PEAR: *P*osition-*E*mbedding-*A*gnostic Attention *R*e-weighting Enhances Retrieval-Augmented Generation with Zero Inference Overhead

Anonymous Author(s)
Submission Id: 74*

Abstract

Large language models (LLMs) enhanced with retrieval-augmented generation (RAG) have introduced a new paradigm for web search. However, the limited context awareness of LLMs degrades their performance on RAG tasks. Existing methods to enhance context awareness are often inefficient, incurring time or memory overhead during inference, and many are tailored to specific position embeddings. In this paper, we propose **P**osition-**E**mbedding-**A**gnostic attention **R**e-weighting (*PEAR*), which enhances the context awareness of LLMs with zero inference overhead. Specifically, on a proxy task focused on context copying, we first detect heads which suppress the models' context awareness, thereby diminishing RAG performance. To weaken the impact of these heads, we re-weight their outputs with learnable coefficients. The LLM (with frozen parameters) is optimized by adjusting these coefficients to minimize loss on the proxy task. During inference, the optimized coefficients are fixed to re-weight these heads, regardless of the specific task at hand. Our proposed *PEAR* offers two major advantages over previous approaches: (1) It introduces zero additional inference overhead in terms of memory usage or inference time, while outperforming competitive baselines in accuracy and efficiency across various RAG tasks. (2) It is independent of position embedding algorithms, ensuring broader applicability. We promise to open our code[1].

CCS Concepts

• **Computing methodologies** → **Natural language generation**.

Keywords

Retrieval-Augmented Generation, Large Language Model, Context Awareness, Re-weighting Attention Heads

ACM Reference Format:

Anonymous Author(s). 2025. *PEAR*: *P*osition-*E*mbedding-*A*gnostic Attention *R*e-weighting Enhances Retrieval-Augmented Generation with Zero Inference Overhead. In *Proceedings of The Web Conference 2025 (WWW).* ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

[1] https://anonymous.4open.science/r/PEAR-RAG-B95A

WWW, 28 April–02 May, Sydney, Australia

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Retrieval-augmented generation (RAG, [11]) is widely utilized to enhance large language models (LLMs) on tasks like question answering. Typically, an RAG framework retrieves documents related to users' questions from external knowledge bases or web pages, and then arranges them in the LLMs' context as the references to form answers. This LLM-based question-answering paradigm has given rise to a promising web search paradigm [19, 22].

Recent research demonstrated LLMs' limitations on context awareness, especially when processing long context. These limitations in LLMs' context awareness challenge the effectiveness and robustness of RAG frameworks. For instance, Liu et al. [13] found that when performing in-context retrieval tasks, LLMs exhibit insensitivity to information located in the middle of the context, a phenomenon referred to as "lost in the middle." Chen et al. [2] identified a mathematical property in rotary position embedding (RoPE, [27]) that results in LLMs assigning less attention to specific contextual positions, leading to varying context awareness throughout the entire context.

Existing approaches to enhancing LLMs' context awareness are inefficient in terms of memory and time cost. Some works [23] segment and re-arrange the input context, with the assumption that placing important information in positions the model attends well can improve RAG's effectiveness. This method incurs additional inference time costs, negatively affecting user experience, as it requires multiple forward passes to obtain attention weights for guiding segment rearrangement. Another group of studies modifies the model's working mechanism, specifically by employing a set of position embeddings to adjust the attention preferences of attention heads. While these methods are input-agnostic, they also lack efficiency due to "parallelable" forward passes [2] with slightly increased time cost in additional aggregation operation, or disrupt the parallelism of multi-head attention (resulting in increased time cost) to achieve low memory cost [12], alternatively, still requiring "non-parallelable" multiple forward passes [36] which is similar to [23]. Moreover, these studies are mainly designed for RoPE and face challenges in generalizing to other position embedding algorithms, limiting their broader applications.

In this paper, we introduce **P**osition-**E**mbedding-**A**gnostic attention head **R**e-weighting (PEAR), which unleashes the context awareness of LLMs, thereby improving their RAG performance. *PEAR* achieves zero additional overhead in memory usage and inference time. Our motivation relies on the following facts:

(1) Prior research [15, 17] detected some attention heads decreasing the language model's prediction confidence by suppressing the flow of contextual information to the final position within the context, where the output is to be generated.

Table 1: An example input for the proxy task, where unique letters representing distinct tokens, with $n = 4$. For example, at position $n + i = 5$ (with $i = 1$), when an LLM receives "ABCDA" as input, it is likely to output "B." This happens because the last occurrence of "A" in the preceding context is followed by "B." If a head suppresses copying "B" from position $i + 1 = 2$ to position $n + i = 5$, it could negatively impact RAG performance.

Input Sequence	A	B	C	D	A	B	C	D
Position Index	1	2	3	4	5	6	7	8

(2) This suppression negatively impacts LLMs' context awareness, particularly abilities in in-context retrieval and context integration, which are crucial for effective RAG.

As a result, we contend that such suppression mechanism in LLMs can be safely[2] weakened in RAG scenarios, thereby improving the RAG performance of LLMs. Our proposed *PEAR* includes two stages:

In the first stage, we discover attention heads negatively affect performance on a proxy task. The proxy task involves feeding the model a random token sequence of length $2n$, which consists of a duplicated sub-sequence of length n. Table 1 illustrates an example input. At position $n + i$, the model typically predicts the token from position $i + 1$, as the natural continuation for a semantically meaningless context is to copy the existing in-context token pattern [16]. The negatively impactful attention heads are discovered using the path patching technique [31]. Since this proxy task is free from semantic bias and requires both in-context retrieval and generation based on the context—fundamental capacities for RAG, we refer to discovered heads as RAG-suppression heads[3].

In the second stage, we weaken detected RAG-suppression heads by re-weighting their outputs using learnable coefficients. These coefficients are optimized by minimizing the LLM's loss on the proxy task, with the objective of next-token prediction in a supervised fine-tuning process (loss is computed only for the second half of the random-token sequence). During the optimization, the original LLM parameters are frozen. Intuitively, most of the learned coefficients are optimized to values less than one, reducing the relative weight of these heads compared to others in the same layer when multi-head outputs are aggregated. Consequently, their influence during the forward pass is weakened. Once optimized, the coefficients remain fixed and are agnostic to downstream RAG tasks.

PEAR achieves two-fold contributions:

(1) *PEAR* introduces zero inference overhead in terms of both memory usage and inference time—an advantage not achieved by competitive baselines. Across a wide range of RAG tasks, *PEAR* surpasses previous works in both efficiency and accuracy.

(2) *PEAR* is independent of specific position embedding algorithms, making it broadly applicable. We demonstrate that *PEAR* enhances the RAG performance of various LLMs using distinct position embeddings (e.g., learnable embeddings [35], RoPE [27], and Alibi [24]).

2 Related Works

In this section, we discuss two research areas closely related to this paper: enhancements to LLMs' context awareness and studies on mechanistic interpretability.

2.1 Context awareness enhancement

Many studies highlighted limitations in LLMs' context awareness. For example, Lu et al. [14] found that the order of in-context learning (ICL) demonstrations significantly affects ICL accuracy. Liu et al. [13] demonstrated that LLMs exhibit stronger awareness of content at the beginning and end of context but weaker awareness in the middle, a phenomenon termed "lost in the middle." Chen et al. [2] proposed that LLMs' context awareness fluctuates across token positions due to mathematical properties in rotary position embedding. These challenges impact applications like RAG that rely on robust context awareness.

Several approaches have been proposed to tackle these issues. However, existing methods often come at the cost of increased inference time or memory overhead. Attention Buckets (AB, [2]) enhances context awareness by integrating positional information from a set of various RoPE angles, but it incurs significant inference overhead due to multiple parallel forward passes, leading to increased memory usage. Ms-PoE [36] calculates distinct re-scaling factors for each attention head, requiring multiple non-parallel forward passes that introduce noticeable time delays. MoICE [12] builds on Attention Buckets by employing a Mixture-of-Experts (MoE, [26]) approach, treating each RoPE embedding as a unique in-context expert, thereby limiting extra attention computations to within each layer rather than across the entire forward pass.

Our proposed method, *PEAR*, enhances context awareness by weakening the RAG-suppression heads during the forward pass. It introduces no additional modules or extra forward passes, resulting in *zero* additional overhead in memory usage and inference time. Additionally, *PEAR* operates independently of position embedding algorithms, making it applicable to more LLMs compared to existing approaches.

2.2 Mechanistic interpretability

Investigating the role of a specific head during a forward pass is one of key focuses in mechanistic interpretability research. Wang et al. [31] reported that in GPT-2 small [25], the 7th head in the 10th layer, termed the "negative head," significantly hinders answer copying from context. McDougall et al. [17] comprehensively studied this head, suggesting it functions as a self-repair mechanism to prevent overconfident outputs. Lv et al. [15] found that negative heads exist across various LLMs, employing different mechanisms to mitigate overconfidence, such as generating counteracting vectors or introducing high-frequency tokens' information. This paper does not examine what specific mechanisms the heads employ

[2]"Safely" means that the parametric knowledge and fundamental capabilities remain unaffected. Detailed experimental results are presented in Section 5.4.
[3]We do not imply that these heads hinder RAG through the same mechanisms (discussed in Section 4.1).

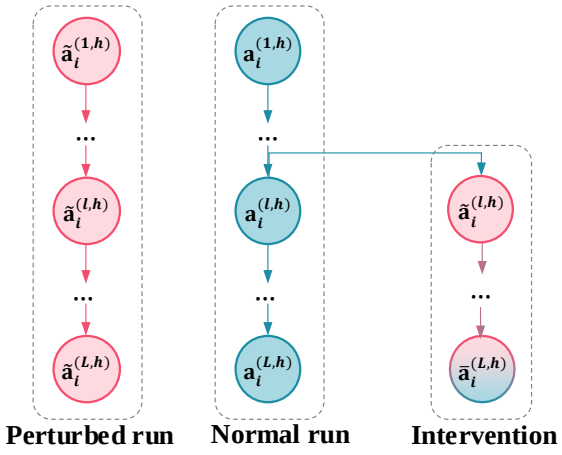


Figure 1: An illustration of causal mediation methods for circuit discovery.

to suppress RAG performance but instead aims to discover and suppress heads negatively impactful across general RAG tasks.

Yu et al. [33] detected two types of heads in Transformer-based language models during counter-factual task execution (where counter-factual knowledge is provided in the context): memory heads, which prefer to use stored knowledge, and in-context heads, which prefer to use facts in the context. However, when re-weighting these heads, only reducing the weight of memory heads successfully enhances the model's preference of using contextual knowledge, while enhancing or mitigating in-context heads does not bring much influence. Moreover, existing results only demonstrate effectiveness of these heads in the "country-capital" task (i.e., prompting the model to answer the capital city of the given country). Additionally, there is no evidence suggesting that re-weighting these heads improves the comprehensive context awareness of LLMs. We owe these failures to their head detection method, and the same re-weighting value applied for all the heads of the same type. By contrast, in our proposed *PEAR*, each individual head is re-weighted by a specific learnable coefficient, which is optimized through a proxy task independent of downstream tasks.

Wu et al. [32] also relates to head re-weighting. They reported that diminishing induction heads [21], which are known for handling context copying, can impair LLMs' long-context performance. However, they did not offer a solution to improve context awareness.

3 Preliminaries: Discovery of influential attention heads

For a particular task, research has shown that only a sparse sub-network is activated during the forward pass in Transformer language models [6, 18, 31]. Such a sub-network is referred to as a circuit [20]. Discovering circuits provides interpretability into the working mechanisms of language models and offers insights for model enhancement.

The primary method for circuit discovery is based on causal mediation analysis. The core idea is to view the forward computation graph as a causal graph, where the output of one module serves as the input for the next. In such a case, if the output of a module is changed, the computation of subsequent modules in the causal graph is also affected, as their inputs change.

In this paper, we primarily focus on analyzing the working mechanism of attention heads in language models. We briefly introduce a paradigm from a series of works [30, 31, 34] that discovers which attention heads are crucial for processing an input sequence X of length n. Suppose the language model consists of L layers, with H attention heads per layer. Let $A^{(l,h)}$ denote the h-th attention head in the l-th layer, and let its outputs be denoted by $\mathbf{a}^{(l,h)} \in \mathbb{R}^{n\times d}$. We use $\mathbf{a}_i^{(l,h)} \in \mathbb{R}^d$, where $1 \le i \le n$, to represent the output at position i. The discovery paradigm typically includes three steps, as illustrated in Figure 1:

(1) In the *normal run*, with an input sequence X (e.g., X ="The capital of France is"), $\mathbf{a}^{(l,h)}$ for every attention head are recorded.
(2) In the *perturbed run*, the forward computation runs using the same input sequence X, but with some mediation. This mediation either changes the discrete input tokens within X by substituting specific keywords (e.g., replacing "France" with "England"), or corrupts the hidden states by adding noise. The modified $\tilde{\mathbf{a}}^{(l,h)}$ for each attention head are then recorded.
(3) We conduct an intervention on a particular head $A^{(l,h)}$ at a specific position i (e.g., the country token position in above examples) in the normal run by substituting its outputs with $\tilde{\mathbf{a}}_i^{(l,h)}$. The subsequent activations in the computational graph are then recomputed (these reccomputed activations are denoted as $\bar{\mathbf{a}}$ in the figure). If the language model's final output matches the intervention's expectation (e.g., the predicted token changes from "Paris" to "London"), the head $A^{(l,h)}$ is considered to have a positive influence on the processing of sequence X.

This overview outlines a simplified discovery paradigm; detailed measurements of intervention impact are tailored to specific experimental needs.

4 Methodology

In this section, we provide a detailed introduction to our proposed method, *PEAR*, which is executed in two stages: (1) discovering RAG-suppression heads and (2) re-weighting coefficient learning. The first stage discovers attention heads that have a negative impact on general RAG tasks based on circuit discovery for a proxy task. In the second stage, we optimize learnable coefficients to re-weight the outputs of the discovered heads, aiming to mitigate their RAG-suppression effect. These coefficients remain fixed during inference, irrespective of the specific input. Figure 2 demonstrates the overview of *PEAR*.

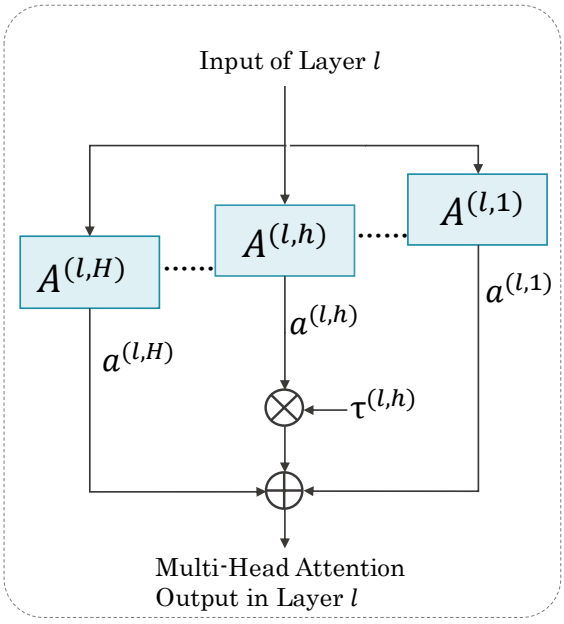


Figure 2: Suppose in layer l, $A^{(l,h)}$ is discovered as a RAG-suppression head. *PEAR* re-weights its output with a learnable scalar $\tau^{(l,h)}$.

4.1 Discovery of RAG-suppression heads

We set up a proxy task and use this task as input for circuit discovery algorithms to discover influential attention heads that hamper LLMs' performance on general RAG tasks.

Task Input. For each input sample, we create a sequence of length n, denoted as $\{x_1, \ldots, x_n\}$, where each x_i is a randomly sampled token from the vocabulary. This sequence is repeated to form an input sample $X = \{x_1, \ldots, x_{2n}\}$, with $x_i = x_{i+n}$ for $i \in [1, n]$. Research has shown that, in semantically meaningless contexts, models tend to check if the last few tokens in the sequence appeared previously and copy the suffix of their last appearance as the output [16, 21]. We consider an arbitrary LLM to successfully perform the proxy task when, at position $n + i$, the token with the highest output logits is x_{i+1}. Table 1 shows an example input.

This proxy task exhibits two key characteristics that facilitate the effective discovery of RAG-related heads:

(1) Completing the proxy task requires LLM capabilities essential for a robust RAG framework, such as in-context retrieval and generation based on context, making it suitable for discovering RAG-related attention heads.
(2) The random token composition in X ensures semantically meaningless input, minimizing knowledge bias and thereby ensuring the discovered attention heads to have general RAG-related functions, independent of specific downstream tasks.

Head discovery. We previously outlined the head discovery algorithm in Section 3. Here, we provide additional practical details for the first stage of *PEAR*.

(1) During the *normal run*, the input sequences X are constructed as above described, with a length of $2n$.
(2) In the *perturbed run*, we do not modify the input or hidden states; instead, we average the outputs of each attention head along the sequence dimension and record the resulting mean vectors.
(3) We focus on detecting changes in logits at position $2n - 1$, where the model is expected to copy the token from position n. Consequently, we intervene at $\mathbf{a}^{(l,h)}_{2n-1}$ by replacing it with the saved mean vectors.
(4) Our intervention measurements are based on the logits difference, defined as:

$$\Delta\pi^{(l,h)} = \frac{\tilde{\pi}^{(l,h)}_{2n-1}[x_n]}{\pi_{2n-1}[x_n]} - 1, \tag{1}$$

where π_{2n-1} represents the final logits at position $2n - 1$ during the *normal run*, and $[x_n]$ denotes selecting the value of the token x_n from the logits. $\tilde{\pi}^{(l,h)}$ indicates the logits after intervention on $A^{(l,h)}$. We contend that a higher value of this metric suggests a stronger suppression effect from $A^{(l,h)}$.
(5) For an arbitrary LLM, we repeat the proxy task multiple times with varying values of n to mitigate bias in context length. The final metric score for each head is the average of the results from these repeated experiments. The detailed setup is provided in Section 5.1.
(6) Based on the final metric scores, we identify the heads with the top-K most negative influence on the proxy task as a set $\mathcal{S}$, defined as:

$$\mathcal{S} = \{A^{(l,h)} | A^{(l,h)} \text{ has one of the top-}K \text{ values of } \Delta\pi^{(l,h)}\}.$$

These heads are collectively referred to as RAG-suppression heads.

Notably, we do not suggest that the heads we discovered suppressing RAG tasks operate through the same mechanisms. In Section 5.6, we demonstrate that these heads may serve various functions, such as copy suppression [17], incorporating high-frequency token information [15], or influencing the behavior of other heads to indirectly affect outputs [31]. Analyzing the specific mechanism for each head is not the focus of this paper and is left for future works.

4.2 Re-weighting coefficient learning

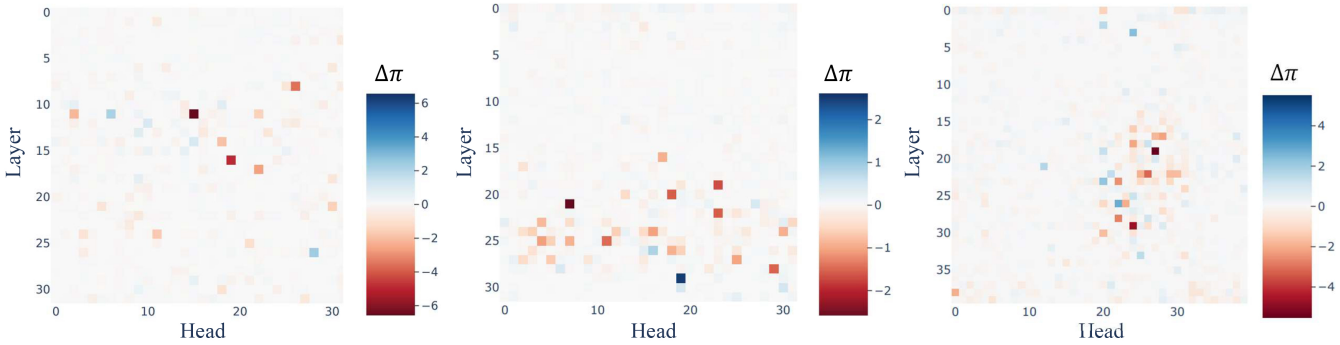
Optimization. In standard multi-head attention mechanisms, the outputs of all attention heads are aggregated with equal weighting. We propose that re-weighting these relative aggregation weights to values less than 1 can mitigate the RAG-suppression effect from our discovered heads. To implement this, we modify the forward computation by multiplying the output of each head, $A^{(l,h)}$, in the set $\mathcal{S}$ by a learnable scalar, $\tau^{(l,h)}$, referred to as the re-weighting coefficient. The modified output for each head is:

$$\mathbf{a}^{(l,h)} = \tau^{(l,h)} * \mathbf{a}^{(l,h)}, \text{ for each } A^{(l,h)} \in \mathcal{S}. \tag{2}$$

To optimize these re-weighting coefficients for RAG-suppression heads, we freeze the original parameters of the LLM and train only the re-weighting coefficients to minimize the loss on a proxy

Table 2: Discovered RAG-suppression heads in Llama2-Chat-7B-4k, OPT-6.7B-2k and Baichuan-13B-chat-4k, respectively.

Model Name	(l, h) for discovered $A^{(l,h)}$
Llama2-7B-chat-4k	(26, 28), (11, 6), (14, 15), (30, 9), (18, 9), (15, 10), (13, 9), (12, 10), (15, 14), (10, 18), (15, 25), (19, 15), (29, 15), (14, 0), (10, 2), (31, 17), (8, 22), (17, 0), (20, 26), (9, 13), (13, 14), (7, 9), (10, 1), (15, 12), (11, 9), (15, 7), (9, 16), (26, 9), (28, 22), (15, 2)
OPT-6.7B-2k	(29, 19), (0, 22), (0, 6), (26, 16), (26, 15), (30, 19), (0, 18), (23, 30), (0, 10), (31, 31), (28, 6), (30, 30), (21, 27), (0, 17), (31, 25), (12, 23), (22, 16), (0, 0), (23, 0), (0, 1), (24, 31), (23, 8)
Baichuan-13B-chat-4k	(26, 22), (33, 25), (28, 26), (32, 13), (23, 20), (25, 24), (19, 20), (38, 16), (22, 21), (21, 12), (3, 24), (39, 39), (20, 27), (37, 21), (0, 32), (24, 39), (39, 28), (39, 20), (27, 24), (2, 20), (36, 10)



(a) Llama2-7B-chat-4k (b) OPT-6.7B-2k (c) Baichuan-13B-4k

Figure 3: Heatmaps of $\Delta\pi$ scores for each head across three LLMs ($n = 10$).

task. Importantly, the loss is calculated only over the latter half of the sequence, optimizing the coefficients to enhance in-context retrieval capacities rather than predicting the next token. Formally, our adopted loss can be written as:

$$\mathcal{L} = -\sum_{i=n}^{2n-1} \log p(x_{i+1}|x_{1:i}) \tag{3}$$

Figure 2 illustrates a re-weighting process during optimization. Notably, the re-weighting process shown in this figure adds extra multiplication operations in a forward pass. In practice, when coefficient learning ends, we re-scale $W_O^{(l,h)}$ (the output projection matrix in head $A^{(l,h)}$) by $\tau^{(l,h)}$, which is equivalent to Eq. 2 and does not add any extra computation during inference.

Inference on Downstream Tasks. We highlight several points regarding the inference process of our proposed *PEAR* on downstream RAG tasks:

(1) In downstream RAG tasks, the re-weighting coefficients are task-independent and remain fixed.
(2) RAG-suppression heads are optimized once for each LLM via the proxy task. For a new RAG task, head discovery and coefficient learning do not need to be repeated.

In theory, our approach, *PEAR*, introduces *zero* additional overhead during inference on downstream RAG tasks, as it does not incorporate extra computational modules; instead, it only adjusts the aggregation weights of specific heads. Additionally, the learning of re-weighting coefficients is independent of the LLM architecture, thus making our method compatible with various position embedding algorithms.

5 Experiments

5.1 Setup

In this section, we introduce the LLMs we used for experiments, the baseline methods for enhancing context awareness, the setups for the proxy tasks, hyperparameters for learning re-weighting coefficients.

Models and baselines. We conducted experiments with three LLMs, each employing a different position embedding algorithm: Llama2-7B-chat-4k [28] using RoPE [27], OPT-6.7B-2k [35] using learnable position embeddings, and Baichuan-13B-4k [1] using Alibi position embeddings [24].

We also compared several competitive baseline methods for enhancing LLMs' context awareness, including Attention Buckets (AB, [2]), Ms-PoE [36], and MoICE [12]. Details on these methods can be found in Section 2.1.

Detailed setups of proxy task. For head discovery, we constructed 200 task samples. In the case of the Llama and OPT models, we repeat the discovery process four times with varying values of n: 10, 15, 25, 50. For the Baichuan model, the n values are 10, 20, 50, 80.

Table 3: Performance comparison of Llama2-7B-chat-4k and its enhancements across three RAG tasks.

Method	2WikiMultiHopQA↑	MuSiQue↑	Qasper↑	Avg. ↑
Llama2-7B-chat-4k	29.50	6.50	17.00	17.67
+ Ms-PoE [36]	27.50	9.00	**18.00**	18.17
+ AB [2]	31.00	**11.00**	16.50	19.50
+ MoICE [12]	30.00	10.00	15.50	18.50
+ *PEAR* (Ours)	**35.00**	8.50	**18.00**	**20.50**

Table 4: Practical inference time (in seconds) and GPU memory cost (in GB) per test sample for different methods. For a fair comparison, Flash-Attention [3] was not applied. The experiments were conducted on a single H800-80G GPU.

Method	2WikiMultiHopQA↓	MuSiQue↓	Qasper↓	Avg.↓
Llama2-7B-chat-4k	0.63/31.33	0.70/31.33	1.23/31.33	0.88/31.33
+ Ms-PoE [36]	0.95 (+0.32) / 39.21 (+7.88)	1.11(+0.41) / 39.21 (+7.88)	1.84 (+0.61) / 34.59 (+3.26)	1.30 (+0.42) /37.67 (+6.34)
+ AB [2]	2.50(+1.87) /66.19(+34.86)	2.70(+2.00) / 66.19 (+34.86)	5.67(+4.44) /66.34(+35.01)	3.62(+2.74)/66.24(+34.91)
+ MoICE [12]	2.91(+2.28) / 79.13(+47.80)	3.06(+2.36) / 79.12(+47.79)	5.86(+4.63) / 79.10(+47.77)	3.94(+3.06) / 79.12(+47.79)
+ *PEAR* (Ours)	0.63 (**+0.00**) /31.33 (**+0.00**)	0.70 (**+0.00**) / 31.33 (**+0.00**)	1.23 (**+0.00**) / 31.33 (**+0.00**)	0.88 (**+0.00**) / 31.33 (**+0.00**)

We found that each model has a group of heads with significantly large $\Delta\pi$ values, leading us to select the K values based on the observed group sizes: 30, 22, and 21, respectively. Table 2 presents the discovered RAG-suppression heads for the LLMs under study. Figure 3 illustrates $\Delta\pi$ values when $n = 10$ for each model. Further detailed results can be found in Appendix A.

For the re-weighting coefficient learning stage, we constructed 500 task samples, setting n to 50 for all models.

Hyperparameters for re-weighting coefficient learning. We employed the AdamW optimizer with a learning rate of 0.005 and parameters $(\beta_1, \beta_2) = (0.9, 0.999)$. τ are initialized at 1.0. Training was performed for a single epoch using BF16 precision on an A100-PCIE-40GB GPU.

Difference between learning and inference. We conducted internal loading training on the re-weighting coefficients $\tau^{(l,h)}$ during the training process. However, during inference, we externally re-weighting the output matrix weights of specific heads before loading the model. Therefore, our method is simple to train and has *zero* inference overhead.

5.2 Comparison with baselines on RAG tasks

We compare *PEAR* against various baselines on RAG tasks we constructed using three datasets: 2WikiMultihopQA [8], MuSiQue [29], and Qasper [4]. The first two datasets require the model to answer questions based on multiple documents, while the third focuses on questions related to NLP research papers, formulated and answered by NLP researchers. We truncate the context to 4,000 tokens for the first two datasets; the third dataset has an average context length of 3,619 tokens.

Our experiments are conducted with Llama2-7B-chat-4k, as the baselines are tailored specifically for RoPE. We evaluate the models' performance using exact match scores, with results reported in Table 3. Notably, our method achieves the highest average improvement across all three tasks. Although *PEAR* does not achieve the top performance on the MuSiQue task, it outperforms the original model by a large margin.

Additionally, we present inference time and memory costs for these datasets in Table 4. *PEAR* does not increase GPU memory usage and inference time costs. This makes it significantly more efficient than other enhancement methods.

These experiments underscore the effectiveness and efficiency of *PEAR* in enhancing LLMs for RAG tasks.

5.3 Applicability to LLMs using various position embeddings

In this section, we demonstrate the applicability of *PEAR* to LLMs utilizing different position embeddings. We conduct a multi-document question-answering (MDQA) experiment based on data from [13], which leverage a subset of NaturalQuestions-Open [9, 10], consisting of 2,655 queries. Each query is paired with a context consisting of 10 documents with an average of 1,722 tokens. Following [2, 13], we position the gold document (i.e., the document contains the ground truth answer) at various contextual positions to evaluate the robustness of a context-awareness enhancement method. In our experiments, we set the maximum document count to 10 and assess the question-answering accuracy when the gold document is placed as the 1st, 3rd, 5th, 7th, and 10th document, respectively. Since baseline methods are not compatible with the OPT and Baichuan models, we compare *PEAR* only with the original models. The results are presented in Table 5.

Table 5: Experimental results on the MDQA task show that *PEAR* achieves the highest accuracy in 14 out of 15 comparisons across three LLMs, demonstrating its broad applicability to various position embeddings and its robustness in enhancing awareness to different contextual positions.

Position Embedding	Method	Gold Document Position					Avg.
		1	3	5	7	10	
RoPE	Llama2-7B-chat-4k	64.14	65.95	64.97	62.67	67.53	65.05
	+ Ms-PoE [36]	66.06	64.29	63.99	62.22	64.75	64.34
	+ AB [2]	**66.36**	66.14	65.25	63.20	64.93	65.18
	+ MoICE [12]	65.50	66.33	65.61	64.11	65.84	65.48
	+ *PEAR* (Ours)	62.71	**67.01**	**68.32**	**66.44**	**69.57**	**66.81**
Learnable Embeddings	OPT-6.7B-2k	19.07	15.45	17.03	16.54	22.61	18.14
	+ *PEAR* (Ours)	**20.23**	**17.18**	**17.60**	**17.22**	**22.87**	**19.02**
Alibi	Baichuan-13B-chat-4k	12.28	13.45	11.98	11.04	12.96	12.34
	+ *PEAR* (Ours)	**14.16**	**14.84**	**13.67**	**12.77**	**13.94**	**13.88**

Table 6: Results on the MMLU benchmark showing that *PEAR* does not enhance LLMs' context capacities at the expense of knowledge ability.

Model	Humanities	Social Science	STEM	Other	Avg.
Llama2-7B-chat-4k	42.55	52.29	37.14	52.47	45.81
+ *PEAR* (Ours)	42.06	52.03	36.61	52.19	45.41

Table 7: The experiment results on the question answering task with ablation settings, which show that our control over the number of suppression heads is effective.

Method	2WikiMultiHopQA	MuSiQue	Qasper	Avg.
Llama2-7B-chat-4k	29.50	6.50	17.00	17.67
+ *PEAR* (K=10)	33.00	8.00	16.00	19.00
+ *PEAR* (K=20)	33.50	8.50	16.50	19.50
+ *PEAR* (K=30)	**35.00**	**8.50**	**18.00**	**20.50**
+ *PEAR* (K=40)	32.50	8.00	17.00	19.17

Table 8: The experiment results on the MDQA task with ablation settings, which show that our control over the number of suppression heads is effective.

Method	Gold Document Position					Avg.
	1	3	5	7	10	
Llama2-7B-chat-4k	**64.14**	65.95	64.97	62.67	67.53	65.05
+ *PEAR* (K=10)	63.43	66.26	66.82	64.67	67.76	65.79
+ *PEAR* (K=20)	63.88	66.29	66.44	65.65	68.51	66.15
+ *PEAR* (K=30)	62.71	**67.01**	**68.32**	66.44	**69.57**	**66.81**
+ *PEAR* (K=40)	62.90	66.00	67.16	**66.59**	68.40	66.21

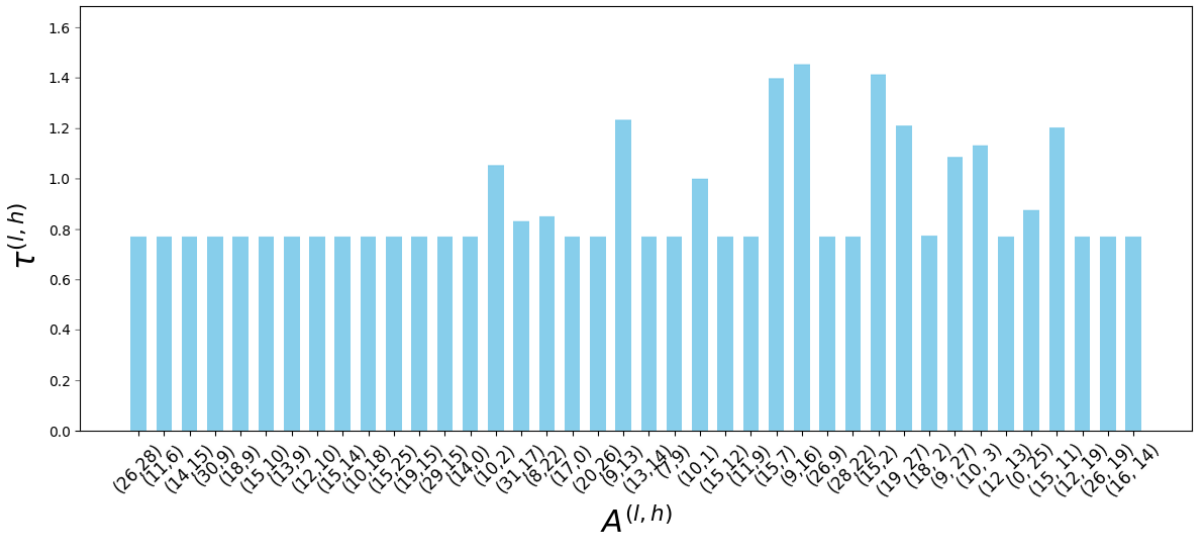


Figure 4: The learned coefficients of *PEAR* (on Llama2-7B-chat and $K = 40$).

5.4 *PEAR* does not diminish knowledge capabilities in LLMs

Previous research [5, 15] has shown that certain attention heads store or play a crucial role in eliciting parametric knowledge. This raises the question of whether *PEAR* enhances context awareness of LLMs at the expense of their ability to utilize this parametric knowledge.

To investigate this, we evaluated a *PEAR*-enhanced Llama2-7B-chat model using the MMLU benchmark [7], and the results are presented in Table 6. The performance of the enhanced Llama2-7B-chat and the original Llama2-7B-chat did not show a significant difference. Consequently, we argue that *PEAR*, through its effective head discovery and re-weighting learning approaches, does not compromise the knowledge capabilities of LLMs.

5.5 Analysis: The effect of K

While we have demonstrated the effectiveness of *PEAR* from various angles, a key point for discussion is the role of K, representing the number of heads to re-weight.

Using Llama2-7B-chat as a case study, we vary K and observe its impact on *PEAR*'s performance. Table 7 presents results on RAG tasks, while Table 8 details analysis for MDQA tasks. The findings indicate that *PEAR* performs optimally when K matches the inherent threshold of the model (i.e., $K = 30$ for Llama2-7B-chat), i.e., the number of heads with a significantly higher $\Delta\pi$ than others.

Re-weighting fewer heads fails to fully alleviate the suppression from RAG-suppression heads, while exceeding this optimal number can harm the performance of non-RAG-suppression heads, ultimately diminishing overall effectiveness.

5.6 Analysis: The Value of τ

Using Llama2-7B-chat as an example, we present the learned coefficients of PEAR in Figure 4, with heads ranked by their $\Delta\pi$ scores. Intuitively, most heads are optimized to values less than one, which reduces their relative weight compared to other heads within the same layer when multi-head outputs are aggregated. Due to BF16 training precision, many τs are optimized to the same value. However, using FP32 precision for training did not significantly impact the results.

Notably, $\tau^{(l,h)}$s for 9 heads, which have relatively low $\Delta\pi$, are greater than one. We do not attribute this to the precision of the discovery process, as constraining the re-weighting coefficients to be less than one led to suboptimal performance. Thus, a plausible explanation is that RAG suppression is a complex, cooperative effect involving multiple heads, each with distinct working mechanisms, as discussed in Section 4.1.

6 Conclusion

In this paper, we introduce *PEAR*, a position-embedding-agnostic method designed to enhance the performance of LLMs on RAG tasks with zero inference overhead. Our method is based on a certain number of RAG-suppression heads, which not only outperforms competitive baselines in both effectiveness and efficiency, but also demonstrates broad applicability across various LLMs. We also presented that *PEAR* improves context awareness in LLMs without compromising their inherent knowledge capabilities. These benefits make *PEAR* a promising approach for a wide range of applications that require robust context abilities, such as in-context learning and strict instruction following, which we leave for future research.

References

[1] Baichuan. 2023. A 13B large language model developed by Baichuan Intelligent Technology. https://github.com/baichuan-inc/Baichuan-13B/blob/main/README_EN.md

[2] Yuhan Chen, Ang Lv, Ting-En Lin, Changyu Chen, Yuchuan Wu, Fei Huang, Yongbin Li, and Rui Yan. 2024. Fortify the Shortest Stave in Attention: Enhancing Context Awareness of Large Language Models for Effective Tool Use. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 11160–11174. https://aclanthology.org/2024.acl-long.601

[3] Tri Dao. 2024. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. In *International Conference on Learning Representations (ICLR)*.

[4] Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A Dataset of Information-Seeking Questions and Answers Anchored in Research Papers. arXiv:2105.03011 [cs.CL] https://arxiv.org/abs/2105.03011

[5] Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting Recall of Factual Associations in Auto-Regressive Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 12216–12235. https://doi.org/10.18653/v1/2023.emnlp-main.751

[6] Zhuocheng Gong, Ang Lv, Jian Guan, Junxi Yan, Wei Wu, Huishuai Zhang, Minlie Huang, Dongyan Zhao, and Rui Yan. 2024. Mixture-of-Modules: Reinventing Transformers as Dynamic Assemblies of Modules. arXiv:2407.06677 [cs.CL] https://arxiv.org/abs/2407.06677

[7] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. arXiv:2009.03300 [cs.CY] https://arxiv.org/abs/2009.03300

[8] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, Barcelona, Spain (Online), 6609–6625. https://www.aclweb.org/anthology/2020.coling-main.580

[9] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics* 7 (2019), 453–466.

[10] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300* (2019).

[11] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv:2005.11401 [cs.CL] https://arxiv.org/abs/2005.11401

[12] Hongzhan Lin, Ang Lv, Yuhan Chen, Chen Zhu, Yang Song, Hengshu Zhu, and Rui Yan. 2024. Mixture of In-Context Experts Enhance LLMs' Long Context Awareness. arXiv:2406.19598 [cs.CL] https://arxiv.org/abs/2406.19598

[13] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the Middle: How Language Models Use Long Contexts. arXiv:2307.03172 [cs.CL]

[14] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (Eds.). Association for Computational Linguistics, Dublin, Ireland, 8086–8098. https://doi.org/10.18653/v1/2022.acl-long.556

[15] Ang Lv, Yuhan Chen, Kaiyi Zhang, Yulong Wang, Lifeng Liu, Ji-Rong Wen, Jian Xie, and Rui Yan. 2024. Interpreting Key Mechanisms of Factual Recall in Transformer-Based Language Models. arXiv:2403.19521 [cs.CL] https://arxiv.org/abs/2403.19521

[16] Ang Lv, Ruobing Xie, Xingwu Sun, Zhanhui Kang, and Rui Yan. 2024. Language Models "Grok" to Copy. arXiv:2409.09281 [cs.CL] https://arxiv.org/abs/2409.09281

[17] Callum McDougall, Arthur Conmy, Cody Rushing, Thomas McGrath, and Neel Nanda. 2023. Copy Suppression: Comprehensively Understanding an Attention Head. arXiv:2310.04625 [cs.LG] https://arxiv.org/abs/2310.04625

[18] Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. 2024. Circuit Component Reuse Across Tasks in Transformer Language Models. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=fpoAYV6Wsk

[19] Microsoft. 2023. Reinventing search with a new AI-powered Microsoft Bing and Edge, your copilot for the web.

[20] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom In: An Introduction to Circuits. *Distill* (2020). https://doi.org/10.23915/distill.00024.001 https://distill.pub/2020/circuits/zoom-in.

[21] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2022. In-context Learning and Induction Heads. *Transformer Circuits Thread* (2022). https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

[22] OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL] https://arxiv.org/abs/2303.08774

[23] Alexander Peysakhovich and Adam Lerer. 2023. Attention Sorting Combats Recency Bias In Long Context Language Models. arXiv:2310.01427 [cs.CL]

[24] Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. arXiv:2108.12409 [cs.CL] https://arxiv.org/abs/2108.12409

[25] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

[26] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. *CoRR* abs/1701.06538 (2017). arXiv:1701.06538 http://arxiv.org/abs/1701.06538

[27] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. RoFormer: Enhanced Transformer with Rotary Position Embedding. arXiv:2104.09864 [cs.CL] https://arxiv.org/abs/2104.09864

[28] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL] https://arxiv.org/abs/2307.09288

[29] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multihop Questions via Single-hop Question Composition. *Transactions of the Association for Computational Linguistics* 10 (2022), 539–554. https://doi.org/10.1162/tacl_a_00475

[30] Boshi Wang, Xiang Yue, Yu Su, and Huan Sun. 2024. Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization. arXiv:2405.15071 [cs.CL] https://arxiv.org/abs/2405.15071

[31] Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593* (2022).

[32] Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2024. Retrieval Head Mechanistically Explains Long-Context Factuality. arXiv:2404.15574 [cs.CL] https://arxiv.org/abs/2404.15574

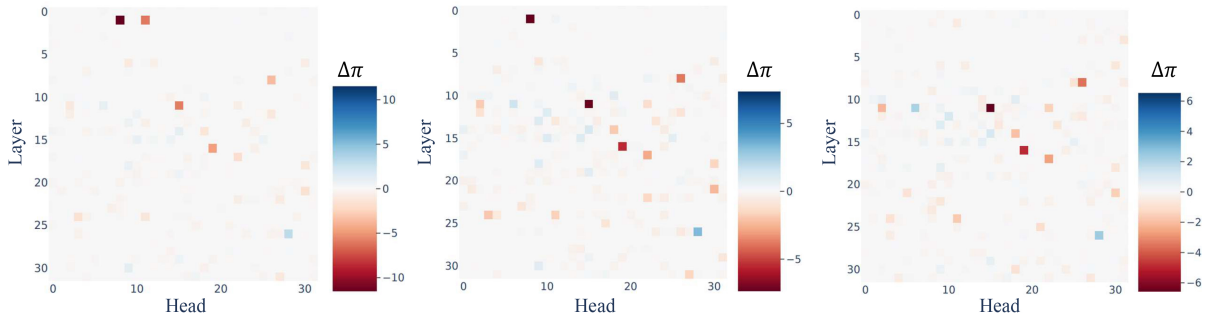
[33] Qinan Yu, Jack Merullo, and Ellie Pavlick. 2023. Characterizing Mechanisms for Factual Recall in Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 9924–9959. https://doi.org/10.18653/v1/2023.emnlp-main.615

[34] Fred Zhang and Neel Nanda. 2024. Towards Best Practices of Activation Patching in Language Models: Metrics and Methods. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=Hf17y6u9BC

[35] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open Pre-trained Transformer Language Models. arXiv:2205.01068 [cs.CL] https://arxiv.org/abs/2205.01068

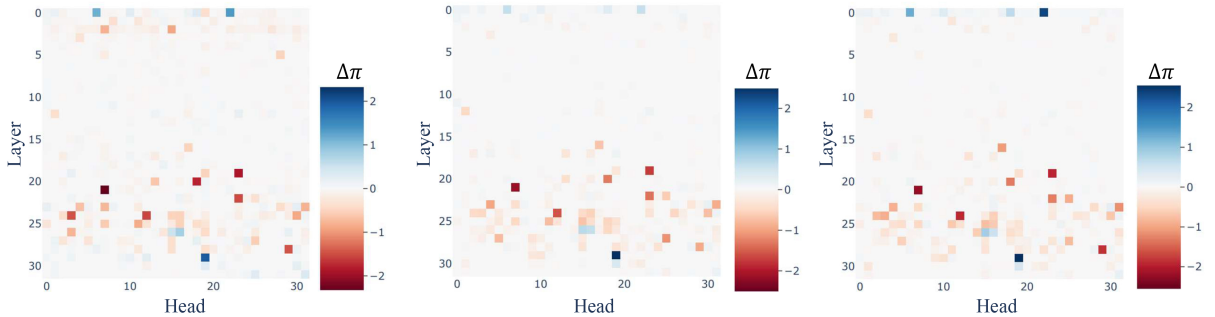
[36] Zhenyu Zhang, Runjin Chen, Shiwei Liu, Zhewei Yao, Olatunji Ruwase, Beidi Chen, Xiaoxia Wu, and Zhangyang Wang. 2024. Found in the Middle: How Language Models Use Long Contexts Better via Plug-and-Play Positional Encoding. arXiv:2403.04797 [cs.CL] https://arxiv.org/abs/2403.04797

A Head discovery results



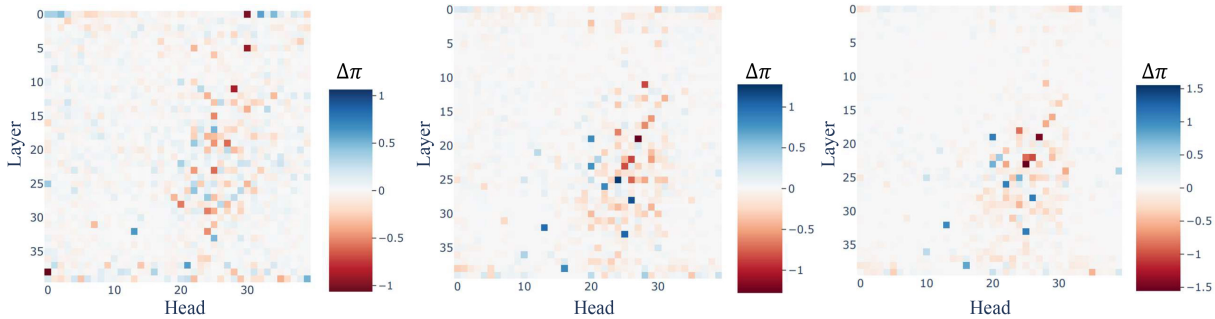
(a) $n = 15$ (a) $n = 25$ (a) $n = 50$

Figure 5: Heatmaps of $\Delta\pi$ scores for each head of llama2-7B-chat ($n = 15, n = 25, n = 50$).



(a) $n = 15$ (b) $n = 25$ (b) $n = 50$

Figure 6: Heatmaps of $\Delta\pi$ scores for each head of OPT-6.7B ($n = 15, n = 25, n = 50$).



(a) $n = 20$ (b) $n = 50$ (c) $n = 80$

Figure 7: Heatmaps of $\Delta\pi$ scores for each head of Baichuan-13B-chat ($n = 20, n = 50, n = 80$).