haviv et al., 2022).

Based on the above findings, we propose a method named **Scale Positional Hidden States** to mitigate position bias. We first design a prior-based search algorithm to quickly identify positional hidden states using monotonicity, smoothness, and validation set loss as indicators. Next, we develop an attention modification algorithm that scales these hidden states to influence the last token's attention,

[1] Tsinghua University [2] Microsoft Corporation. Correspondence to: Huiqiang Jiang <hjiang@microsoft.com>, Yijiong Yu <yuyj22@mails.tsinghua.edu.cn>.

Proceedings of the 1^{st} Workshop on Long-Context Foundation Models, Vienna, Austria. 2024.

efficiently implemented using FlashAttention (Dao, 2023).

Experiments on models like LLaMA-2 (Touvron et al., 2023), Vicuna (Chiang et al., 2023), Mistral (Jiang et al., 2023a), Gemma (Team et al., 2024), Qwen (Bai et al., 2023a), and MPT (Team, 2023), across tasks such as Multi-document QA, KV retrieval, LongBench (Bai et al., 2023b), and timeline reorder (Li et al., 2023), show our method mitigates position bias by modifying one dimension of hidden states, achieving up to 15.2% improvement. Our method is compatible with various position embeddings, including RoPE (Su et al., 2024) and Alibi (Press et al., 2022), demonstrating good generalization.

2. Beyond Position Embeddings: Causal Masks Also Contribute to Position Bias in LLMs

This section identifies patterns in attention weights that closely correspond to position bias. Additionally, we discover that, except position embeddings, position information in LLMs can also be generated by the causal mask. This information tends to accumulate in specific hidden state channels, significantly contributing to the emergence of position bias.

2.1. Microscopic Manifestations of Position Bias in Transformers: Attention Weight Patterns

The attention of auto-regressive can be represented by the following equations:

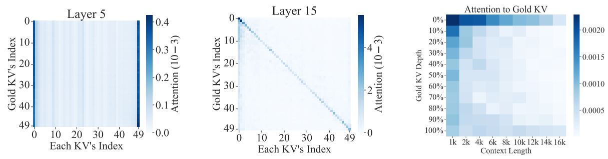
$$\boldsymbol{q} = \mathcal{P}(W^Q \boldsymbol{h}(n), n), \quad \boldsymbol{k} = \mathcal{P}(W^K \boldsymbol{h}(m), m)$$
$$\boldsymbol{a}_{n,m} = \text{Softmax}(\frac{\boldsymbol{q}\boldsymbol{k}^{\mathrm{T}} + \text{Mask}}{\sqrt{d}}) \tag{1}$$

where $\boldsymbol{h}(n)$ is the hidden state of the n-th token. W^Q, W^K are the weights of the linear layers, $\mathcal{P}$ is the position encoding function like RoPE (Su et al., 2024), and n and m are the positional order information.

To explore the micro-level manifestations of position bias in Transformers, we analyzed the attention weights for sentences containing key information in different positions. As shown in Figures 1, in deep layers, the model exhibits retrieval-like behavior by focusing on ground-truth information, forming a diagonal pattern observed in Figure 1b. In contrast, shallow layers focus most attention on the start or end of the prompt, wherever the key information is located, exhibiting vertical line patterns, as shown in Figure 1a.

In these layers exhibiting retrieval-like behavior, the attention weights for key information (Gold KV) show patterns similar to position bias: when key information is located at the start or end of the prompt, the attention weights are relatively higher, while in the middle, they are significantly lower. Additionally, as shown in Figure 1c, we observe the attention to key information with different context lengths. As the context length increases, the attenuation of attention weights with respect to position becomes more pronounced, reaching almost zero in the middle. More details about this can be found in Appendix I and C.

Furthermore, in Appendix E, we found that artificially adjusting the attention weights to the key information can directly improve the corresponding accuracy. Thus, we claim that position bias is largely caused by the attention weight patterns at the micro level.



(a) Vertical Line (b) Diagonal Line (c) Across Context

Figure 1. Attention distribution of the ground-truth KV pair to each KV pair across different positions on the KV retrieval task (Liu et al., 2024b) using Mistral-7B (Jiang et al., 2023a). (a) and (b) show the results averaged across all heads of the layer. (c) shows the attention of the ground-truth KV to the ground-truth KV (i.e., diagonal lines from (b)) across different context lengths.

2.2. Causal Mask Also Contributes to Position Bias

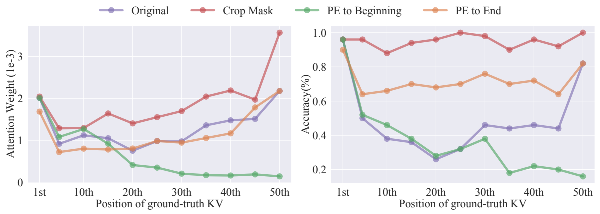


Figure 2. Performance of different methods with the ground-truth KV at different positions in the KV retrieval task (Liu et al., 2024b) using Mistral-7B (Jiang et al., 2023a).

Based on Eq. (1), position embedding $\mathcal{P}$ allows LLMs to acquire positional information. However, recent works (Haviv et al., 2022; Wang et al., 2024; Chi et al., 2023) indicate that, besides position embeddings, the causal mask can also introduce positional information. We determine whether these two factors affect position bias by modifying different properties of the ground-truth KV pair, by introduce three baselines: (1) Crop Mask, which modifies the causal mask so the ground-truth KV pair only sees itself but not previous tokens; (2) and (3) PE to Beginning/End, which sets the position IDs of the ground-truth KV pair to match the first or last KV pair. More details are provided in Appendix F.

As shown in Figure 2, the original results exhibit a "lost in the middle" pattern in both accuracy and attention weight. While PE to End provides some improvement, it does not match the accuracy when the ground-truth KV pair is at the start or end of the prompt. PE to Beginning results in a noticeable performance drop and reduced attention weight when the gold KV is near the end. In contrast, modifying the causal mask effectively enhances attention, especially to the latter KVs, and improves performance in the middle to almost match that at the beginning. Based on these observations, we conclude that besides position embedding, the causal mask is also a significant factor affecting position bias and attention weights. Solely modifying the position embedding does not completely alleviate position bias.

2.3. Casual Mask Stores Position Information in Specific Hidden states Channels

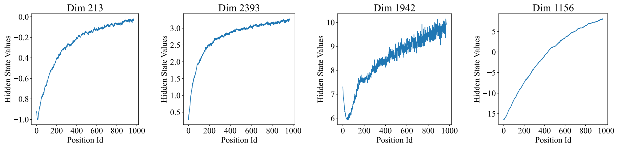
Definition 2.1 (Positional Hidden States)**.** Let $h_k(p)$ denote the k-th dimension of the hidden states across each token's position p. We define positional hidden states h_t as hidden states whose values vary consistently and monotonically with the position sequence. Therefore, their derivative (after curve fitting) should always be positive or negative:

- $h'_t(p) > 0, \forall p$ or $h'_t(p) < 0, \forall p$

To analyze positional information transmission in transformers, we define positional hidden states, which reflect absolute positional information with high correlations to position IDs (Definition 2.1). We employ monotonicity rather than correlation as the primary property of positional hidden states, as correlation does not account for the sequential nature of positions. As shown in Figure 3, our experiments reveal that causal LLMs consistently possess such hidden states across most layers (details in Appendix J), even though these models do not have explicit absolute position embeddings. This suggests that the causal mask is a likely factor providing absolute positional information. We demonstrate its major influence on positional hidden states through perturbation experiments on the causal mask and position embedding in Appendix F. Combining this with the conclusion from Section 2.2, we conclude that the causal mask stores positional information in some special hidden states and further affects attention weights, leading to position bias.

3. Methodology

Based on the findings in Section 2, although the causal mask profoundly influences position bias, it is difficult to know the positions of effective information in the prompt and modify the causal mask. Therefore, we propose a method to mitigate position bias by scaling the positional hidden states, as shown in Figure 4. Specifically, this method involves two steps: identifying the positional hidden states h_t and scaling them by the factor s.



(a) Mistral-7b (b) LLaMA2-7b (c) MPT-30b (d) NoPE-1.1B

Figure 3. Averaged positional hidden states across all layers in different models, which use TinyLlama-NoPE-1.1B in NoPE.

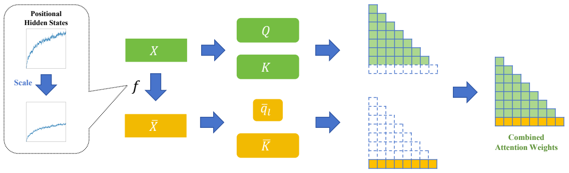


Figure 4. The framework of scaling positional hidden states and modifying attention.

3.1. Identifying Positional Hidden States

We have defined positional hidden states in Definition 2.1. However, the original values of hidden states may not strictly satisfy monotonicity. After curve fitting, we can identify dozens or hundreds of dimensions that exhibit various degrees of relevance to positional information. Thus, the first step of our method is to find the dimension that best fits the properties of positional hidden states.

To efficiently search for the positional hidden states from the LLMs' hidden states set, we propose a prior-based positional hidden search algorithm, as shown in Algorithm 1, including the following two steps: 1) Identify the top-k dimensions ρ in the hidden states that are monotonic in more than ε layers and are as smooth as possible. Here, c_t is the number of layers where $h_t(p)$ is monotonic, and g_t is the smooth score of $h_t(p)$ as in Equ.(2). 2) Use a small validation dataset $\mathcal{D}$val $= \boldsymbol{x}, \boldsymbol{y}$ to evaluate the impact of scaling these positional hidden states, respectively, and select the positional hidden state $h\bar{t}$ that leads to the minimal loss $\mathcal{L}_{\bar{t}}$.

$$\text{Smooth}(h_t) = \int |h''_t(p)|^2 \tag{2}$$

3.2. Scaling the Positional Hidden States

To minimize the impact of this modification on the semantics of LLMs, we propose scaling the positional hidden states to only affect the last token, as shown in Figure 4. Specifically, for the tokens preceding the last token, the attention remains the same as the original. For the last token, we obtain the modified query state $\overline{\boldsymbol{q}}_l$ and key states $\overline{\boldsymbol{K}}$ by scaling the

Table 1. Performance of different methods with different models on NaturalQuestions (20 docs) (Liu et al., 2024b) and KV retrieval (140 KV pairs) (Liu et al., 2024b) dataset.

Methods	NaturalQuestion						KV Retrieval					
	1st	5th	10th	15th	20th	Avg.	0%	25%	50%	75%	100%	Avg.
LLaMA-2-7b-chat	32.4	23.8	30.6	31.6	38.2	31.3	77.6	24.6	62.0	35.6	78.0	55.6
LLaMA-2-7b-chat w/ Ms-PoE	**40.8**	29.2	33.0	32.8	39.6	35.1	**95.0**	29.8	21.4	**51.8**	89.8	57.6
LLaMA-2-7b-chat w/ Ours	33.6	**34.0**	**40.6**	**43.0**	**51.8**	**40.6**	63.6	**38.0**	**82.2**	40.6	**94.6**	**63.8**
LLaMA-2-13b-chat	45.2	39.6	40.4	44.2	51.0	44.1	74.2	**39.0**	**70.4**	**84.4**	**86.8**	**71.0**
LLaMA-2-13b-chat w/ Ms-PoE	48.4	41.4	42.4	45.4	52.6	46.0	**87.8**	28.0	35.4	49.2	83.0	56.7
LLaMA-2-13b-chat w/ Ours	**50.6**	**43.4**	**45.0**	**49.4**	**58.2**	**49.3**	41.2	17.0	49.6	76.8	84.8	53.9
Vicuna-7b-v1.5-16k	**70.4**	54.8	46.8	45.8	47.8	53.1	**98.4**	0.8	0.2	0.2	0.2	20.0
Vicuna-7b-v1.5-16k w/ Ms-PoE	67.0	55.2	50.6	46.8	48.2	53.6	97.4	**36.8**	**15.6**	**5.2**	6.6	**32.3**
Vicuna-7b-v1.5-16k w/ Ours	63.8	**57.6**	**53.6**	**51.2**	**55.6**	**56.4**	95.4	22.0	12.6	**5.2**	**20.4**	31.1
Vicuna-13b-v1.5-16k	67.4	48.2	45.2	45.6	44.4	50.2	95.6	74.2	64.2	58.8	18.2	62.2
Vicuna-13b-v1.5-16k w/ Ms-PoE	**70.0**	**51.4**	46.8	42.8	47.0	51.6	91.8	59.4	71.6	**74.4**	**48.8**	69.2
Vicuna-13b-v1.5-16k w/ Ours	67.4	**51.4**	**47.6**	**48.8**	**48.0**	**52.7**	**97.2**	**83.4**	**80.8**	68.8	35.4	**73.1**
Mistral-7b-Instruct-v0.2	57.2	55.0	61.2	**61.6**	62.6	59.5	**99.8**	93.0	89.0	95.0	94.2	94.2
Mistral-7b-Instruct-v0.2 w/ Ms-PoE	58.2	**60.0**	62.6	58.8	62.2	60.4	**99.8**	**95.6**	88.4	**96.0**	**95.4**	**95.0**
Mistral-7b-Instruct-v0.2 w/ Ours	**61.2**	56.4	**63.2**	59.8	**64.0**	**60.9**	97.6	93.2	**90.6**	95.6	93.8	94.2
Gemma-1.1-7b-it	29.6	25.2	28.2	29.6	27.4	28.0	**98.6**	67.0	62.4	83.4	**100.0**	82.3
Gemma-1.1-7b-it w/ Ms-PoE	33.8	29.0	31.6	28.6	28.6	30.3	0.0	0.0	0.0	0.0	0.0	0.0
Gemma-1.1-7b-it w/ Ours	**35.4**	**31.4**	**36.0**	**35.4**	**35.0**	**34.6**	97.6	**95.8**	**97.6**	**96.8**	99.6	**97.5**
Qwen1.5-7b-chat	**72.4**	53.8	52.2	51.2	54.4	56.8	**100.0**	**97.2**	84.6	60.0	56.4	79.6
Qwen1.5-7b-chat w/ Ms-PoE	67.4	49.8	48.2	47.4	47.0	52.0	3.4	1.4	2.8	2.6	0.6	2.2
Qwen1.5-7b-chat w/ Ours	67.4	**55.2**	**53.6**	**56.0**	**59.4**	**58.3**	97.2	95.6	**98.8**	**76.6**	**94.4**	**92.5**
MPT-30b-chat	**75.6**	**49.6**	39.0	33.4	39.6	47.4	71.4	34.8	31.6	41.6	74.0	50.7
MPT-30b-chat w/ Ms-PoE	/	/	/	/	/	/	/	/	/	/	/	/
MPT-30b-chat w/ Ours	75.0	48.8	**41.6**	**40.6**	**44.0**	**50.0**	**99.0**	**65.8**	**48.6**	**46.6**	**69.4**	**65.9**

Table 2. Performance of different methods with different models on LongBench (Bai et al., 2023b).

Models	SingleDoc	MultiDoc	Synth.	Summ.	FewShot	Code	AVG
LLaMA-2-7b-chat	28.9	29.7	6.6	26.3	10.2	12.2	19.0
LLaMA-2-7b-chat w/ Ms-PoE	**29.8**	**31.7**	**10.5**	**26.7**	6.4	13.2	19.7
LLaMA-2-7b-chat w/ Ours	29.2	29.3	9.7	25.0	**18.9**	**20.8**	**22.1**
LLaMA-2-13b-chat	21.4	14.6	11.2	26.1	4.7	16.9	15.8
LLaMA-2-13b-chat w/ Ms-PoE	20.8	**15.4**	**12.7**	**27.3**	3.1	15.7	15.8
LLaMA-2-13b-chat w/ Ours	**30.6**	9.6	10.8	25.7	**27.6**	**18.7**	**20.5**
Vicuna-7b-v1.5-16k	30.2	21.6	7.2	26.7	9.4	21.2	19.4
Vicuna-7b-v1.5-16k w/ Ms-PoE	**32.3**	**24.2**	8.3	**28.0**	9.8	22.2	20.8
Vicuna-7b-v1.5-16k w/ Ours	27.1	22.1	**11.2**	26.1	**16.7**	**20.2**	**20.6**
Vicuna-13b-v1.5-16k	31.1	33.8	21.2	26.2	21.6	23.8	26.3
Vicuna-13b-v1.5-16k w/ Ms-PoE	**34.5**	33.1	16.0	**27.5**	21.0	**25.0**	26.2
Vicuna-13b-v1.5-16k w/ Ours	30.1	**35.1**	**25.0**	25.8	**27.0**	24.7	**27.9**
Mistral-7b-Instruct-v0.2	37.8	28.5	49.7	28.8	49.9	44.0	39.8
Mistral-7b-Instruct-v0.2 w/ Ms-PoE	**41.7**	22.2	38.4	24.9	14.0	19.5	26.8
Mistral-7b-Instruct-v0.2 w/ Ours	38.4	**30.4**	**49.8**	**29.4**	**51.4**	**45.3**	**40.8**
Gemma-1.1-7b-it	39.4	**23.2**	32.2	24.2	14.4	**19.8**	25.5
Gemma-1.1-7b-it w/ Ms-PoE	**41.7**	22.2	**38.4**	**24.9**	14.0	19.5	**26.8**
Gemma-1.1-7b-it w/ Ours	39.0	23.0	35.5	24.5	**14.9**	19.3	25.7
Qwen1.5-7b-chat	**46.4**	39.5	38.4	22.3	39.9	44.6	38.5
Qwen1.5-7b-chat w/ Ms-PoE	42.0	**41.5**	30.3	**25.7**	**43.2**	41.4	37.4
Qwen1.5-7b-chat w/ Ours	45.8	38.8	**38.5**	22.1	40.0	**48.1**	**38.9**
MPT-30b-chat	27.9	**21.9**	**7.5**	25.7	18.8	16.7	19.7
MPT-30b-chat w/ Ms-PoE	/	/	/	/	/	/	/
MPT-30b-chat w/ Ours	**29.4**	19.5	6.7	**25.8**	**23.0**	**21.2**	**20.9**

positional hidden states. That is,

$$\begin{aligned} \overline{\boldsymbol{q}}_l &= \mathcal{P}(W^Q f(\boldsymbol{h}(l), p, s), l), \\ \overline{\boldsymbol{K}} &= \mathcal{P}(W^K f(\boldsymbol{h}, p, s), [1, 2, ..., l]) \end{aligned} \tag{3}$$

Here, $f(\boldsymbol{h}, p, s)$ means the p-th dimension of $\boldsymbol{h}$ is scaled by the factor s. Therefore, the corresponding attention calculation is as follows:

$$\boldsymbol{z} = \begin{cases} \text{Softmax}(\dfrac{\boldsymbol{q}_i \boldsymbol{K}^\top + \text{Mask}}{\sqrt{d}})\boldsymbol{V}, & i < l \\ \text{Softmax}(\dfrac{\overline{\boldsymbol{q}}_l \overline{\boldsymbol{K}}^\top}{\sqrt{d}})\boldsymbol{V}, & i = l \end{cases} \tag{4}$$

where $\boldsymbol{z}$ is the attention output. We use FlashAttention (Dao, 2023) to implement our method with minimal overhead. After obtaining the combined attention weights, the remaining computations are the same as the original.

4. Experiments

Tables 1, 2, and 6 present the performance of various methods across different benchmarks. Several observations and conclusions can be drawn: 1) Our method consistently improves overall performance at different positions, with increases of up to 9.3%, 15.2%, and 4.7% in NQ, KV retrieval, and LongBench, respectively, except for LLaMA-2-13B in KV retrieval. Compared to the SoTA method Ms-PoE, our method shows significant improvements of up to 6.3%, 97.5%, and 14% in NQ, KV retrieval, and LongBench, respectively. The poor performance of Ms-PoE in KV retrieval is attributed to interpolation causing information loss. 2) Our method effectively enhances LLMs' understanding of information located in the middle and latter parts of the prompt. For key information at the beginning of the prompt, performance is comparable to baselines. Considering only the average performance of the last four positions, our method's improvements over the original increase to 11.3% and 16.8% in NQ and KV retrieval, respectively, and over Ms-PoE increase to 8.7% and 97.5% in NQ and KV retrieval, respectively. 3) Our approach is effective not only for RoPE models but also for context window extended models like Vicuna-16K, which already readjust RoPE (Chen et al., 2023a). Additionally, our method can be adapted to different position embeddings, such as Alibi (Press et al., 2022) models like MPT, resulting in improvements of 2.6%, 15.2%, and 1.2% in NQ, KV retrieval, and LongBench, respectively. 4) Our method demonstrated improvement across different tasks, with the most significant increases being 22.9% in few-shot learning tasks, 8.6% in code tasks, 4% in synthetic tasks, 9.2% in single document QA tasks, and 1.9% in multi-document QA tasks. In summarization tasks, performance was nearly on par with the original results. 5) Our method does not disrupt the necessary position information in LLMs, as detailed in Appendix H.

5. Conclusion

This paper proposes a method to mitigate position bias in LLMs by scaling positional hidden states. The study confirms that attention weights manifest position bias within transformers and shows that the causal mask also contributes to this bias, transmitted through hidden states containing absolute positional information, termed positional hidden states. We introduce a prior-based search algorithm to identify and scale these states, reducing position bias. Testing eight open-source models with different position embeddings on tasks such as NaturalQuestions Multi-document QA, KV Retrieval, and LongBench, our method effectively reduces position bias and improves model performance.

References

Abdin, M., Jacobs, S. A., Awan, A. A., Aneja, J., Awadallah, A., Awadalla, H., Bach, N., Bahree, A., Bakhtiari, A., Behl, H., Benhaim, A., Bilenko, M., Bjorck, J., Bubeck, S., Cai, M., Mendes, C. C. T., Chen, W., Chaudhary, V., Chopra, P., Giorno, A. D., de Rosa, G., Dixon, M., Eldan, R., Iter, D., Garg, A., Goswami, A., Gunasekar, S., Haider, E., Hao, J., Hewett, R. J., Huynh, J., Javaheripi, M., Jin, X., Kauffmann, P., Karampatziakis, N., Kim, D., Khademi, M., Kurilenko, L., Lee, J. R., Lee, Y. T., Li, Y., Liang, C., Liu, W., Lin, E., Lin, Z., Madan, P., Mitra, A., Modi, H., Nguyen, A., Norick, B., Patra, B., Perez-Becker, D., Portet, T., Pryzant, R., Qin, H., Radmilac, M., Rosset, C., Roy, S., Ruwase, O., Saarikivi, O., Saied, A., Salim, A., Santacroce, M., Shah, S., Shang, N., Sharma, H., Song, X., Tanaka, M., Wang, X., Ward, R., Wang, G., Witte, P., Wyatt, M., Xu, C., Xu, J., Yadav, S., Yang, F., Yang, Z., Yu, D., Zhang, C., Zhang, C., Zhang, J., Zhang, L. L., Zhang, Y., Zhang, Y., Zhang, Y., and Zhou, X. Phi-3 technical report: A highly capable language model locally on your phone, 2024.

An, S., Ma, Z., Lin, Z., Zheng, N., and Lou, J.-G. Make Your LLM Fully Utilize the Context, 2024. URL https://arxiv.org/abs/2404.16811.

Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., Hui, B., Ji, L., Li, M., Lin, J., Lin, R., Liu, D., Liu, G., Lu, C., Lu, K., Ma, J., Men, R., Ren, X., Ren, X., Tan, C., Tan, S., Tu, J., Wang, P., Wang, S., Wang, W., Wu, S., Xu, B., Xu, J., Yang, A., Yang, H., Yang, J., Yang, S., Yao, Y., Yu, B., Yuan, H., Yuan, Z., Zhang, J., Zhang, X., Zhang, Y., Zhang, Z., Zhou, C., Zhou, J., Zhou, X., and Zhu, T. Qwen technical report. *ArXiv preprint*, abs/2309.16609, 2023a. URL https://arxiv.org/abs/2309.16609.

Bai, Y., Lv, X., Zhang, J., Lyu, H., Tang, J., Huang, Z., Du, Z., Liu, X., Zeng, A., Hou, L., Dong, Y., Tang, J., and Li, J. Longbench: A bilingual, multitask benchmark for long context understanding. *ArXiv preprint*, abs/2308.14508, 2023b. URL https://arxiv.org/abs/2308.14508.

Bairi, R., Sonwane, A., Kanade, A., Vageesh, D., Iyer, A., Parthasarathy, S., Rajamani, S., Ashok, B., and Shet, S. Codeplan: Repository-level coding using llms and planning. 2023.

Bertsch, A., Alon, U., Neubig, G., and Gormley, M. R. Unlimiformer: Long-range transformers with unlimited length input. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=lJWUJWLCJo.

Caciularu, A., Peters, M. E., Goldberger, J., Dagan, I., and Cohan, A. Peek across: Improving multi-document modeling via cross-document question-answering. pp. 1970–1989, 2023.

Chen, S., Wong, S., Chen, L., and Tian, Y. Extending Context Window of Large Language Models via Positional Interpolation, 2023a. URL https://arxiv.org/abs/2306.15595.

Chen, Y., Lv, A., Lin, T.-E., Chen, C., Wu, Y., Huang, F., Li, Y., and Yan, R. Fortify the Shortest Stave in Attention: Enhancing Context Awareness of Large Language Models for Effective Tool Use, 2023b. URL https://arxiv.org/abs/2312.04455.

Chi, T.-C., Fan, T.-H., Chen, L.-W., Rudnicky, A., and Ramadge, P. Latent positional information is in the self-attention variance of transformer language models without positional embeddings. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1183–1193, 2023.

Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., Stoica, I., and Xing, E. P. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Dao, T. Flashattention-2: Faster attention with better parallelism and work partitioning. 2023.

DeepSeek-AI. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024.

Ding, Y., Zhang, L. L., Zhang, C., Xu, Y., Shang, N., Xu, J., Yang, F., and Yang, M. LongRoPE: Extending LLM Context Window Beyond 2 Million Tokens, 2024. URL https://arxiv.org/abs/2402.13753.

Fu, Y., Panda, R., Niu, X., Yue, X., Hajishirzi, H., Kim, Y., and Peng, H. Data engineering for scaling language models to 128k context. *ArXiv preprint*, abs/2402.10171, 2024. URL https://arxiv.org/abs/2402.10171.

Gao, Y., Wang, L., Fang, J., Hu, L., and Cheng, J. Empower Your Model with Longer and Better Context Comprehension, 2023. URL https://arxiv.org/abs/2307.13365.

Gradient. Llama-3 8b instruct gradient 4194k (v0.1), 2024.

Haviv, A., Ram, O., Press, O., Izsak, P., and Levy, O. Transformer language models without positional encodings still learn positional information. In *Findings of* *the Association for Computational Linguistics: EMNLP 2022*, pp. 1382–1390, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-emnlp.99.

He, Z., Jiang, H., Wang, Z., Yang, Y., Qiu, L., and Qiu, L. Position engineering: Boosting large language models through positional information manipulation. *arXiv preprint arXiv:2404.11216*, 2024.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *ArXiv preprint*, abs/2310.06825, 2023a. URL https://arxiv.org/abs/2310.06825.

Jiang, H., Wu, Q., Luo, X., Li, D., Lin, C.-Y., Yang, Y., and Qiu, L. LongLLMLingua: Accelerating and Enhancing LLMs in Long Context Scenarios via Prompt Compression, 2023b. URL https://arxiv.org/abs/2310.06839.

Junqing, H., Kunhao, P., Xiaoqun, D., Zhuoyang, S., Yibo, L., Yuxin, L., Hao, W., Qianguo, S., Songxin, Z., Zejian, X., et al. Never lost in the middle: Improving large language models via attention strengthening question answering. *ArXiv preprint*, abs/2311.09198, 2023. URL https://arxiv.org/abs/2311.09198.

Kamradt, G. Needle in a haystack - pressure testing llms, 2023.

Li, J., Wang, M., Zheng, Z., and Zhang, M. LooGLE: Can Long-Context Language Models Understand Long Contexts?, 2023. URL https://arxiv.org/abs/2311.04939.

Li, T., Zhang, G., Do, Q. D., Yue, X., and Chen, W. Long-context LLMs Struggle with Long In-context Learning, 2024. URL https://arxiv.org/abs/2404.02060.

Liu, H., Zaharia, M., and Abbeel, P. Ring attention with blockwise transformers for near-infinite context. 2023.

Liu, H., Yan, W., Zaharia, M., and Abbeel, P. World model on million-length video and language with ringattention. *ArXiv preprint*, abs/2402.08268, 2024a. URL https://arxiv.org/abs/2402.08268.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024b.

Nijkamp, E., Xie, T., Hayashi, H., Pang, B., Xia, C., Xing, C., Vig, J., Yavuz, S., Laban, P., Krause, B., Purushwalkam, S., Niu, T., Kryściński, W., Murakhovs'ka, L., Choubey, P. K., Fabbri, A., Liu, Y., Meng, R., Tu, L., Bhat, M., Wu, C.-S., Savarese, S., Zhou, Y., Joty, S., and Xiong, C. Xgen-7b technical report. *ArXiv preprint*, abs/2309.03450, 2023. URL https://arxiv.org/abs/2309.03450.

Peng, B., Quesnelle, J., Fan, H., and Shippole, E. Yarn: Efficient context window extension of large language models, 2023.

Peysakhovich, A. and Lerer, A. Attention Sorting Combats Recency Bias In Long Context Language Models, 2023. URL https://arxiv.org/abs/2310.01427.

Press, O., Smith, N. A., and Lewis, M. Train short, test long: Attention with linear biases enables input length extrapolation. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=R8sQPpGCv0.

Reid, M., Savinov, N., Teplyashin, D., Lepikhin, D., Lillicrap, T., Alayrac, J.-b., Soricut, R., Lazaridou, A., Firat, O., Schrittwieser, J., et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *ArXiv preprint*, abs/2403.05530, 2024. URL https://arxiv.org/abs/2403.05530.

Shi, F., Chen, X., Misra, K., Scales, N., Dohan, D., Chi, E., Schärli, N., and Zhou, D. Large language models can be easily distracted by irrelevant context, 2023.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding, 2024.

Sun, M., Chen, X., Kolter, J. Z., and Liu, Z. Massive Activations in Large Language Models, 2024. URL https://arxiv.org/abs/2402.17762.

Tang, R., Kong, D., Huang, L., and Xue, H. Large language models can be lazy learners: Analyze shortcuts in in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 4645–4657, 2023.

Team, G., Mesnard, T., Hardin, C., Dadashi, R., Bhupatiraju, S., Pathak, S., Sifre, L., Rivière, M., Kale, M. S., Love, J., et al. Gemma: Open models based on gemini research and technology. *ArXiv preprint*, abs/2403.08295, 2024. URL https://arxiv.org/abs/2403.08295.

Team, M. N. Introducing mpt-30b: Raising the bar for open-source foundation models, 2023. URL www.mosaicml.com/blog/mpt-30b. Accessed: 2023-06-22.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *ArXiv preprint*, abs/2307.09288, 2023. URL https://arxiv.org/abs/2307.09288.

Tworkowski, S., Staniszewski, K., Pacek, M., Wu, Y., Michalewski, H., and Miłoś, P. Focused transformer: Contrastive training for context scaling. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=s1FjXzJ0jy.

Wang, J., Ji, T., Wu, Y., Yan, H., Gui, T., Zhang, Q., Huang, X., and Wang, X. Length Generalization of Causal Transformers without Position Encoding, 2024. URL https://arxiv.org/abs/2404.12224.

Wu, W., Wang, Y., Xiao, G., Peng, H., and Fu, Y. Retrieval Head Mechanistically Explains Long-Context Factuality, 2024. URL https://arxiv.org/abs/2404.15574.

Yin, Q., He, X., Zhuang, X., Zhao, Y., Yao, J., Shen, X., and Zhang, Q. StableMask: Refining Causal Masking in Decoder-only Transformer, 2024. URL https://arxiv.org/abs/2402.04779.

Young, A., Chen, B., Li, C., Huang, C., Zhang, G., Zhang, G., Li, H., Zhu, J., Chen, J., Chang, J., et al. Yi: Open foundation models by 01. ai. *ArXiv preprint*, abs/2403.04652, 2024. URL https://arxiv.org/abs/2403.04652.

Yu, Y. Training With "Paraphrasing the Original Text” Improves Long-Context Performance, 2023. URL https://arxiv.org/abs/2312.11193.

Zhang, Z., Chen, R., Liu, S., Yao, Z., Ruwase, O., Chen, B., Wu, X., and Wang, Z. Found in the Middle: How Language Models Use Long Contexts Better via Plug-and-Play Positional Encoding, 2024. URL https://arxiv.org/abs/2403.04797.

Zhao, J., Zu, C., Xu, H., Lu, Y., He, W., Ding, Y., Gui, T., Zhang, Q., and Huang, X. Longagent: Scaling language models to 128k context through multi-agent collaboration. *ArXiv preprint*, abs/2402.11550, 2024. URL https://arxiv.org/abs/2402.11550.

A. Additional Methodology Details

A.1. Problem Formulation

Given a pre-trained LLM $\boldsymbol{\theta}$ and a general dataset $\{\boldsymbol{x}, \boldsymbol{y}\}$, our objective is to find the optimal positional hidden states h_t and the corresponding scaling factor s to maximally reduce position bias, which can be formulated as follows:

$$\underset{h_t \in \mathcal{H}, s<1}{\arg\min}\ \mathbb{E}\left[\sum_{i=1}^{|\boldsymbol{P}|} \mathcal{L}\left(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{p}_i; F(\boldsymbol{\theta}, h_t, s)\right)\right] \tag{5}$$

where $\boldsymbol{P}$ represents the set of different positions of the ground-truth information within the prompt $\boldsymbol{x}$, $F(\boldsymbol{\theta}, h_t, s)$ denotes the operation of scaling the LLM $\boldsymbol{\theta}$ on the t-th dimension of its hidden states by the scaling factor s, and $\mathcal{L}$ denotes the loss for general downstream tasks of the modified model.

A.2. Identifying Positional Hidden States Algorithm

Algorithm 1 Positional Hidden State Search

```
Input: LLM $\boldsymbol{\theta}$, hidden states $\mathcal{H}$, layer number $L$, validation set $\mathcal{D}_{\text{val}}$, positions set $\boldsymbol{P}$, threshold $\varepsilon$
# Indentify top-K positional dimensions
$\boldsymbol{\rho} \leftarrow \phi$
for $t \leftarrow 1$ to $|\mathcal{H}|$ do
    $c_t \leftarrow 0, g_t \leftarrow 0$
    for $l \leftarrow 1$ to $L$ do
        if $h'_t(p) > 0, \forall\, p$ or $h'_t(p) < 0, \forall\, p$ then
            $c_t \leftarrow c_t + 1, g_t \leftarrow g_t + \text{Smooth}(h_t^l)$
        end if
    end for
    if $c_t > \varepsilon$ then
        $\boldsymbol{\rho} \leftarrow \boldsymbol{\rho} \cup \{t\}$
    end if
end for
$\boldsymbol{\rho} \leftarrow \underset{t \in \boldsymbol{\rho}}{\arg\min}_K\, g_t$
# Evaluate on the validation dataset
for $t \in \boldsymbol{\rho}$ do
    $\mathcal{L}_t \leftarrow 0$
    for $p \in \boldsymbol{P}$ do
        $\mathcal{L}_t \leftarrow \mathcal{L}_t + \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, p; F(\boldsymbol{\theta}, h_t, s))$
    end for
end for
$\bar{t} \leftarrow \underset{t \in \boldsymbol{\rho}}{\arg\min}_k\, \mathcal{L}_t$
return $\bar{t}$
```

B. Experiment Details

B.1. Evaluation Tasks and Models

We apply our method to a wide range of state-of-the-art open-source LLMs, including: 1) RoPE (Chen et al., 2023a) models: LLaMA-2 (7B, 13B) (Touvron et al., 2023), Mistral-7B (Jiang et al., 2023a), Gemma-7B (Team et al., 2024), Qwen1.5-7B (Bai et al., 2023a); 2) Context window extended models: Vicuna (7B, 13B) (Chiang et al., 2023); 3) Alibi (Press et al., 2022) models: MPT-30B (Team, 2023). All the models we use are instruction-tuned versions.

And we evaluate the performance across three aspects: 1) Position-bias-related tests on NaturalQuestion multi-document QA (Liu et al., 2024b) and KV retrieval (Liu et al., 2024b) with ground-truth at different positions in the prompt. The NaturalQuestion task includes 20 documents with a prompt length of about 2.3k tokens, while the KV retrieval task includes 140 KV pairs with an average length of about 10k tokens. 2) General long-context benchmark on LongBench (Bai et al., 2023b), including multi-document QA, single-document QA, summarization, few-shot learning, synthetic tasks, and code completion, totaling 16 tasks with an average length of 37k tokens. 3) Position-sensitive tasks on timeline reordering in LooGLE (Li et al., 2023), with an average length of 10k tokens. For prompts that exceed the context windows of LLMs, we follow LongBench's approach by truncating from the middle and retaining the head and tail of the prompt to fit within the context windows. We use the provided metrics and scripts from the following benchmarks for evaluation.

B.2. Baselines

We include two training-free positional bias mitigation approaches as our baselines: (i) Original, the original results of LLMs with the ground-truth at different positions in the prompt. (ii) w/ Ms-PoE (Zhang et al., 2024), uses a head-aware position embedding scaling method to mitigate position bias. We follow the paper's settings and apply scaling coefficients of 1.2 to 1.8 from the 3-rd layer.

B.3. Datasets Details

We choose NaturalQuestion Multi-document QA and Key-Value Retrieval datasets used in "lost in the middle" paper (Liu et al., 2024b) to evaluate the degree to which our method alleviates position bias. NaturalQuestion Multi-document QA require the model to answer the question based on one key information document which is inserted in a long context consisting of many irrelevant documents. And Key-Value Retrieval needs the model to retrieve the value corresponding to the given key from a list consisting of hundreds of Key-Value pairs. These two datasets are both classic in-context tasks which aim to evaluate the differences of model performance when key information is located at different positions in the context. The evaluation metric is accuracy, based on whether the model's response contains a string of the correct answer. In addition, we evaluate our method's improvements across multi task types, using LongBench (Bai et al., 2023b), a benchmark for bilingual, multitask, and comprehensive assessment of long context understanding capabilities of LLMs. It contains six major categories, covering single-document QA, multi-document QA, summarization, few-shot learning, synthetic tasks and code completion. The evaluation metrics are: F1 for single-document QA and multi-document QA, Rouge-L for summarization, accuracy (exact match) for few-shot learning and synthetic tasks, and edit similarity for code completion. During inference, since the original context may sometimes be too long, the input sequences will be truncated in the middle part to avoid exceeding the context window of the model.

B.4. Additional Implemention Details

In this paper, we implement our approach using PyTorch, HuggingFace Transformers, and FlashAttention (Dao, 2023) in an A100 GPU. To ensure stable and reproducible results, we use greedy decoding in all experiments. For the search part, we set the top-k size of positional hidden states to 10 and ε to $L/4$, where L is the number of layers. The validation set is a synthetic KV retrieval dataset consisting of 100 examples, which do not overlap with the test set. The search process takes approximately 10 minutes. For the scaling part, we only modify the intermediate layers of the model to minimize the negative impact on performance. The details of the scaling dimensions, layer ranges, and factors are shown in Table 4.

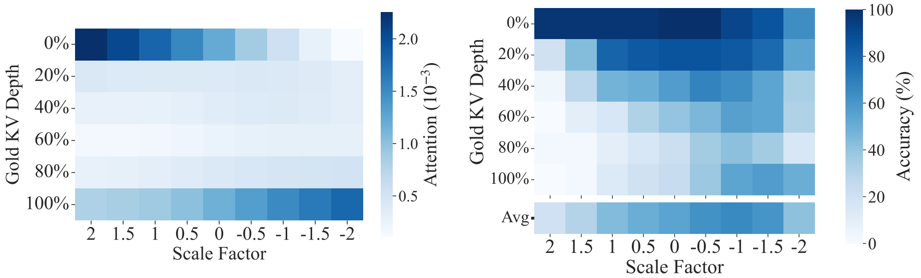
B.5. Analysis

Curve Fitting When we perform curve fitting on $h(p)$, we use least-squares cubic polynomial fit. And when judging its monotonicity, we skip the first 100 positions because the first a few values are often outliers. Since $h(p)$ is originally a discrete function, in practice, we employ the second-order difference to approximate the second-order derivative when computing smoothness.

Ms-PoE on Mistral When applying Ms-PoE (Zhang et al., 2024) to mistral-7b (Jiang et al., 2023a) with its default parameters (minimal scale factor is 1.2 and maximal is 1.8), we found the model fail to generate normal responses, so we set the maximal scale factor to 1.2, under which Ms-PoE (Zhang et al., 2024) is equal to PI (Chen et al., 2023a) with scale factor 1.2.

Ablation of the Searching Algorithm We conducted ablation experiments to demonstrate the necessity of using the three indicators (monotonicity, smoothness, validation loss) in our searching algorithm. Ours w/o monotonicity means we just select top-10 smoothest dimensions and then use the validation loss to determine. Ours w/o smoothness means we select top-10 dimensions with the highest number of monotonic layers and then use validation loss. Ours w/o validation loss means we first select top-10 dimensions with the highest number of monotonic layers and then just choose the smoothest one among them.

From Bias to Balance As shown in Table 1, there is an phenomenon that our method mainly benefits when the key information is not at the beginning, but can often decrease performance if the model performs significantly better when the key information is at the beginning. It reveals a possible fact that the positional hidden may be an important factor causing the model to miss the rear parts of the context while focus too much to the beginning parts. Therefore, scaling such dimension can shift the model's attention from being too focused at the beginning to a more balanced distribution. We validated the above points by testing different scale factors, as shown in Figure 5.



(a) Scale factor v.s. Attention

(b) Scale factor v.s. Performance

Figure 5. Attention distribution and performance when scaling dimension 2393 of Vicuna-7b-v1.5-16k with different scale factors on KV retrieval (Liu et al., 2024b) of 100 KV pairs.

Scale Factor The scaling factor directly controls the degree and direction of the impact of position hidden states on position bias. As shown in Figure 5, when the scaling factor is positive, the model exhibits a clear bias towards focusing more on the beginning, while when negative, this bias shifts to focusing more on the end. The factor between 0.5 and -1 leads attention to the most balanced distribution, meanwhile, the improvement in accuracy also reaches its peak. This result proves that the positional hidden states we scaled can indeed influence the bias of LLMs towards focusing excessively on the beginning. By adjusting the coefficients appropriately, this bias can be effectively mitigated.

B.6. Ablation Study

Table 3. Average performance of different ground-truth positions using different methods on NaturalQuestions multi-document QA dataset (20 docs) (Liu et al., 2024b).

Method	LLaMA-2-7b	Vicuna-13b	Gemma-7b	Mistral-7b	Qwn1.5-7b
Original	31.3	50.2	28.0	59.5	56.8
Ours	40.6	**52.7**	**34.6**	**60.9**	**58.3**
w/o monotonicity	40.6	51.8	**34.6**	**60.9**	**58.3**
w/o smoothness	40.6	**52.7**	27.8	**60.9**	**58.3**
w/o validation set	30.1	51.8	26.5	**60.9**	**58.3**
w/ scale 2 dimensions	37.2	50.8	31.7	60.1	57.2
w/ modify last 16 tokens	41.6	51.5	**34.6**	59.7	58.1
w/ modify all tokens	**44.0**	50.8	31.7	59.5	57.4

To evaluate the contributions of different components in our method, we introduce the following sets for the ablation study: (1) Ours w/o monotonicity, w/o smoothness, and w/o validation set, which adjust the search algorithm by not considering these three indicators, respectively (details in Appendix B.4). (2) Ours w/ scale 2 dimensions, which modifies the top-2 positional hidden states simultaneously. (3) Ours w/ modify last 16 tokens and w/ modify all tokens, which adjust the range of tokens affected by the scaling operation in Equ.(4).

Table 3 shows the ablation results. It can be seen that without filtering by monotonicity or smoothness, performance may decline, and removing the validation set results in more decline in model performance. When the range of tokens or dimensions affected by scaling is expanded, most models experience varying degrees of performance loss. Considering these factors, we choose to modify only the last token and the top-1 positional dimension to achieve the best performance.

B.7. Scaled Dimensions Details

Table 4. The scaled dimensions, scale factors and applied layers of models.

Model	Dimension	Scale factor	Applied layers
LLaMA-2-7b-chat	2,393	-1	10~25
LLaMA-2-13b-chat	4,283	-1	10~34
Vicuna-7b-v1.5-16k	2,393	0	10~25
Vicuna-13b-v1.5-16k	4,923	0	10~34
Mistral-7B-Instruct-v0.2	213	0	10~25
Gemma-1.1-7b-it	1,665	0	10~22
Qwen1.5-7b-chat	1,081	0.2	10~25
MPT-30b-chat	6,926	0	10~42

The scaled dimensions, scale factors and applied layers of each model we use in out experiments are shown in Table 4.

B.8. Inference Latency

Table 5. Time consumed (minutes) of LLaMA-2-7b-chat in a single A100.

Method	KV Retrieval	NaturalQuestion
FlashAttention-2	22	14
Ours	32	15
Ms-PoE	61	26

Table 5 shows the running time of LLaMA-2-7b-chat with different methods in the KV retrieval dataset consisting of 500 samples with average length of about 10,000, and the multi-document QA dataset consisting of 500 samples with average length of about 3,300. Our method requires recompute the query and key states, thus inevitably requires more time compared to baseline, but the cost is within an acceptable range. In contrast, Ms-PoE (Zhang et al., 2024) need to compute the attention weights twice, resulting in a doubling of time consumption.

C. Obtain Attention to Key Information

To avoid the influence of internal knowledge in the model and make attention calculation simpler, we conduct a KV retrieval task, whose prompt format is as follows:

> Json data: {"os08jbk1limft6wgxeda": "imx6lyp4b8ogjaq7ret1", ...(n key-value pairs)} The value of key "os08jbk1limft6wgxeda" is "

The last token of the prompt will directly take on the task of predicting the answer, i.e., the value which need to be retrieved. Hence, the last token's attention weights to the previous text can reflect whether it accurately retrieves the key information. We define the model's attention (in some layer) to the key information as A_G in Eq 6, where G represents the set of token positions corresponding to where the key information is at, l is the position of the last token of the prompt, and $a_{l,j}$ represents the attention weight of the l-th token to the j-th token. By shifting G, we use the same method to calculate its attention to each other KV pairs.

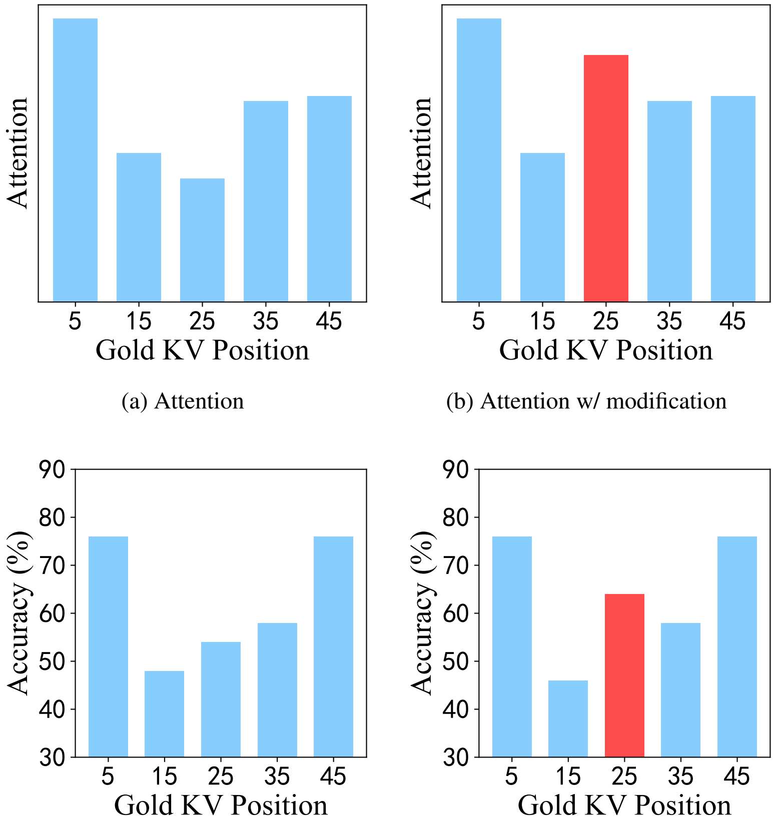
$$A_G = \frac{1}{|G|} \sum_{j \in G} a_{l,j} \tag{6}$$

D. Related Works

Long-Context LLMs Recent research has focused on expanding the context window size of LLMs. The main approaches include: 1) Staged pre-training (Nijkamp et al., 2023; Fu et al., 2024): Gradually increasing the context window size during training. 2) Modifying or interpolating position embeddings (Press et al., 2022; Chen et al., 2023a; Peng et al., 2023; Ding et al., 2024). 3) Utilizing external memory modules for context storage (Bertsch et al., 2023; Tworkowski et al., 2023). 4) Expanding computations across multiple devices in a distributed manner (Liu et al., 2023). While these methods address context window expansion, their impact on positional bias in downstream tasks has yet to be discussed.

Addressing Position Bias Although LLMs incorporate explicit positional information through methods like RoPE (Su et al., 2024) or Alibi (Press et al., 2022), studies such as (Liu et al., 2024b; Kamradt, 2023) have found that LLMs exhibit varying degrees of position bias, referred to as "lost in the middle." Recent works aimed to mitigate this issue and improve LLM performance in long-context scenarios can be categorized as follows: 1) RoPE-based methods: These approaches modify the RoPE computation process to alleviate long-distance information decay, including Attention Bucket (Chen et al., 2023b), which uses an ensemble of multiple RoPE bases to mitigate position bias, and Ms-PoE (Zhang et al., 2024), which dynamically interpolates with a small coefficient for different heads. 2) SFT-based methods (Junqing et al., 2023; Yu, 2023; An et al., 2024): These methods construct data with more diverse key information distributions or employ system2think SFT tasks to mitigate position bias. They require further training of the model. 3) Attention mask-based methods (He et al., 2024): These methods modify attention mechanisms, including Attention Transition (Gao et al., 2023), which redirects attention to significant parts of the context and Stable Mask (Yin et al., 2024), which introduces pseudo attention into the causal mask, ensuring stable attention distribution when facing lengthy texts. 4) Prompt-based methods (Jiang et al., 2023b; Peysakhovich & Lerer, 2023): These methods introduce an external module to reorder or compress information in the prompt, thereby mitigating position bias.

E. Attention v.s. Performance



(a) Attention

(b) Attention w/ modification

(c) Accuracy

(d) Accuracy w/ modification

Figure 6. Distribution of attention weight and accuracy as the ground-truth KV is placed at different positions in the prompt. (b) and (d) are situations when the attention on the 25th KV pair is modified.

As illustrated in Figure 6, when we manually multiply all the attention weights to the tokens belong to the key information (here we only choose the 25th KV pair, as shown in Figure 6b) by 2 in the model's forward process on the KV retrieval task, the corresponding retrieval accuracy of the 25th KV also show improvements, while the other parts mainly keep unchanged, as shown in Figure 6d. This result proves that the attention weights for the key information is positively correlated with the retrieval accuracy.

F. How We Modify Causal Mask and Position Embedding in KV Retrieval

In the method 1 in section 2.2, we crop the causal mask to let the "key tokens" unable to attend the previous tokens. As shown in Figure 7, the white part represents the cropped part, which means attention weights are 0, and the orange part represents the attention between tokens within key tokens. In addition, we have retained the attention of key tokens to the first token to maintain the stability of attention distribution. What is more, we only modify the causal mask in layers 1~8, but as the results, the attention to the key information is still significantly improved in layers 15~31, which indicates the positional information generated by causal mask in former layers can be transmitted to latter layers using posisional hidden states as the medium, thus modifying the causal mask solely in the former layers can induce a profound shift in the model's comprehension of positional information.

In the method 2 and 3 in section 2.2, we modify the position embeddings through altering the position ids. The specific operation is shown in the Figure 8, in which we directly replace the position ids corresponding to the key tokens with the position ids of the starting tokens (or the ending tokens) , and actually only the attention weights of the last token to previous tokens are modified. We apply this modification in all the layers. Compared to modifying the causal mask, if only modify position embedding in former layers, the attention in the latter layers remains almost unchanged, which indicates the positional information generated by position embedding may be temporary and can hardly be transmitted across layers.

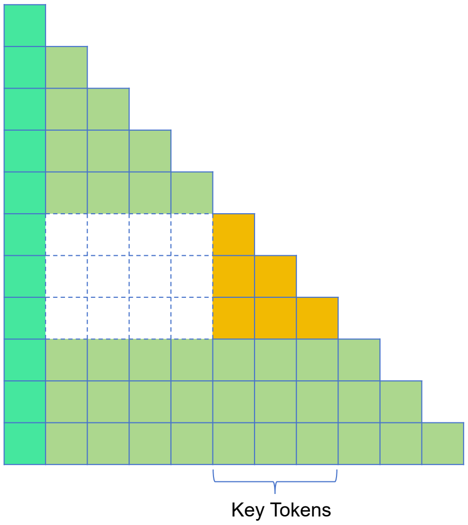


Figure 7. Cropping the causal mask to let key tokens unable to see previous tokens, except the first token.

G. Perturbation on Causal Mask and Position Embedding

To further explore the origin of these position hidden states, we performed perturbation experiments. As depicted in Figure 9c, subtracting 200 from the position ids corresponding to the 400th to 600th tokens (reducing PE) had only a minor effect on the position hidden states, whereas, in Figure 9b, crop the causal mask to make the 400th to 600th tokens unable to attend the 1st to 400th tokens (cropping causal mask) led to significant fluctuations in positional hidden states of the 400th to 600th tokens. This result proves the causal mask is the main factor causing this kind of positional hidden states, and it is the token's position in the causal mask that determines its value in the positional hidden states, but not position ids of position embedding.

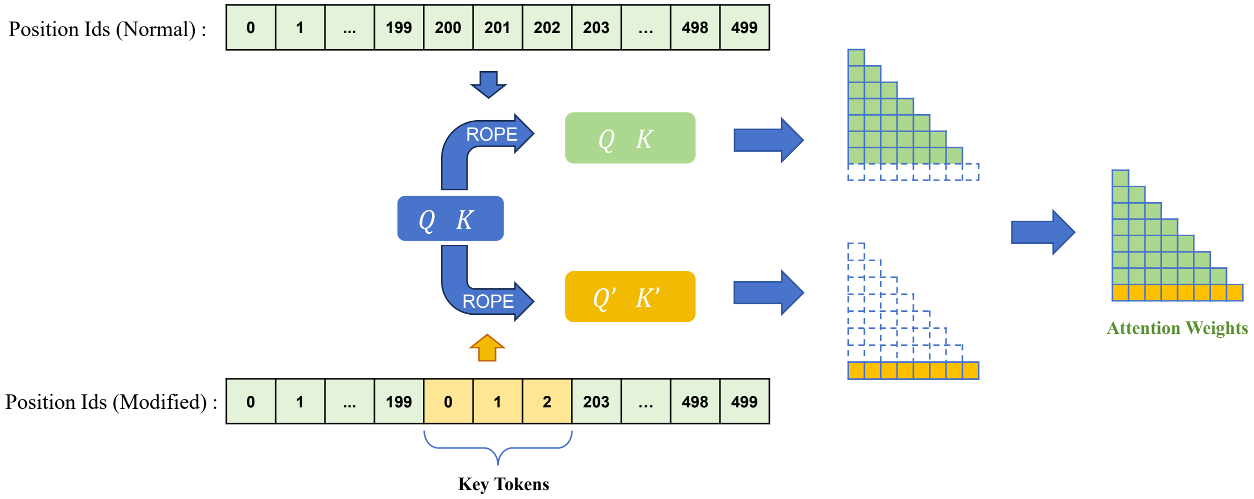
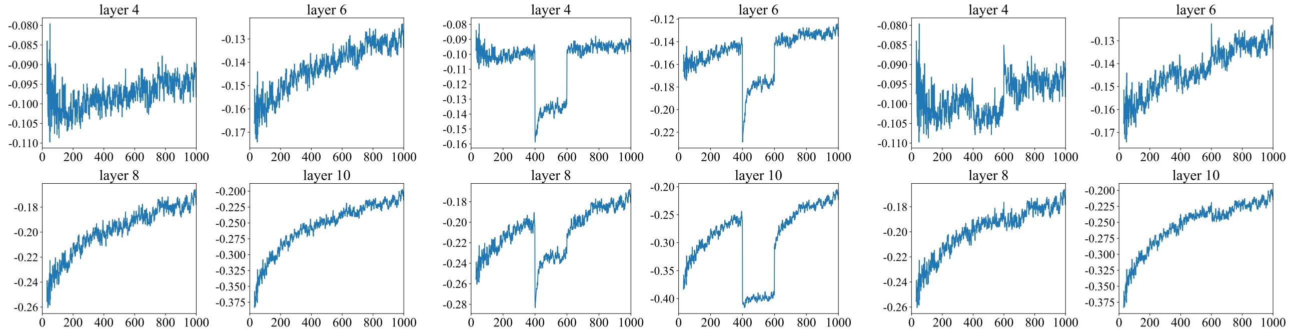


Figure 8. Shifting position ids to the start (PE to beginning).



(a) Original hidden state

(b) Crop mask

(c) Reduce PE

Figure 9. We performed perturbation experiments on the causal mask and position embedding (PE), showing the dimension 213 of hidden states of Mistral-7b (Jiang et al., 2023a) using randomly synthesized corpus as input.

H. Does this Method Compromise the Ability to Perceive Positional Information?

To prove our method is harmless for position-sensitive tasks though eliminating some positional information, we conduct the timeline reorder task from LooGLE (Li et al., 2023), whose objective is to arrange the events in accordance with their chronological sequence as dispersed throughout the extensive text. The results in Table 6 proved our methods will not impair model's performance on position-sensitive tasks. This also reflects that the positional information we eliminate may not be necessary for the model to function well.

I. Attention Distribution Layer-wise and Head-wise

Figure 10 shows Mistral-7b's attention to each KV pair of each layer (average across all attention heads) in the context in a KV retrieval task when the gold KV is put at different positions. The y-axis is the gold KV's position, x-axis is each KV's position, and the scale of the colorbar represents attention (10^{-3}). We can observe that diagonal patterns, which indicates the attention is concentrated on the "key tokens", appear only in the latter layers (start from layer 14), and may be a manifestation of retrieval behavior. In contrast, the former layers only focus on the beginning or end, regardless of where the key information is located.

Figure 11 shows the head-wise situation of layer 15. We can see actually only a portion of attention heads exhibit diagonal patterns, which may correspond to *retrieval heads* (Wu et al., 2024). The attention distribution in these heads also shows a pattern corresponding "loss in the middle", being larger at the beginning or end while significantly smaller at the middle.

Table 6. Performance of difference models on position-sensitive task-timeline reorder (Li et al., 2023).

Model	Accuracy
Vicuna-7b-v1.5-16k	20.83
Vicuna-7b-v1.5-16k w/ Ours	20.83
Qwen1.5-7b-chat	28.13
Qwen1.5-7b-chat w/ Ours	28.13
Mistral-7B-Instruct-v0.2	18.75
Mistral-7B-Instruct-v0.2 w/ Ours	19.79

J. Positional Hidden States Visualization

We shown various models' positional hidden states of each layer in Figure 12. When visualizing, we discarded the first 30 tokens because the hidden states values of these tokens are often huge (usually hundreds of times larger than the normal value (Sun et al., 2024)), which can disrupt monotonicity. We observed its monotonic trend first appears just in the first layer (actually just after the first attention mechanism), and continue to be more marked.

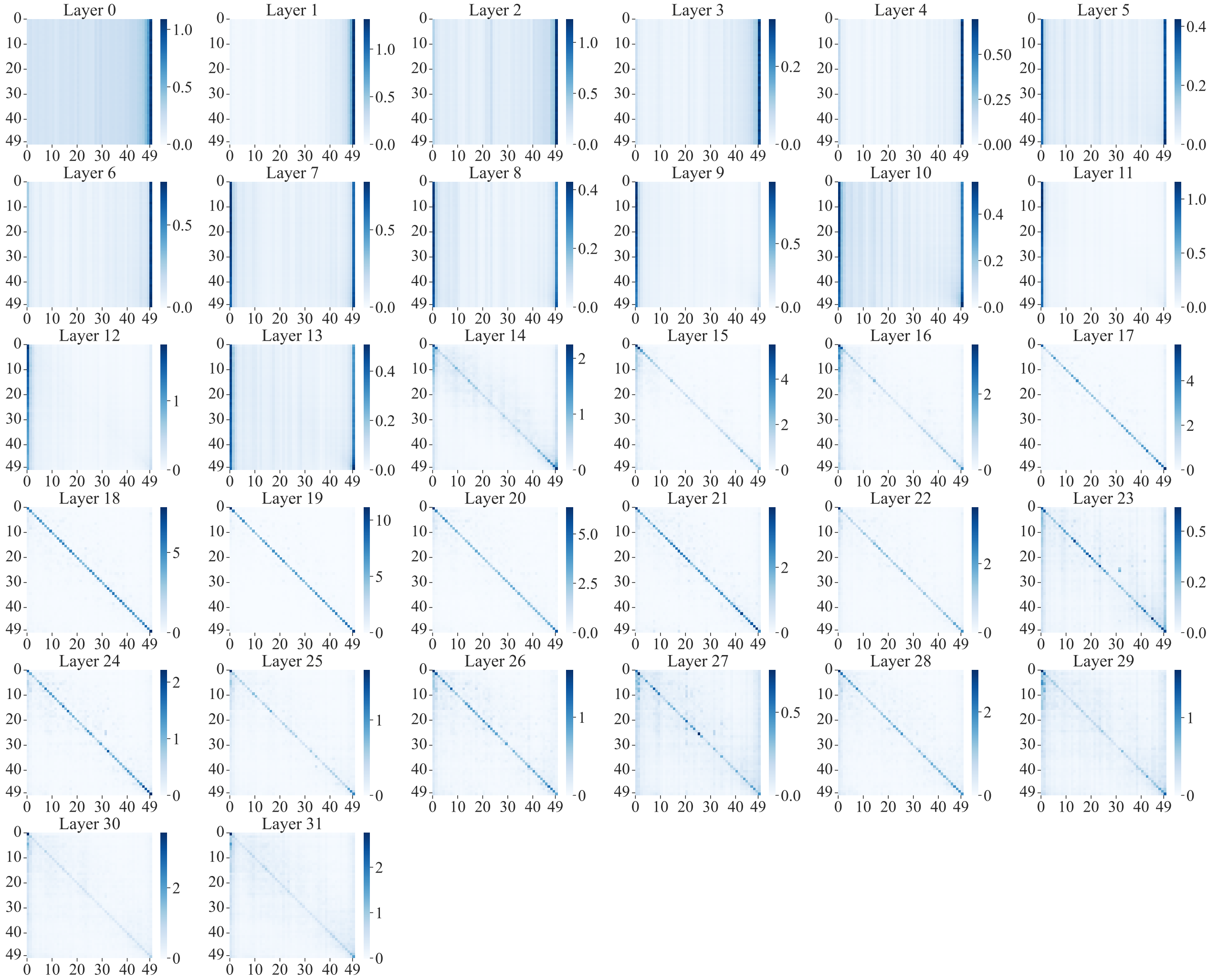


Figure 10. The average attention weight distributed on each KV, of all the 32 layers of Mistral-7b, on a 50 KV pairs retrieval task, when the gold KV is put at each different position.

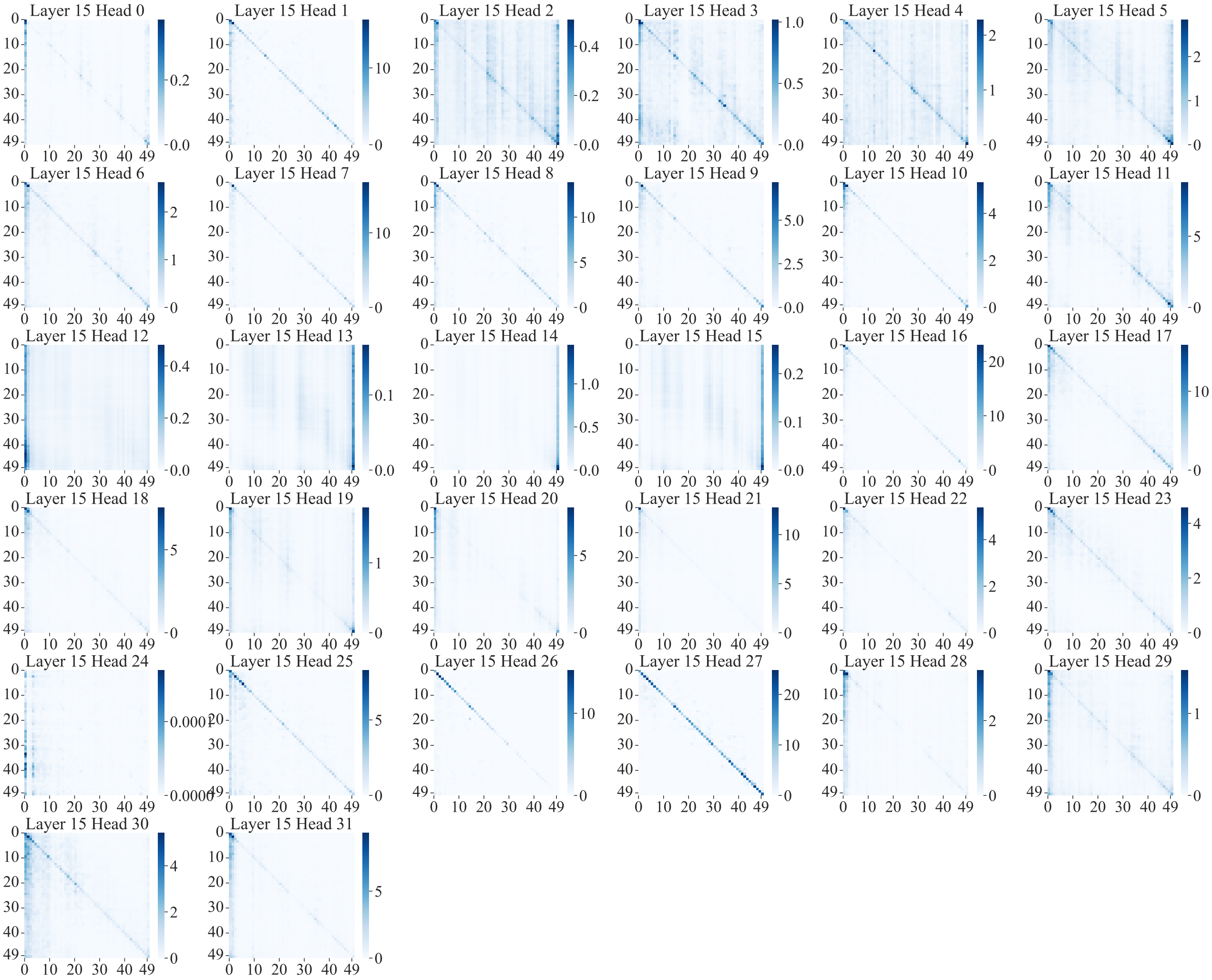
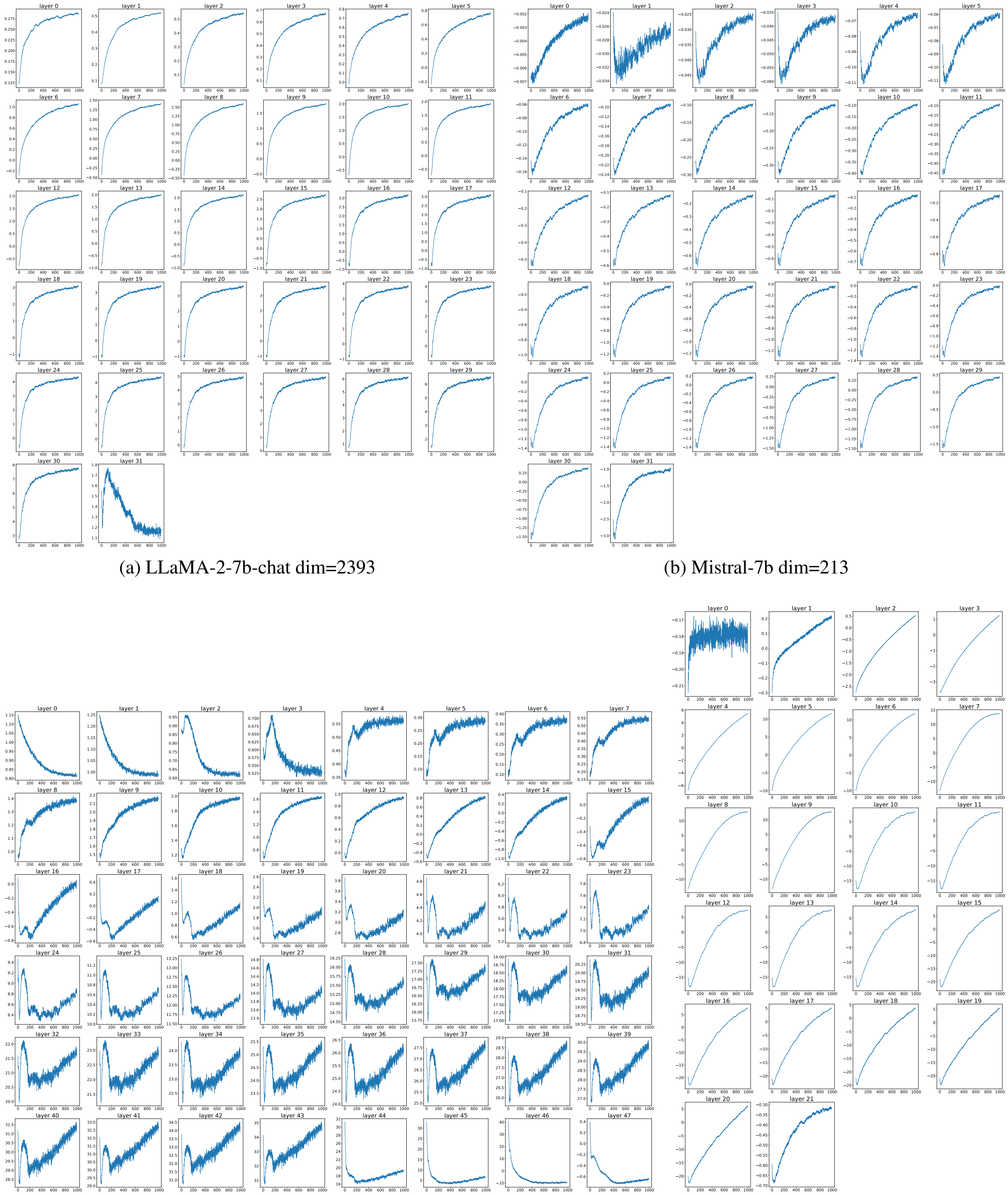


Figure 11. The average attention weight distributed on each KV, of all the 32 attention heads of layer 15 of Mistral-7b, on a 50 KV pairs retrieval task, when the gold KV is put at each different position.



(a) LLaMA-2-7b-chat dim=2393

(b) Mistral-7b dim=213

(c) MPT-30b dim=6926

(d) Tinyllama-NoPE dim=1156

Figure 12. Positional hidden states output by each layer of LLaMA-2-7b-chat, Mistral-7b-Instruct-v0.2, MPT-30b-chat and TinyLlama-NoPE-1.1B. The x-axis represents the position, and the y-axis represents the value of the states.