✂⚠ Cutting Off the Head Ends the Conflict: A Mechanism for Interpreting and Mitigating Knowledge Conflicts in Language Models

Anonymous ACL submission

Abstract

Recently, retrieval augmentation and tool augmentation have demonstrated a remarkable capability to expand the **internal memory** boundaries of language models (LMs) by providing **external context**. However, internal memory and external context inevitably clash, leading to **knowledge conflicts** within LMs. In this paper, we aim to interpret the mechanism of knowledge conflicts through the lens of information flow, and then mitigate conflicts by precise interventions at the pivotal point. We find there are some attention heads with opposite effects in the later layers, where **memory heads** can recall knowledge from internal memory, and **context heads** can retrieve knowledge from external context. Moreover, we reveal that the pivotal point at which knowledge conflicts emerge in LMs is the integration of inconsistent information flows by memory heads and context heads. Inspired by the insights, we propose a novel method called **P**runing **H**ead via **P**at**H** **P**atc**H**ing (**PH3**), which can efficiently mitigate knowledge conflicts by pruning conflicting attention heads without updating model parameters. PH3 can flexibly control eight LMs to use internal memory (↑ 44.0%) or external context (↑ 38.5%). Moreover, PH3 can also improve the performance of LMs on open-domain QA tasks. We also conduct extensive experiments to demonstrate the cross-model, cross-relation, and cross-format generalization of our method.

1 Introduction

Language models (LMs) (Brown et al., 2020; Touvron et al., 2023; OpenAI, 2023) have memorized a substantial amount of factual knowledge during pre-training, and stored the knowledge within their parameters as **internal memory** (*i.e.*, **parametric knowledge**) (Meng et al., 2022). During the inference phase, LMs rely on their internal memory to understand and generate text. However, the internal memory may be limited or outdated, making LMs prone to producing factually incorrect content.

To alleviate the problem, one promising solution is to employ additional retrievers or tools to augment LMs by providing **external context** (*i.e.*, **non-parametric knowledge**). Nevertheless, internal memory and external context can often contradict each other, which is known as **knowledge conflicts** (Longpre et al., 2021; Chen et al., 2022; Xie et al., 2023; Yu et al., 2023). Recent works have mainly investigated the behavior and preference of LMs, attempting to determine whether these models are more inclined towards internal memory or external context when faced with knowledge conflicts. However, there is a limited understanding of the underlying mechanism of knowledge conflicts. Insights into the mechanism will facilitate precise interventions at the pivotal point to mitigate knowledge conflicts, which can not only empower LMs to more reliably adhere to internal memory (*e.g.*, ignoring misleading external context) but also enhance faithfulness in generating text based on external context (*e.g.*, correcting outdated memory).

In this paper, we reveal that the pivotal point at which knowledge conflicts emerge in LMs is the integration of inconsistent information flows by various attention heads in later layers. To investigate this, we consider a simple factual recall task (*i.e.*, subject attribute prediction) inspired by the work of Yu et al. (2023). As illustrated in Figure 1, given the question (*i.e.*, "*What is the capital of France?*") and the conflicting external context (*i.e.*, "*The capital of France is Rome.*"), the model can either use internal memory (*i.e.*, "*Paris*") or external context (*i.e.*, "*Rome*") to predict the subject's attribute. Following this, we present a set of "*top-down*" analyses to locate the pivotal point where conflicts emerge and to identify the model components that are significant in knowledge conflicts, which primarily involves the following three steps:

Step 1: We start by answering the *first* question "*What function do model components serve in knowledge conflicts?*". We knock out the acti-

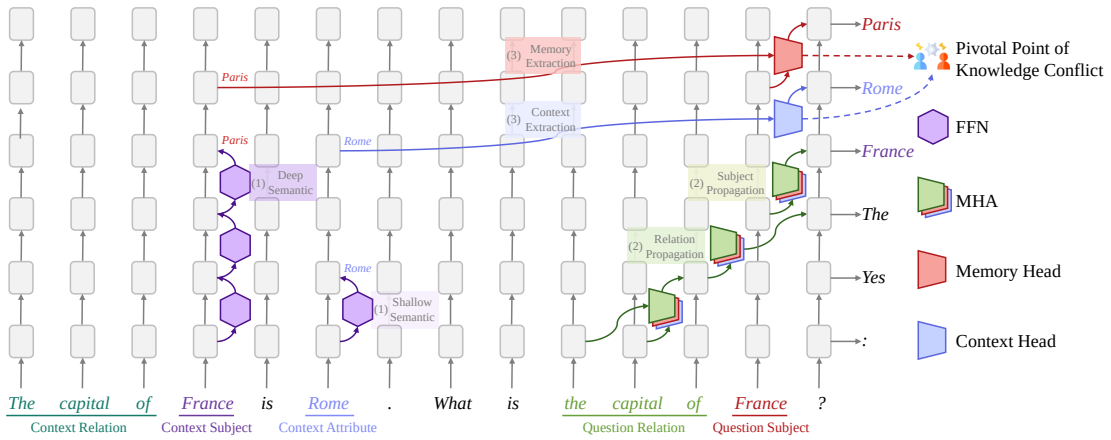


Figure 1: An illustration of the mechanism of knowledge conflicts in LMs: (1) Enriching the semantic information of context subject and context attribute; (2) Propagating question information to the last token through MHAs; (3) Extracting attribute information through memory attention heads and context attention heads at later layers.

vations to examine the functionality of *multi-head attention* (MHA) blocks and *feed-forward network* (FFN) blocks. We find that FFNs enrich the semantic information of input elements in early layers, while MHAs play an important role in passing information to the last token in later layers; **Step 2**: Based on this, the *second* question naturally arises, namely "*When and where do MHAs pass information to the last token?*". We investigate the MHAs by knocking out the attention weights from the last token to other input elements. Results reveal that the question information is first propagated to the last token, and then the last token extracts attribute information from the subject and the attribute in the context; **Step 3**: Inspired by this, we aim to answer the *final* question "*How do MHAs extract attribute information under knowledge conflicts?*". We find that some attention heads in late MHAs play opposite roles, where **memory heads** can recall attributes from internal memory, and **context heads** can retrieve attributes from external context. According to our findings, the mechanism by which LMs use both internal memory and external context can be summarized as three stages in Figure 1: (1) Enriching semantic information; (2) Propagating question information; and (3) Extracting attribute information, where knowledge conflicts arise at the third stage, due to the inconsistent information flows between memory heads and context heads.

Inspired by our insights into knowledge conflicts, we propose a minimally-invasive control method called **P**runing **H**ead via **P**at**H** **P**atc**H**ing (**PH3**), which can efficiently mitigate knowledge conflicts by intervening on attention heads without updating model parameters. First, we use the **path patching** (Goldowsky-Dill et al., 2023; Wang et al., 2023a) technique to localize important memory heads and context heads. Our method can avoid the noise interference of other heads, enabling a more accurate calculation of the importance score for the target head. Then, we perform **structured pruning** on those negative attention heads to mitigate conflicts. In this way, our method can flexibly control LMs to use internal memory or external context. Experimental results on the `World Capital` dataset show that our method can not only reliably and consistently increase the average internal memory usage rate of eight LMs by **44.0%** (from 49.7% to 93.7%) but also increase the external context usage rate by **38.5%** (from 50.3% to 88.8%). PH3 also enables LMs to generate answers more faithfully according to retrieved passages in open-domain QA tasks. We conduct extensive experiments to demonstrate the cross-model (*e.g.*, from GPT series to LLaMA2 series), cross-relation (*e.g.*, from `World Capital` to `Official Language`), and cross-format (*e.g.*, from triple format to document format) generalization. Our contributions are summarized as follows:

- We perform an exploration into the mechanism of interpreting knowledge conflicts, and reveal that memory heads and context heads at later layers can cause knowledge conflicts when inconsistent information flows merge.
- We propose a novel method called **P**runing **H**ead via **P**at**H** **P**atc**H**ing (**PH3**), which can efficiently mitigate knowledge conflicts by pruning those conflicting attention heads.
- We demonstrate that our PH3 can flexibly control LMs to use internal memory (↑ 44.0%) or external context (↑ 38.5%). We also prove the cross-model, cross-relation, and cross-format generalization ability of our method.

2 Background

In this work, we mainly focus on the autoregressive transformer-based language models. Given a sequence of input tokens $x = [x_1, \cdots, x_N]$, the LM $\mathcal{G}$ first embeds each token x_i into a vector $\mathbf{x}_i^0 \in \mathbb{R}^d$ using an embedding matrix $E \in \mathbb{R}^{|\mathcal{V}| \times d}$, over a vocabulary $\mathcal{V}$. The input embeddings are processed by L transformer layers. Each layer consists of an MHA and an FFN. Formally, the hidden state $\mathbf{x}_i^\ell$ of token x_i at layer ℓ is calculated as:

$$\mathbf{x}_i^\ell = \mathbf{x}_i^{\ell-1} + \mathbf{a}_i^\ell + \mathbf{m}_i^\ell, \tag{1}$$

where $\mathbf{a}_i^\ell$ and $\mathbf{m}_i^\ell$ are the outputs from the MHA block and the FFN block in the ℓ-th layer. Then, the vocabulary head $\phi(\cdot)$ and the softmax function $\sigma(\cdot)$ predict the output probability:

$$\mathbf{p}_i^L = \sigma\left(\phi\left(\mathbf{x}_i^L\right)\right). \tag{2}$$

MHA. A MHA block consists of M attention heads, which are capable of aggregating global information from the hidden states (Halawi et al., 2023; Wang et al., 2023b). An individual attention head h in layer ℓ consists of three learnable matrices, $\mathbf{W}_Q^{\ell,h}, \mathbf{W}_K^{\ell,h}, \mathbf{W}_V^{\ell,h} \in \mathbb{R}^{d \times \frac{d}{M}}$. Formally, for the input $\mathbf{X}^{\ell-1} = \left[\mathbf{x}_1^{\ell-1}, \cdots, \mathbf{x}_N^{\ell-1}\right]$ in layer ℓ:

$$\mathbf{A}^\ell = \left[\mathbf{H}^{\ell,1}; \cdots; \mathbf{H}^{\ell,M}\right] \mathbf{W}_o^\ell, \tag{3}$$

$$\mathbf{H}^{\ell,h} = \mathbf{s}^{\ell,h} \mathbf{X}^{\ell-1} \mathbf{W}_V^{\ell,j}, \tag{4}$$

$$\mathbf{s}^{\ell,h} = \sigma\left(\frac{\left(\mathbf{X}^{\ell-1}\mathbf{W}_Q^{\ell,h}\right)\left(\mathbf{X}^{\ell-1}\mathbf{W}_K^{\ell,h}\right)^T}{\sqrt{d/M}}\right) \tag{5}$$

where $\mathbf{A}^\ell = \left[\mathbf{a}_1^\ell, \cdots, \mathbf{a}_N^\ell\right]$ is the MHA block's output. $\mathbf{W}_O^{\ell,h} \in \mathbb{R}^{d \times d}$ is a learnable output matrix.

FFN. A FFN block can work as a key-value memory to store factual knowledge (Geva et al., 2021), enriching the hidden states of token i:

$$\mathbf{m}_i^\ell = f\left(\left(\mathbf{x}_i^{\ell-1} + \mathbf{a}_i^\ell\right)\mathbf{W}_1^\ell\right)\mathbf{W}_2^\ell. \tag{6}$$

3 Experimental Setup

3.1 Tasks

In this paper, we conduct controlled experiments to construct knowledge conflicts, wherein the internal memory is factual while the external context is counterfactual. To avoid the LM being influenced by other irrelevant factors (*i.e.*, reasoning ability), we adopt a simple factual recall task (Geva et al., 2023), which requires predicting the corresponding attribute a_m based on the given subject s and relation r. Building on previous work (Yu et al., 2023), we use the `World Capital` dataset to interpret this problem in §4, where the LM needs to predict the capital city of the country based on the question q:

Q: What is the capital of {s}? A:

We retain those questions that the LM can correctly predict the factual attributes a_m based on internal memory, then provide the counterfactual attributes a_c in the external context c to construct conflicts:

The capital of {s} is {a_c}. {q}

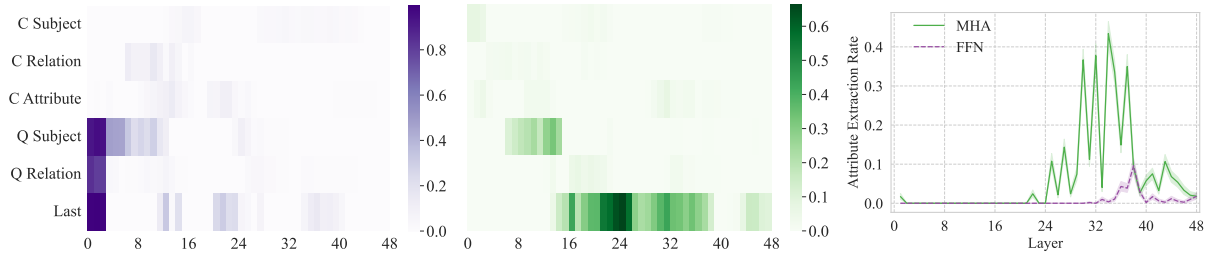
To mitigate knowledge conflicts, we further construct three datasets for verifying the generalization of our method in §5, including the `Official Language`, `Country`, and `Continent` datasets. We also generate a more complex `World Capital D` dataset based on the `World Capital` dataset, using `gpt-3.5-turbo` to rewrite the external context from triplet form into document form. More details about these datasets are shown in Appendix B.

3.2 Models

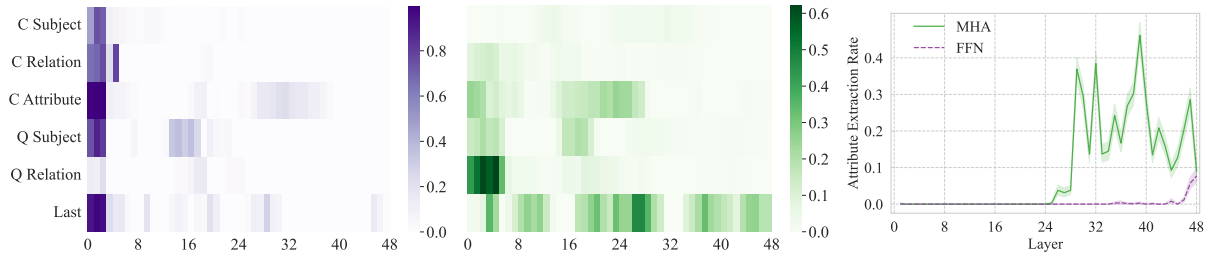
We analyze two GPT-series LMs: GPT-2 XL (Radford et al., 2019) and GPT-J (Wang and Komatsuzaki, 2021) in §4. Additionally, we also validate the effectiveness of our method on six LMs: OPT-1.3B, OPT-2.7B (Zhang et al., 2022), Pythia-6.9B, Pythia-12B (Biderman et al., 2023), LLaMA2-7B and LLaMA2-13B (Touvron et al., 2023) in §5.

4 Interpreting Knowledge Conflicts

We utilize a "*top-down*" analysis approach to locate the pivotal point where conflicts emerge and to identify the model components that are significant in knowledge conflicts. We start by examining the functionality of model components by knocking out activations, and reveal that MHAs in the middle and late layers play a crucial role in passing information to the last token (§4.1). Then, we further investigate MHAs by knocking out the attention weights. We find the question information is first passed to the last token, then the last token extracts information from the subject and the attribute in the context (§4.2). Last, we discover that some attention heads in later MHAs play opposite roles in conflicts, where memory heads can recall knowledge from internal memory, and context heads can retrieve knowledge from external context (§4.3).



(a) Effect of FFNs on internal memory. (b) Effect of MHAs on internal memory. (c) Extraction rate of internal memory.



(d) Effect of FFNs on external context. (e) Effect of MHAs on external context. (f) Extraction rate of external context.

Figure 2: Effect of model components (FFNs and MHAs) in GPT-2 XL on the final prediction probability. Figures 2a and 2b (Figures 2d and 2e) show the effect of different model components and input elements when the model predicts based on internal memory (external context). The deeper color indicates the greater the impact of knocking out this part on the original prediction probability. Figure 2c (Figure 2f) shows the effect of MHAs and FFNs on the last token's attribute extraction rate when the model predicts based on internal memory (external context).

4.1 Examining Component Functionality

We start by exploring the functionality of model components (including FFNs and MHAs across various layers) in knowledge conflicts.

Experiment 1: Knocking Out Component. We examine which component in the transformer layer is critical for the attribute prediction by knocking out activations. Then, we divide the input into six elements for analysis: context subject s_c, context relation r_c, context attribute a_c, question subject s_q, question relation r_q, and the last token x_N. To measure the impact on the final prediction results, we zero-out the updates to the specified input element from the MHA and FFN blocks within each layer. For example, to intervene in the update of the ℓ-th MHA (FFN) to the input element s_c, we set $\mathbf{a}_i^{\ell'} = \mathbf{0}$ ($\mathbf{m}_i^{\ell'} = \mathbf{0}$) for i in the token range of s_c and $\ell' = \max(1, \ell - W/2), \cdots, \min(L, \ell + W/2)$, where W denotes the window size. We define the effect of a model component as the change in the original prediction probability after knocking it out.

Results. Figure 2 illustrates the effect of model components (FFNs and MHAs) in GPT-2 XL with the window size $W = 5$. Our observation reveals that destroying the FFN blocks in the early layers has a significant effect on the prediction probability while destroying the FFN blocks at the late layers shows minimal or no impact (Figures 2a and 2d). Moreover, the MHA blocks at the middle and late layers are crucial for the last token (Figures 2b and 2e). A possible explanation of the model's behavior on the factual recall task is that *the early FFNs first enrich the semantic information of input elements, and then the enriched semantic information about attributes is extracted to the last token via late MHAs, where knowledge conflicts may arise at the later stage*. To verify this hypothesis, we will examine the attribute extraction function of MHAs.

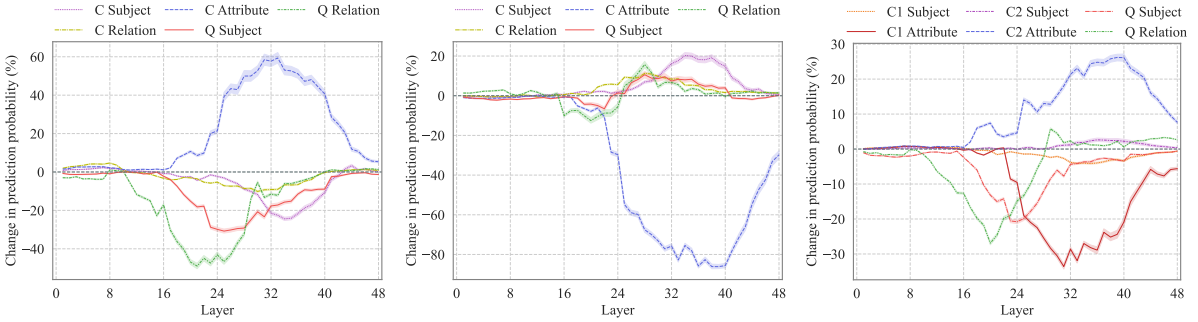
Experiment 2: Extracting Attributes via MHAs. We adopt the extraction rate (Geva et al., 2023) to examine the attribute extraction function of MHAs. We apply the early exit (Schuster et al., 2021; Geva et al., 2022) to project the MHA update $\mathbf{a}_N^\ell$ for the last token x_N over the vocabulary. Then we check whether the top token t^ℓ of each update aligns with the attribute t^* predicted at the final layer L:

$$t^* = \arg\max\left(\mathbf{p}_N^L\right), \tag{7}$$

$$t^\ell = \arg\max\left(\sigma\left(\phi\left(\mathbf{a}_N^\ell\right)\right)\right). \tag{8}$$

We consider that the MHA correctly performs attribute extraction when $t^* = t^\ell$. For comparison, we also examine the extraction rate of FFNs.

Results. As illustrated in Figures 2c and 2f, it is evident that *the attribute extraction rate of MHAs significantly exceeds that of FFNs*. Moreover, attribute extraction mainly takes place at the 24-48 layers. Results for GPT-J show similar trends in Appendix C. The above findings motivate us to conduct an in-depth study on the information flows of MHAs from input elements to the last token.



(a) Prediction based on internal memory. (b) Prediction based on external context. (c) Prediction based on internal memory.

Figure 3: Relative change in the prediction probability when blocking the information flow from the input elements to the last token. Figures 3a and 3b only provide conflicting context. Figure 3c provides both supporting and conflicting context to internal memory, C1 denotes the supporting context, and C2 denotes the conflicting context.

4.2 Tracing Information Flow

The analysis presented above confirms that the last token extracts attribute information for prediction through MHA blocks. Following this, we explore the order and importance of the information flow from the various elements to the last token.

Experiment 3: Blocking Information Flow. We localize the information propagation from the input elements (including s_c, r_c, a_c, s_q and r_q) to the last token by knocking out attention edges between them. For example, to block the information flow from the input element s_c to the last token x_N in the layer ℓ, we set the attention weight $s^{\ell,h}[N,i] = 0$ for i in the token range of s_c, $h = 1, \cdots, M$, and $\ell' = \max(1, \ell - W/2), \cdots, \min(L, \ell + W/2)$. In this way, we can restrict the last token from attending to the target element. If blocking the information propagation between them has a significant impact on the original prediction probability, this indicates that it is a crucial information flow.

Results. Figure 3 illustrates the information flow in GPT-2 XL with the window size $W = 9$. We can observe that in the early to middle layers, blocking the attention to the question relation leads to a decrease in the prediction probability. Similarly, in the subsequent layers, blocking the attention to the question subject also results in a decrease in the prediction probability. This suggests that the critical relation and subject information in the question are sequentially transmitted to the last token.

Then, in the middle to late layers, blocking the attention to the context subject and context attribute has the opposite effect on the final prediction probability. Taking Figure 3a as an example (when the model predicts the attribute based on internal memory), blocking the attention to the context attribute can improve the prediction probability, however, blocking the attention to the context subject can reduce the prediction probability. This suggests that the last token can extract the internal knowledge from the context subject, and extract the external knowledge from the context attribute. In addition, the last token also extracts a certain degree of internal knowledge from the question subject. Results for GPT-J show consistent trends in Appendix C.

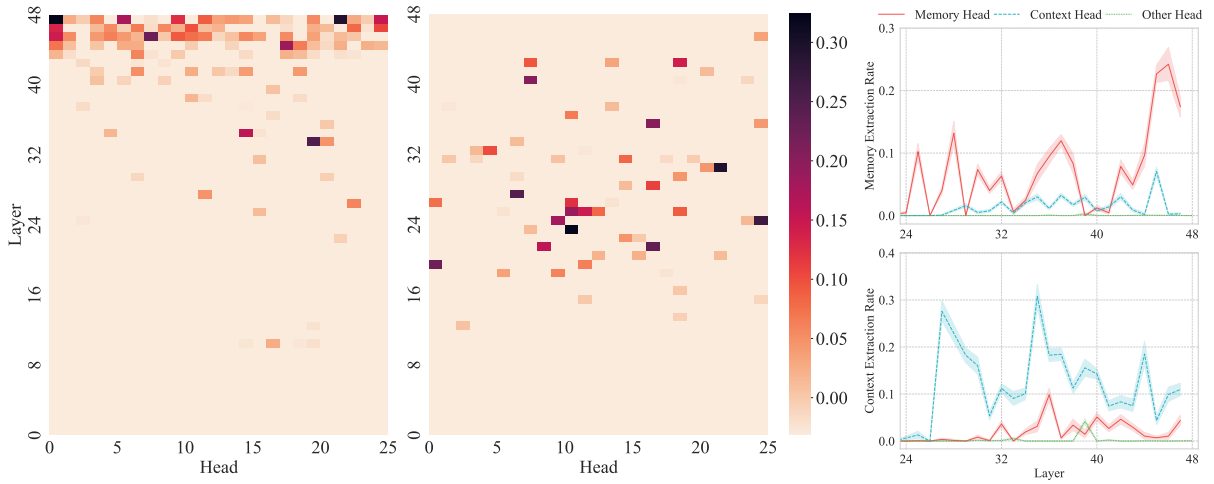
Overall, this shows that there are two specific stages in the process of information flow passing to the last token: (1) *the question information is first passed to the last token*; (2) *the last token extracts or copies the attribute from the context subject or the context attribute*. In the later stage, knowledge conflicts arise during the process of merging inconsistent information flows from MHAs.

Experiment 4: Extending to Conflicts between Contexts. We extend our analysis to a more complex scenario in which the model is presented with both supporting context and conflicting context relative to internal memory. Supporting context and conflicting context contain a_m and a_c respectively:

C1: The capital of $\{s\}$ *is* $\{a_m\}$.
C2: The capital of $\{s\}$ *is* $\{a_c\}$. $\{q\}$

We find that GPT-2 XL prefers to choose attributes consistent with internal memory 97.6% of the time. Hence, we only analyze the cases where the model makes predictions based on its internal memory.

Results. As illustrated in Figure 3c, we can observe that the question information is first passed to the last token in the first stage, which is consistent with the trend of a single conflicting context. In the second stage, a notable distinction is that *the model no longer extracts the memory attribute from the subject; instead, it opts for a more straightforward approach of copying the memory attribute from the context*. The above findings indicate that there exists a mechanism within MHAs capable of distinguishing and selecting between internal knowledge



(a) Memory head. (b) Context head. (c) Extraction rate of attention heads.

Figure 4: Memory heads and context heads in GPT-2 XL. Figure 4a shows the important score heatmap for predicting based on internal memory. Figure 4b shows the important score heatmap for predicting based on external context. Figure 4c illustrates the memory and context attribute extraction rate of different attention heads.

and external knowledge. This motivates us to conduct further analysis of MHAs.

4.3 Looking Deeper into Attention Heads

Attention heads serve as the fundamental component of an MHA block. For example, GPT-2 XL contains a total of 1,200 attention heads. This motivates us to conduct an investigation into the role of attention heads in handling knowledge conflicts.

Experiment 5: Discovering Important Heads. To discover the attention heads that are crucial for predicting memory attributes or context attributes, we compute the gradient-based importance score (Michel et al., 2019; Bansal et al., 2023) for each head. Given a dataset $\mathcal{D}$ with a set of inputs x and outputs y, the importance score of an attention head h captures the expected sensitivity of the model to h and is computed as follows:

$$I^{l,h}(\mathcal{D}) = \mathbb{E}_{(x,y)} \left| \mathbf{H}^{l,h^T} \frac{\partial \mathcal{L}(y \mid x)}{\partial \mathbf{H}^{l,h}} \right|, \quad (9)$$

where $\mathcal{L}(\cdot)$ is the loss function of conditional autoregressive generation. The proxy score of head h for predicting internal memory is calculated as:

$$S_m^{l,h}(\mathcal{D}_m, \mathcal{D}'_m) = I^{l,h}(\mathcal{D}_m) - I^{l,h}(\mathcal{D}'_m), \quad (10)$$

where $(x, a_m) \in \mathcal{D}_m$ denotes the original outputs are memory attributes, $(x, a_c) \in \mathcal{D}'_m$ denotes replacing the original outputs with context attributes. In this way, we can also calculate the proxy score of head h for predicting external context as:

$$S_c^{l,h}(\mathcal{D}_c, \mathcal{D}'_c) = I^{l,h}(\mathcal{D}_c) - I^{l,h}(\mathcal{D}'_c). \quad (11)$$

We compute the proxy score of each head across different layers to discover important heads.

Results. As shown in Figure 4a (Figure 4b), the deeper color of the red square indicates a more significant contribution from this attention head to the model's predictions based on internal memory (external context). We can observe that there are a specific number of attention heads within middle-to-late layers that play opposite roles in predicting attributes. Accordingly, we refer to those heads that contribute to the prediction of memory attributes as **memory heads**, and those that facilitate predicting context attributes as **context heads**. Therefore, we claim that *they may serve in a mutually exclusive capacity during knowledge conflicts*. The heatmaps of GPT-J are provided in the Appendix C.

Experiment 6: Extracting Specific Attributes via Heads. We further analyze the two types of heads discovered above to verify their role in knowledge conflicts. We rank the attention heads in descending order based on their importance scores, $S_m^{l,h}$ for memory and $S_c^{l,h}$ for context, subsequently identifying the top-5% of heads as memory heads and context heads, respectively. For comparison, we also randomly choose an additional 5% of the attention heads as other heads. Then, we examine their memory extraction rate when $t_\ell = a_m$, and context extraction rate when $t_\ell = a_c$.

Results. As shown in Figure 4c, memory heads and context heads are responsible for extracting different attribute information to the last token with a significant difference between memory and context extraction rates. Therefore, we discern that *the pivotal point at which knowledge conflicts emerge in LMs is the integration of inconsistent information flows by memory heads and context heads*.

5 Mitigating Knowledge Conflicts

Building on the above insights, we propose a novel method called **P**runing **H**ead via **P**at**H** **P**atc**H**ing (**PH3**) to efficiently mitigate knowledge conflicts by intervening on attention heads without the need to update model parameters (§5.1). Then, we conduct extensive experiments to show that our method can flexibly control LMs to use internal memory or external context (§5.2). Moreover, we analyze the generalization capability of our method (§5.3).

5.1 Method

Our method consists of two stages, first identifying the important heads through path patching, then intervening on these heads via structured pruning.

Localizing Memory Heads and Context Heads via Path Patching. When we use the gradient-based method in §4.3 to estimate the importance score of the target head h, it is subject to interference from other heads. The calculated gradients may not fully reflect the contribution of the target head, but rather a mixture of the influences from other heads. Therefore, we adopt the path patching technique (Goldowsky-Dill et al., 2023; Wang et al., 2023a) to analyze the causal relationship between the head h and the output attribute (including a_m and a_c) in conflicts. To calculate the important score $S_c^{\ell,h}$ of the target head h, our path patching method consists of three steps shown in Figure 14:

1. Run on the original input $x \in \mathcal{D}_c$ to record the original activations of all heads;
2. Run on the corrupted input $\not{x}$ to record the corrupted activations of all heads, where $\not{x}$ is:

   The capital of $\{s\}$ *is* $\langle$unk$\rangle$. $\{q\}$

   where $\langle$unk$\rangle$ is the special token;
3. Run on the original input x, while keeping all the heads frozen to their activations on x, except for the target head h whose activation is set on $\not{x}$. Then measure the important score as the change of output logits.

The important score $S_c^{\ell,h}$ of head h is computed as:

$$S_c^{l,h}(\mathcal{D}_c) = \mathbb{E}_{(x)}[(\mathbb{P}_x(a_c) - \mathbb{P}_x(a_m)) - (\mathbb{P}_{\not{x}}(a_c) - \mathbb{P}_{\not{x}}(a_m))]. \quad (12)$$

We adopt similar steps to calculate the importance score $S_m^{\ell,h}$ of the target head h for memory attribute prediction in Appendix D. We also provide the importance score heatmaps of memory and context heads for various models in Appendix E, and our method can clearly distinguish between them.

Pruning Attention Heads to Mitigate Knowledge Conflicts. By ranking all the attention heads in ascending order based on the importance score $S_c^{l,h}$ ($S_m^{l,h}$), we can prune the top-k% attention heads that negatively impact the model's capability to predict context (memory) attributes, thereby enhancing the model's ability to utilize external context (internal memory). To prune a head h in layer ℓ in practice, we set $\mathbf{H}^{\ell,h}$ to be the zero matrix.

5.2 Experiment

Setups. We evaluate our method on five datasets, including `World Capital`, `World Capital D`, `Official Language`, `Country`, and `Continent`. To verify the generalization of PH3, we only calculate the importance scores of the attention heads on the `World Capital` dataset, and then directly evaluate PH3 on other datasets. We also select 1,000 test samples from an open-domain QA dataset NQ (Kwiatkowski et al., 2019), providing the LM with the top-5 retrieved passages, and ensuring that at least one relevant passage is among them. We validate the effectiveness of PH3 on eight LMs.

Metrics. We use the internal memory usage rate $RM = \frac{f_m}{f_m+f_c+f_o}$ and the external context usage rate $RC = \frac{f_c}{f_m+f_c+f_o}$ to assess how effectively the method controls the reliance of LMs on either internal memory or external context, where f_m is the frequency of relying on internal memory, f_c is the frequency of relying on external context, and f_o is the frequency of other answers. For the open-domain QA task, we use Recall to evaluate whether the model can provide correct answers based on the retrieved passages following Adlakha et al. (2023).

Baselines. We compare with the following baselines: (1) **Prompt**: We instruct the LM to generate answers based on internal memory or external context through specific prompts; (2) **CAD**: Shi et al. (2023) leverage contrastive decoding (Li et al., 2023b) to encourage the LM to attend to its context during generation; (3) **Gradient**: We replace our path patching method with the gradient-based method to discover the attention heads. We select the optimal pruning rate k on the development set for both Gradient and PH3. More details about hyperparameter settings are in Appendix B.2.

Results. Table 1 shows the results of GPT-2 XL, GPT-J and LLaMA2-7B, and more results of other models are in Table 2. Throughout our experiments, we note the following key observations:

Model	Method		World Capital		World Capital D		Official Language		Country		Continent	
			RM	RC	RM	RC	RM	RC	RM	RC	RM	RC
GPT-2 XL		Base	59.2	40.8	47.2	52.8	42.2	57.8	37.2	62.8	41.5	58.5
	↑ Memory	Prompt	12.5	81.2	21.3	71.2	20.3	74.4	24.5	75.5	16.2	43.4
		Gradient	72.4	9.8	78.6	10.5	41.5	40.5	39.1	60.2	42.7	46.6
		PH3 (Ours)	**97.9**	**0.6**	**93.3**	**2.5**	**74.4**	**9.8**	**50.9**	**36.3**	**53.1**	**38.1**
	↑ Context	Prompt	9.3	87.5	18.9	75.2	17.1	80.7	18.5	81.4	25.5	58.3
		CAD	25.0	65.6	12.5	63.6	**9.1**	80.5	27.2	72.5	22.9	60.4
		Gradient	44.4	49.0	28.0	58.7	29.6	59.5	36.4	63.4	**18.4**	51.5
		PH3 (Ours)	27.5	68.9	7.7	91.3	20.7	74.8	22.7	75.7	27.7	**66.7**
		+ Prompt	**3.6**	**95.1**	**5.2**	**94.4**	9.6	**88.7**	**12.6**	**86.0**	20.9	63.8
GPT-J		Base	37.5	62.5	43.1	56.9	41.5	58.5	54.0	46.0	43.2	56.8
	↑ Memory	Prompt	29.8	67.1	31.6	62.1	23.1	69.1	22.4	77.6	12.5	86.0
		Gradient	67.9	8.3	67.6	**6.7**	39.4	53.4	54.4	45.5	57.2	30.0
		PH3 (Ours)	**93.3**	**1.6**	**76.5**	10.8	**63.3**	**25.3**	**58.9**	**40.5**	**75.1**	**17.6**
	↑ Context	Prompt	31.9	64.5	15.8	76.4	16.2	70.6	17.9	82.1	7.2	**91.4**
		CAD	2.5	89.9	13.4	68.2	4.7	89.9	17.0	81.8	13.0	80.3
		Gradient	6.1	88.3	7.9	67.8	5.7	76.4	29.2	70.5	36.8	60.7
		PH3 (Ours)	0.2	99.3	**0.1**	**98.4**	2.3	**90.6**	9.5	86.7	8.0	64.9
		+ Prompt	**0.1**	**99.5**	0.2	97.8	**2.0**	81.9	**1.4**	**98.6**	**1.4**	90.9
LLaMA2-7B		Base	46.3	53.7	95.5	4.0	18.8	80.3	52.9	46.8	30.9	69.1
	↑ Memory	Prompt	36.0	63.2	96.0	3.7	40.0	59.1	68.2	31.6	77.4	22.6
		Gradient	81.0	5.8	95.1	1.6	50.1	47.4	60.0	38.0	64.5	24.5
		PH3 (Ours)	**98.1**	**1.2**	**98.0**	**1.3**	**73.7**	**17.8**	**76.9**	**20.6**	**90.5**	**8.8**
	↑ Context	Prompt	3.2	96.6	92.4	2.2	25.5	73.8	58.2	41.5	19.2	80.3
		CAD	1.4	95.5	29.1	70.6	**0.0**	**100.0**	13.6	86.1	0.2	98.2
		Gradient	23.6	63.2	40.1	58.8	25.7	74.6	17.6	82.2	27.1	72.9
		PH3 (Ours)	1.6	97.4	19.1	73.4	0.1	99.9	5.2	94.7	0.5	99.4
		+ Prompt	**0.4**	**98.8**	**10.6**	**85.3**	**0.0**	**100.0**	**2.8**	**97.0**	**0.0**	**100.0**

Table 1: Experimental results of GPT-2 XL, GPT-J and LLaMA2-7B on five datasets. Bolds denote the best results.

(1) PH3 significantly outperforms other baselines. Experimental results show that PH3 can not only increase the average internal memory usage rate of eight LMs by 44.0%, but also increase the average external context usage rate by 38.5%. When PH3 is combined with Prompt, it can more effectively control the LMs to use external context.

(2) As shown in Table 3, PH3 can also achieve an average 6.2% Recall improvement on open-domain QA tasks. By pruning a small number of negative context heads, PH3 can make LMs generate answers more faithfully based on retrieved passages.

(3) Although Prompt and CAD can effectively increase the external context usage rate, there are limitations. CAD cannot directly enhance internal memory, and Prompt may even have the opposite effect. In contrast, our method offers a viable solution to enhance the internal memory usage rate.

5.3 Analysis

We conduct a thorough analysis of the generalization ability of PH3. For cross-model generalization, PH3 is effective across a wide range of models. This shows that our method is not limited to small models, but can also be adopted on relatively large models, including the popular LLaMA2 series. For cross-relation generalization, by intervening on the attention heads discovered on `World Capital`, our method can also well resolve knowledge conflicts on other relation types. This indicates that PH3 does not identify attention heads specific to a certain type of relation. Instead, it identifies universal memory and context heads. For cross-format generalization, PH3 can transfer well from triple-form context to document-form context. This indicates that our method does not merely remember the relative positions of elements in context, but is capable of understanding the external context. Compared to the Gradient, our method has demonstrated superior generalizability. We also analyze the impact of the number of pruning heads in Appendix G.

6 Conclusion

In this paper, we perform an exploration into the mechanism of interpreting knowledge conflicts and reveal that memory and context heads in later layers can cause knowledge conflicts when merging inconsistent information flows. Based on our insights, we propose a novel method called **P**runing **H**ead via **P**at**H** **P**atc**H**ing (**PH3**), which can mitigate knowledge conflicts by pruning those conflicting attention heads. We prove that PH3 can flexibly control LMs to use internal memory or external context. We also demonstrate the cross-model, cross-relation, and cross-format generalization.

Limitations

For further study, we conclude some limitations of our work as follows:

- Similar to previous works on mechanism interpretability that adopt tasks such as antonym generation (Todd et al., 2023), fact recall (Meng et al., 2022; Geva et al., 2023), arithmetic operation (Hanna et al., 2023; Stolfo et al., 2023), and text classification (Bansal et al., 2023; Wang et al., 2023b), our work also selects a relatively simpler task to interpret the mechanism behind knowledge conflicts. Simple tasks enable us to better control variables and minimize external distractions. In the future, we plan to extend our analysis to more complex and realistic scenarios, such as where irrelevant information is present within the external context, or where the model needs to reason with both internal and external knowledge.
- Although our research has delved into the attention heads in LMs, there may be more basic elements involved in knowledge conflicts. Furthermore, the memory and context heads we have discovered may not only be responsible for extracting knowledge from internal memory or external context. These heads may also have other functions, such as helping the model capture global dependencies of input texts. By pruning these heads, the original capabilities of the model may be affected. Therefore, we will further explore mitigating knowledge conflicts through more subtle intervention methods.

In summary, the mechanism behind knowledge conflicts remains a largely unexplored area, and we hope our work can offer some useful insights for further research.

Ethics Statement

To enhance the reproducibility of our research, we will make all source code and datasets publicly available upon the acceptance of this paper. Our work focuses on uncovering the mechanisms behind knowledge conflicts in LM, thereby better controlling the model in retrieval augmentation and tool augmentation. Through effective intervention, our method can make the LM more controllable and trustworthy. On the one hand, it can prevent prompt injections from attacking the model, and on the other hand, it can correct the biased knowledge that the model learned during pre-training. Nonetheless, the impact of head pruning on the model's original capabilities remains unexplored. These factors should be taken into careful consideration for future research.

References

Vaibhav Adlakha, Parishad BehnamGhader, Xing Han Lu, Nicholas Meade, and Siva Reddy. 2023. Evaluating correctness and faithfulness of instruction-following models for question answering. *arXiv preprint arXiv:2307.16877*.

Hritik Bansal, Karthik Gopalakrishnan, Saket Dingliwal, Sravan Bodapati, Katrin Kirchhoff, and Dan Roth. 2023. Rethinking the role of scale for in-context learning: An interpretability-based case study at 66 billion scale. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11833–11856, Toronto, Canada. Association for Computational Linguistics.

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

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nick Cammarata, Shan Carter, Gabriel Goh, Chris Olah, Michael Petrov, Ludwig Schubert, Chelsea Voss, Ben Egan, and Swee Kiat Lim. 2020. Thread: Circuits. *Distill*. Https://distill.pub/2020/circuits.

Hung-Ting Chen, Michael Zhang, and Eunsol Choi. 2022. Rich knowledge sources bring complex knowledge conflicts: Recalibrating models to reflect conflicting evidence. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2292–2307, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in

pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1.

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*, pages 12216–12235, Singapore. Association for Computational Linguistics.

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. 2023. Localizing model behavior with path patching. *arXiv preprint arXiv:2304.05969*.

Danny Halawi, Jean-Stanislas Denain, and Jacob Steinhardt. 2023. Overthinking the truth: Understanding how language models process false demonstrations. *arXiv preprint arXiv:2307.09476*.

Michael Hanna, Ollie Liu, and Alexandre Variengien. 2023. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2021. Self-attention attribution: Interpreting information interactions inside transformer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12963–12971.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9318–9333, Singapore. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Daliang Li, Ankit Singh Rawat, Manzil Zaheer, Xin Wang, Michal Lukasik, Andreas Veit, Felix Yu, and Sanjiv Kumar. 2023a. Large language models with controllable working memory. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1774–1793, Toronto, Canada. Association for Computational Linguistics.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023b. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12286–12312, Toronto, Canada. Association for Computational Linguistics.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. 2021. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7052–7063, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Ella Neeman, Roee Aharoni, Or Honovich, Leshem Choshen, Idan Szpektor, and Omri Abend. 2023. DisentQA: Disentangling parametric and contextual knowledge with counterfactual question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10056–10070, Toronto, Canada. Association for Computational Linguistics.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

OpenAI. 2023. Gpt-4 technical report.

Cheng Qian, Xinran Zhao, and Sherry Tongshuang Wu. 2023. "merge conflicts!" exploring the impacts of external distractors to parametric knowledge graphs. *ArXiv*, abs/2309.08594.

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

Mansi Sakarvadia, Aswathy Ajith, Arham Khan, Daniel Grzenda, Nathaniel Hudson, André Bauer, Kyle Chard, and Ian Foster. 2023a. Memory injections: Correcting multi-hop reasoning failures during inference in transformer-based language models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 342–356, Singapore. Association for Computational Linguistics.

Mansi Sakarvadia, Arham Khan, Aswathy Ajith, Daniel Grzenda, Nathaniel Hudson, André Bauer, Kyle Chard, and Ian Foster. 2023b. Attention lens: A tool for mechanistically interpreting the attention head information retrieval mechanism.

Tal Schuster, Adam Fisch, Tommi Jaakkola, and Regina Barzilay. 2021. Consistent accelerated inference via confident adaptive transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4962–4979, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Scott Wen-tau Yih. 2023. Trusting your evidence: Hallucinate less with context-aware decoding. *arXiv preprint arXiv:2305.14739*.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. 2023. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7035–7052, Singapore. Association for Computational Linguistics.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. 2023. Function vectors in large language models.

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

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, volume 33, pages 12388–12401. Curran Associates, Inc.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023a. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023b. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9840–9855, Singapore. Association for Computational Linguistics.

Yike Wang, Shangbin Feng, Heng Wang, Weijia Shi, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. 2023c. Resolving knowledge conflicts in large language models. *arXiv preprint arXiv:2310.00935*.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. 2023. Adaptive chameleon or stubborn sloth: Unraveling the behavior of large language models in knowledge conflicts. *arXiv preprint arXiv:2305.13300*.

Yi Yang, Hanyu Duan, Ahmed Abbasi, John P. Lalor, and Kar Yan Tam. 2023. Bias a-head? analyzing bias in transformer-based language model attention heads.

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*, pages 9924–9959, Singapore. Association for Computational Linguistics.

Fred Zhang and Neel Nanda. 2024. Towards best practices of activation patching in language models: Metrics and methods.

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.

Wenxuan Zhou, Sheng Zhang, Hoifung Poon, and Muhao Chen. 2023. Context-faithful prompting for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14544–14556, Singapore. Association for Computational Linguistics.

A Related Work

A.1 Investigating Knowledge Conflict

Previous research (Longpre et al., 2021; Chen et al., 2022; Yu et al., 2023; Xie et al., 2023; Wang et al., 2023c; Neeman et al., 2023) on knowledge conflicts primarily seek to answer the question: *do language models prefer internal memory or external context?* Yu et al. (2023) find that language models are more inclined to internal memory as the frequency of a fact in the pre-training corpus increases. Xie et al. (2023) demonstrate that large language models (LLMs) are highly receptive to external conflicting evidence. They also reveal that when both supportive and contradictory evidence to their internal memory are present, LLMs show a strong confirmation bias and tend to cling to their parametric memory. The above observed phenomena contribute to a better understanding of knowledge conflicts. However, the underlying mechanism of knowledge conflicts remains unclear. We observe that knowledge conflicts arise when the late attention heads integrate different information flows from internal memory and external context.

A.2 Resolving Knowledge Conflict

Existing work (Shi et al., 2023; Zhou et al., 2023; Li et al., 2023a; Yu et al., 2023; Qian et al., 2023) has conducted preliminary exploration into the mitigation of knowledge conflicts. Shi et al. (2023) propose a simple method to encourage the LM to attend to the external context via contrastive decoding (Li et al., 2023b). Yu et al. (2023) use head attribution to identify individual attention heads that either promote the memorized answer or the in-context answer, then scale the value vector of these heads to increase the rate of the in-context answers. Our work is inspired by their exploration of attention heads, and we propose further analysis to improve understanding of the way knowledge conflicts are formed. Furthermore, while most existing methods (Shi et al., 2023; Yu et al., 2023) primarily focus on improving the model's faithfulness to the context, enabling the model to adhere to its internal memory remains a challenging task.

A.3 Mechanistic Interpretability

Recently, there has been a growing interest in the mechanistic interpretability (Cammarata et al., 2020; Elhage et al., 2021) of parametric knowledge in LMs, with efforts focusing on reverse engineering the computational processes of model parameters. Dai et al. (2022) use a knowledge attribution method (Hao et al., 2021) to identify the knowledge neurons in FFNs. Meng et al. (2022) reveal that FFNs at a range of middle layers can recall facts by using the causal mediation analysis method (Vig et al., 2020). Geva et al. (2023) find that knowledge extraction is typically done via attention heads. Besides, there are some works investigating LMs in mathematical reasoning (Hanna et al., 2023; Stolfo et al., 2023) and in-context learning (Hendel et al., 2023; Olsson et al., 2022; Bansal et al., 2023). Besides, there are some studies (Yang et al., 2023; Sakarvadia et al., 2023a,b; Zhang and Nanda, 2024) focused on interpreting attention heads in LMs. Our work is highly inspired by previous wisdom in mechanistic interpretability, focusing on interpreting and mitigating knowledge conflicts in LMs.

B Implement Details

B.1 Datasets

We construct `Official Language`, `Country`, and `Continent` datasets by sampling knowledge triples from Wikidata. The `Official Language` dataset requires the LM to predict the official language of the given city or country:

> *The official language of* $\{s\}$ *is* $\{a_c\}$.
> *Q: What is the official language of* $\{s\}$*? A:*

The `Country` dataset requires the LM to predict the country to which the given city belongs:

> *The city* $\{s\}$ *is located in* $\{a_c\}$.
> *Q: Which country is the city* $\{s\}$ *in ? A:*

The `Continent` dataset requires the LM to predict the continent on which the given country is located:

> $\{s\}$ *is in the continent of* $\{a_c\}$.
> *Q: Which continent is* $\{s\}$ *located in ? A:*

We also generate a more complex `World Capital D` dataset based on the `World Capital` dataset, using `gpt-3.5-turbo` to rewrite the external context from triplet form into document form.

B.2 Hyperparameter Settings

Our implementation is based on HuggingFace's Transformers[1], PyTorch[2] and baukit[3]. For the Prompt method, we use the following prompt to enhance the internal memory:

> *Please answer the question based on your internal memory, ignoring the given context.*

[1] https://github.com/huggingface/transformers/
[2] https://github.com/pytorch/pytorch/
[3] https://github.com/davidbau/baukit/

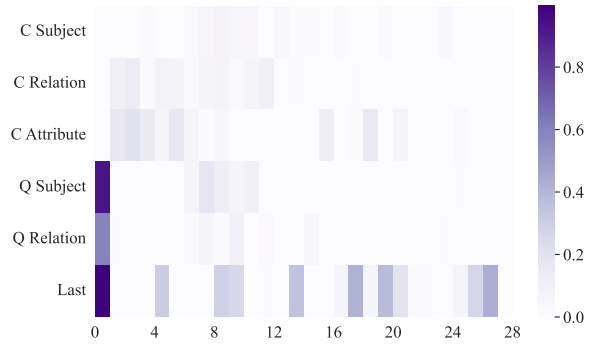


Figure 5: Effect of FFNs in GPT-J on internal memory.

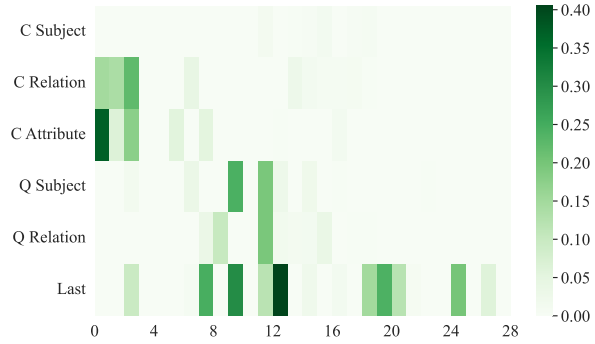


Figure 6: Effect of MHAs in GPT-J on internal memory.

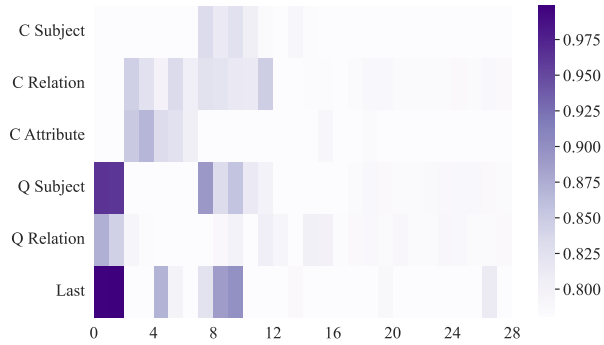


Figure 7: Effect of FFNs in GPT-J on external context.

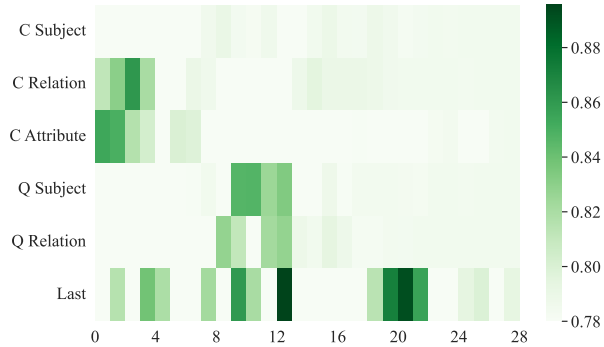


Figure 8: Effect of MHAs in GPT-J on external context.

and we use the following prompt to enhance the external context:

> *Please answer the question based on the given context, ignoring your internal memory.*

For Gradient and PH3, we select the optimal pruning rate $k \in \{1, 3, 5, 7, 9, 15\}$ on the development set with 200 samples. To mitigate knowledge conflicts, setting the pruning rate k of PH3 to 5 usually achieves excellent results. For enhancing the open-domain QA capabilities, we usually set the pruning rate k of PH3 to 3. Details about the models used in this paper are in Table 4. All experiments are conducted with NVIDIA GeForce RTX A6000 GPUs.

C Additional Results for GPT-J

We provide here additional results for GPT-J. Figures 5 and 6 show the effect of FFNs and MHAs on internal memory, and Figures 7 and 8 show the effect of FFNs and MHAs on external context. Figures 9 and 10 illustrate the information flow in GPT-J with the window size $W = 9$. Figure 11 shows the information flow in GPT-J when providing both supporting context and conflicting context relative to internal memory. Figures 12 and 13 show the gradient-based important scores of memory heads and context heads in GPT-J.

D Method Details

To calculate the important score $S_m^{\ell,h}$ of the target head h, our path patching method consists of the following three steps:

1. Run on the original input $x \in \mathcal{D}_m$ to record the original activations of all heads;
2. Run on the corrupted input $\not{x}$ to record the corrupted activations of all heads, where $\not{x}$ is:

   > *The capital of* ⟨unk⟩ *is* $\{a_c\}$.
   > *Q: What is the capital of* ⟨unk⟩ *? A:*

   where ⟨unk⟩ is the special token;
3. Run on the original input x, while keeping all the heads frozen to their activations on x, except for the target head h whose activation is set on $\not{x}$. Then measure the important score as the change of output logits.

The important score $S_m^{\ell,h}$ of head h is computed as:

$$S_m^{l,h}(\mathcal{D}_m) = \mathbb{E}_{(x)}[(\mathbb{P}_x(a_m) - \mathbb{P}_x(a_c)) - (\mathbb{P}_{\not{x}}(a_m) - \mathbb{P}_{\not{x}}(a_c))]. \quad (13)$$

E Heatmaps of Attention Heads

We calculate the important scores of memory heads and context heads via our path patching method,

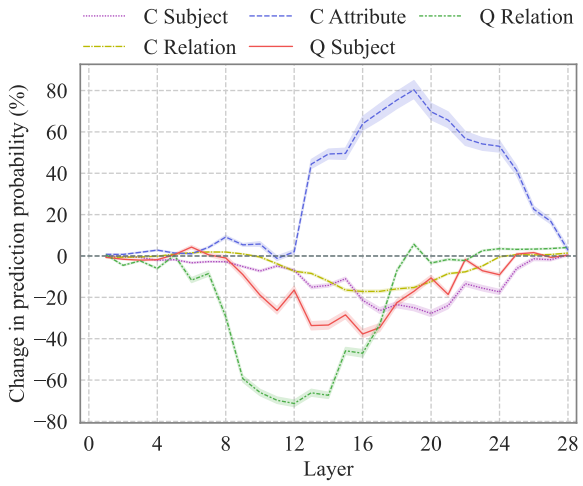


Figure 9: Relative change in the GPT-J's prediction probability based on internal memory.

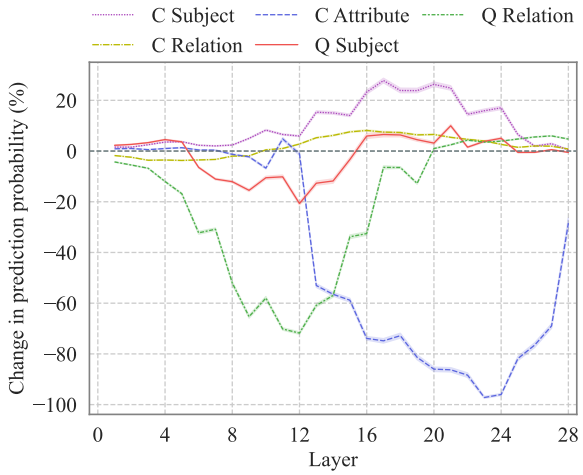


Figure 10: Relative change in the GPT-J's prediction probability based on external context.

then provide the heatmaps for GPT-2 XL (Figures 15 and 16), GPT-J (Figures 17 and 18), OPT-1.3B (Figures 19 and 20), OPT-2.7B (Figures 21 and 22), Pythia-6.9B (Figures 23 and 24), Pythia-12B (Figures 25 and 26), LLaMA2-7B (Figures 27 and 28) and LLaMA2-13B (Figures 29 and 30). The red squares indicate heads that have a significant positive impact, while the blue squares represent heads that have a negative effect.

F Additional Experimental Results

We report experimental results in Table 2 and 3.

G Number of Pruning Heads

As shown in Figures 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41 and 42, we analyze the impact of the number of pruning heads (sparsity ratio) on the Gradient and PH3 methods.

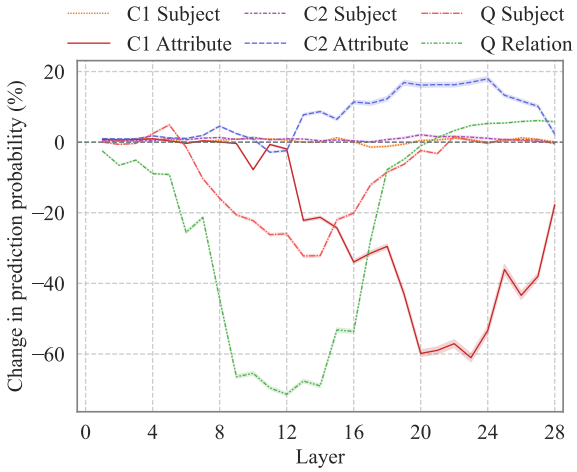


Figure 11: Relative change in the GPT-J's prediction probability based on internal memory when providing both supporting context and conflicting context.

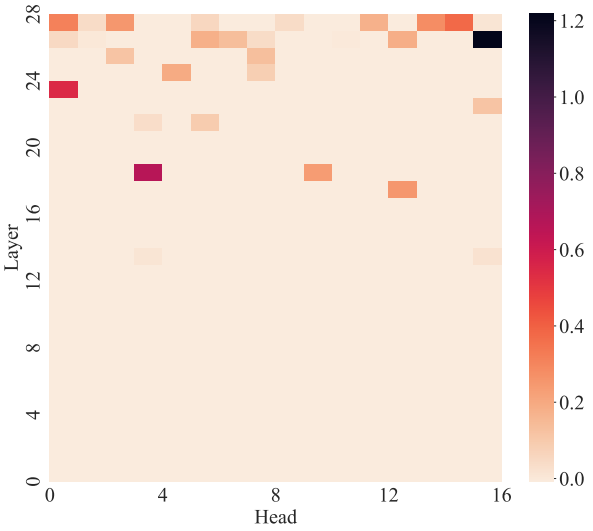


Figure 12: Memory Heads of GPT-J.

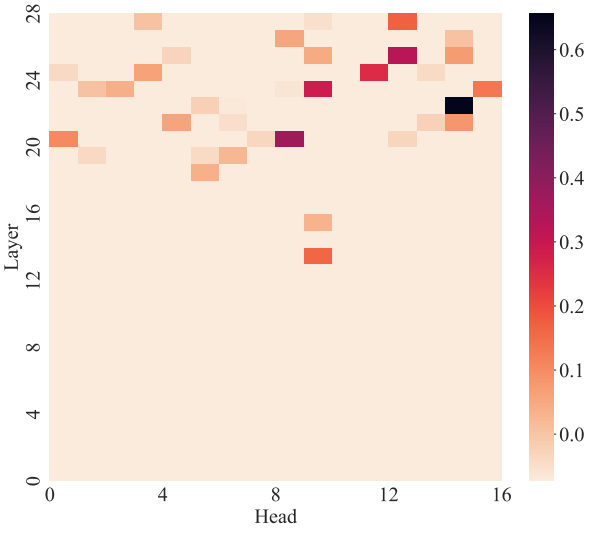


Figure 13: Context Heads of GPT-J.

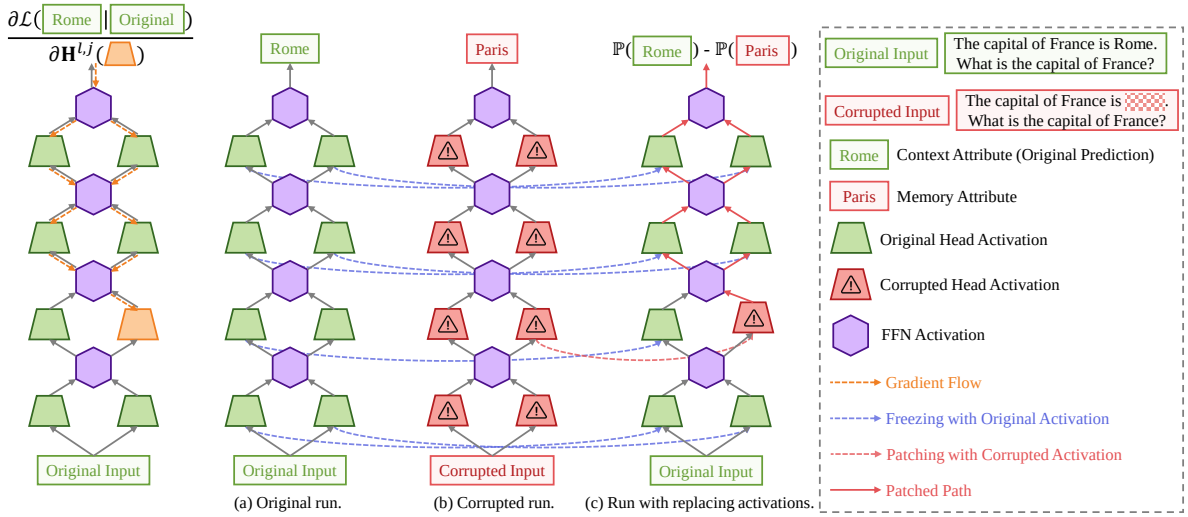


Figure 14: Illustration of gradient-based method and our path patching method.

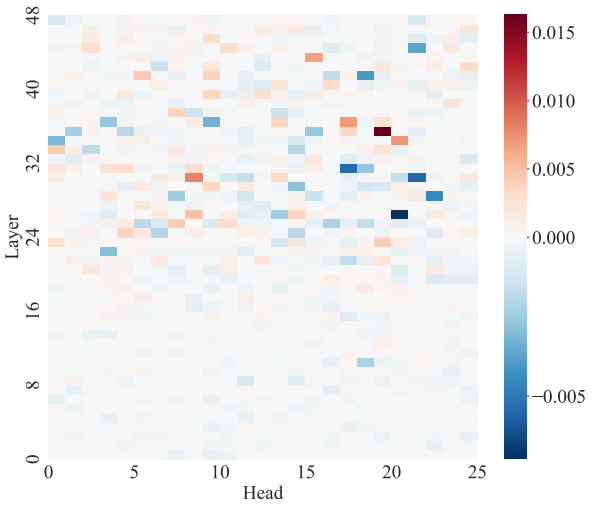


Figure 15: Memory Heads of GPT-2 XL.

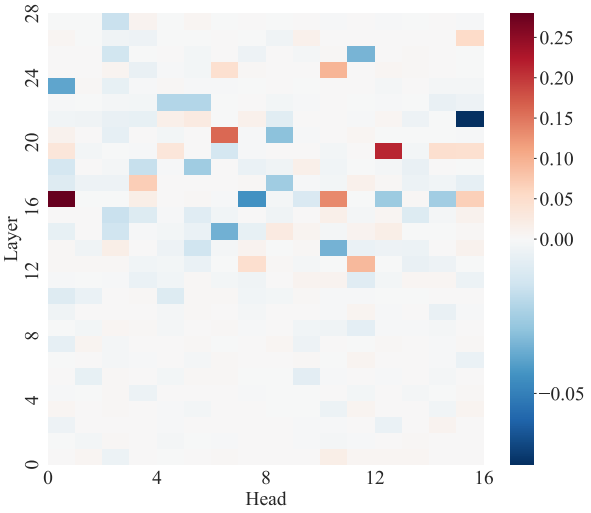


Figure 17: Memory Heads of GPT-J.

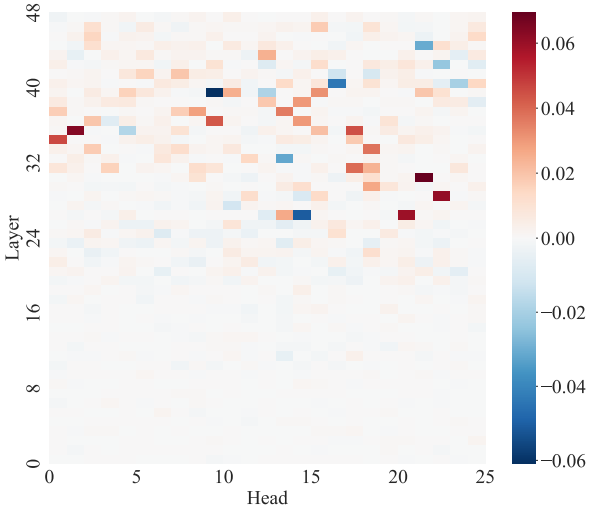


Figure 16: Context Heads of GPT-2 XL.

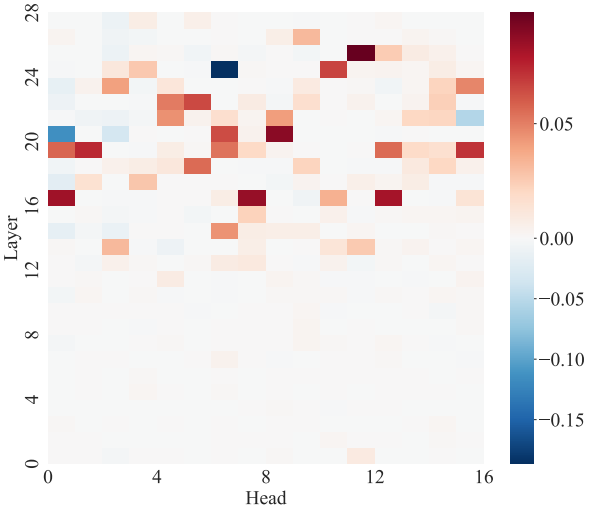


Figure 18: Context Heads of GPT-J.

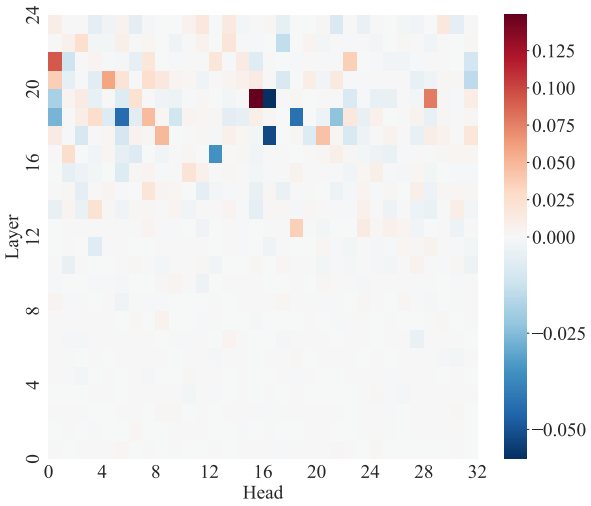


Figure 19: Memory Heads of OPT-1.3B.

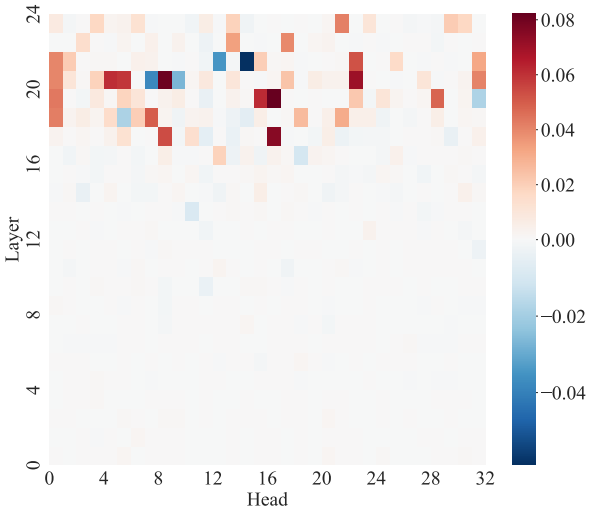


Figure 20: Context Heads of OPT-1.3B.

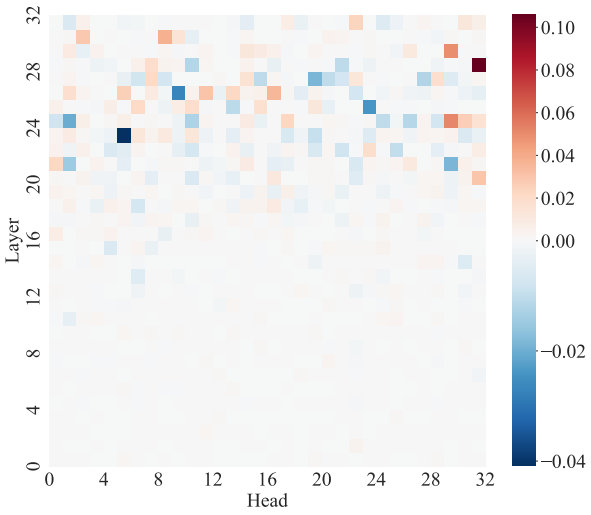


Figure 21: Memory Heads of OPT-2.7B.

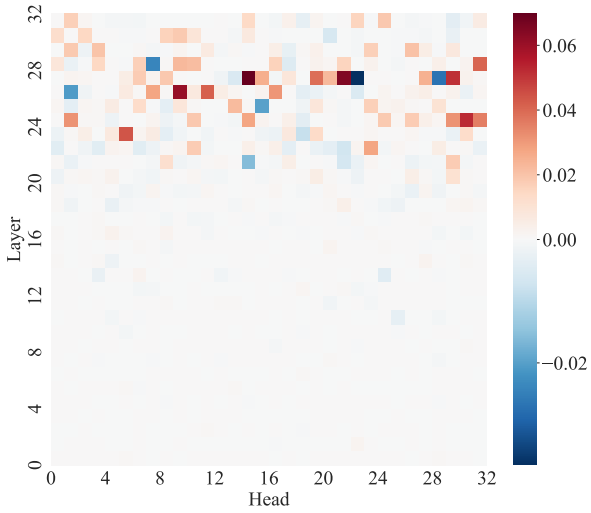


Figure 22: Context Heads of OPT-2.7B.

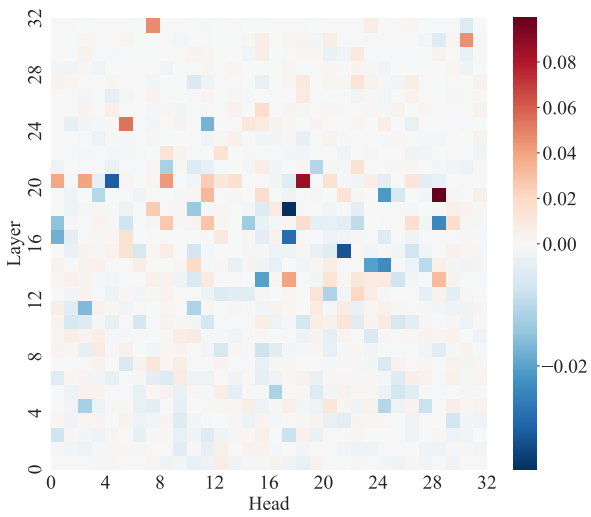


Figure 23: Memory Heads of Pythia-6.9B.

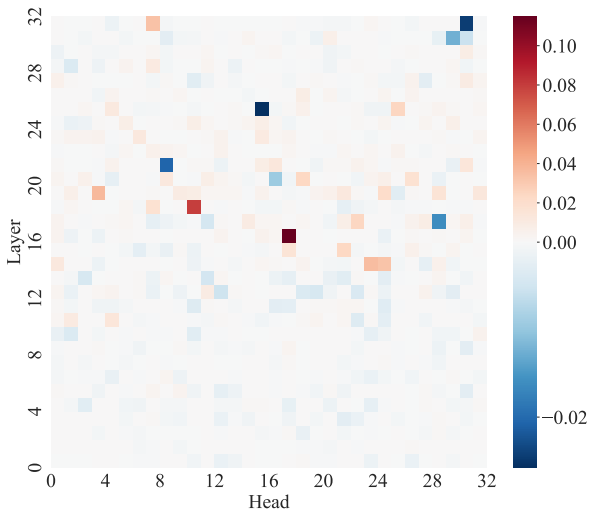


Figure 24: Context Heads of Pythia-6.9B.

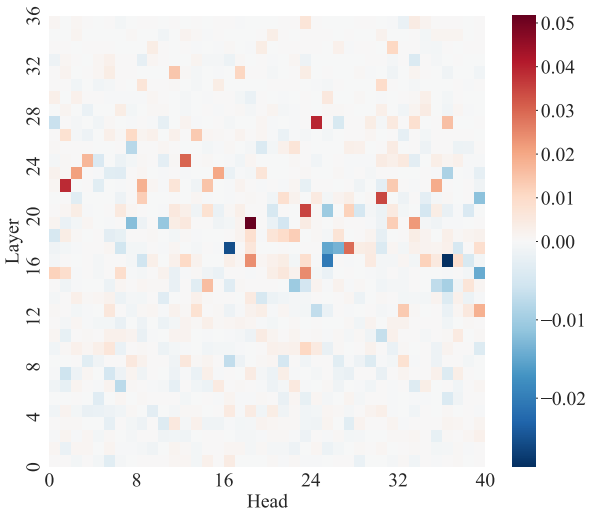


Figure 25: Memory Heads of Pythia-12B.

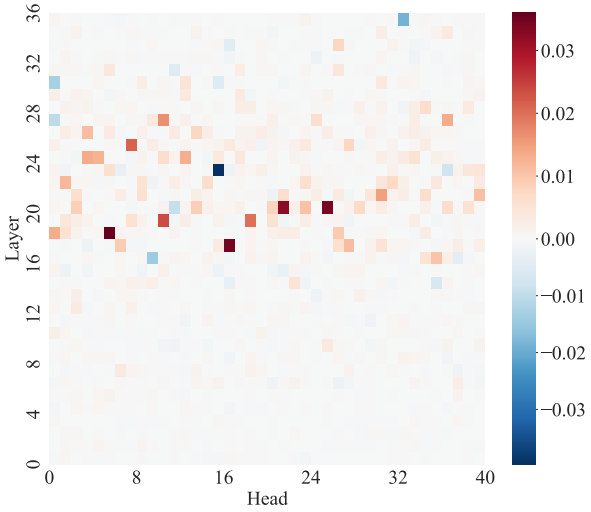


Figure 26: Context Heads of Pythia-12B.

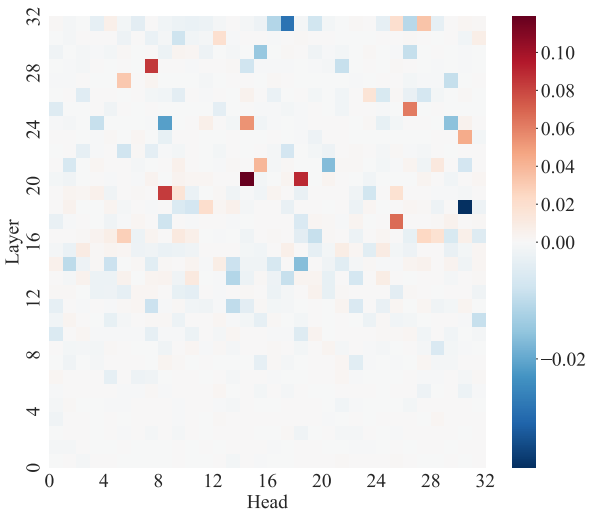


Figure 27: Memory Heads of LLaMA2-7B.

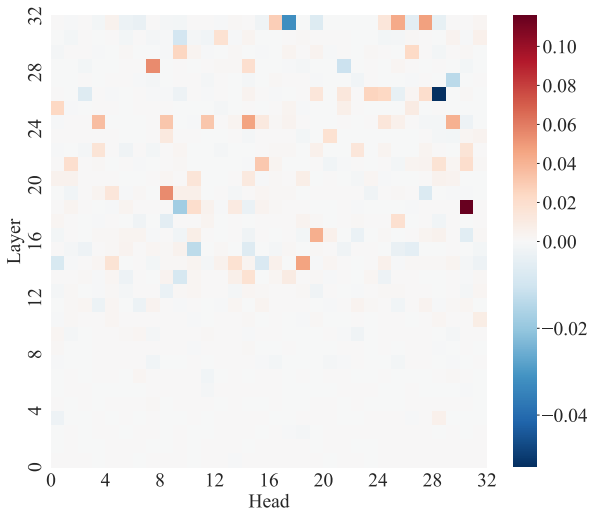


Figure 28: Context Heads of LLaMA2-7B.

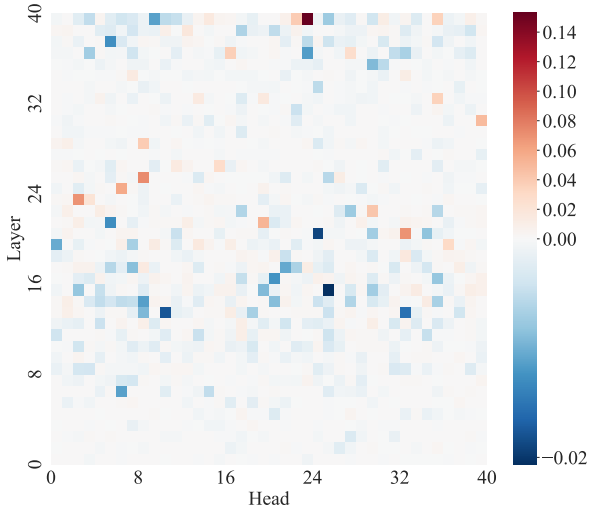


Figure 29: Memory Heads of LLaMA2-13B.

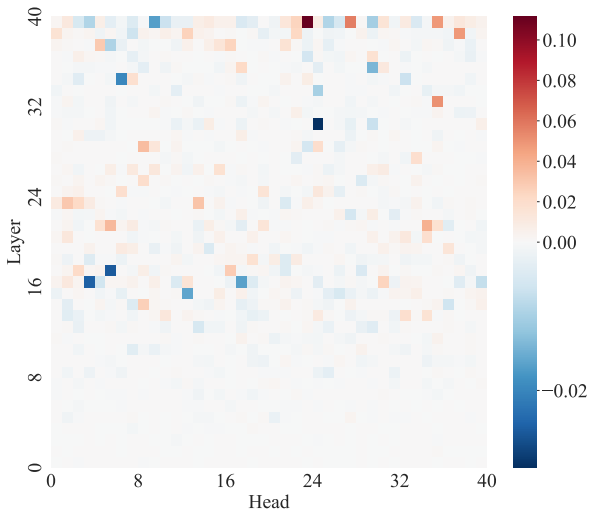


Figure 30: Context Heads of LLaMA2-13B.

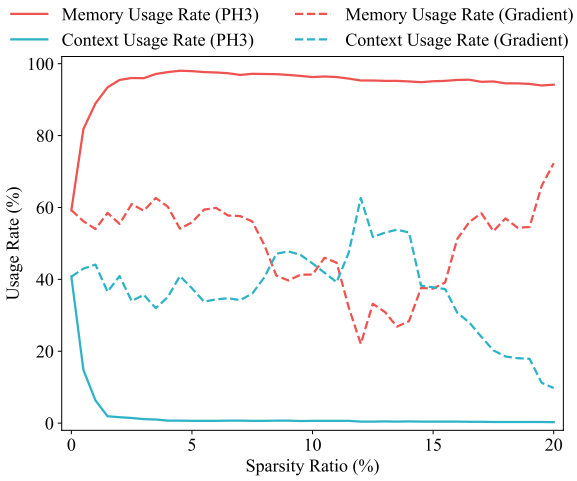


Figure 31: Impact of GPT-2 XL's sparsity ratio on improving internal memory usage rate.

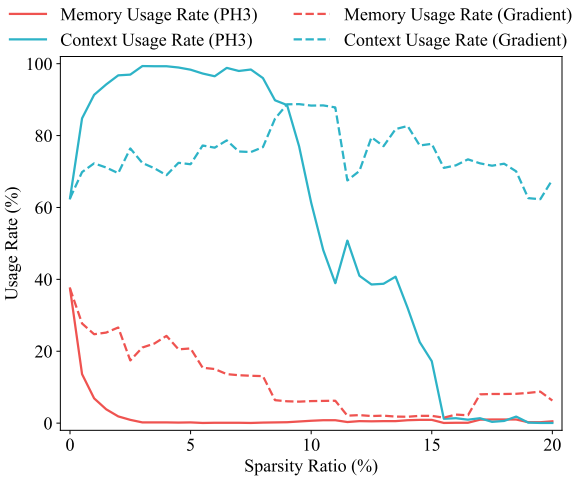


Figure 34: Impact of GPT-J's sparsity ratio on improving external context usage rate.

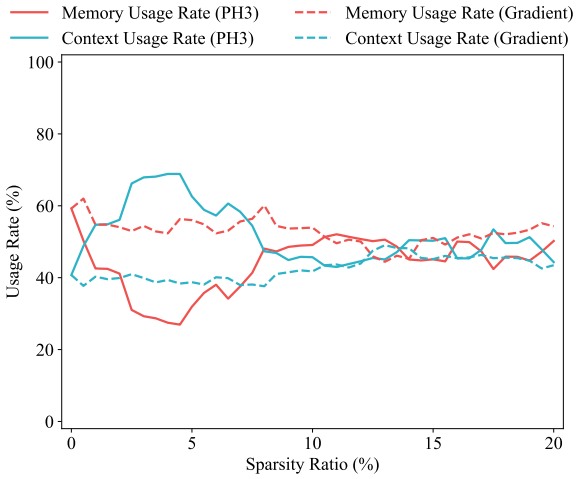


Figure 32: Impact of GPT-2 XL's sparsity ratio on improving external context usage rate.

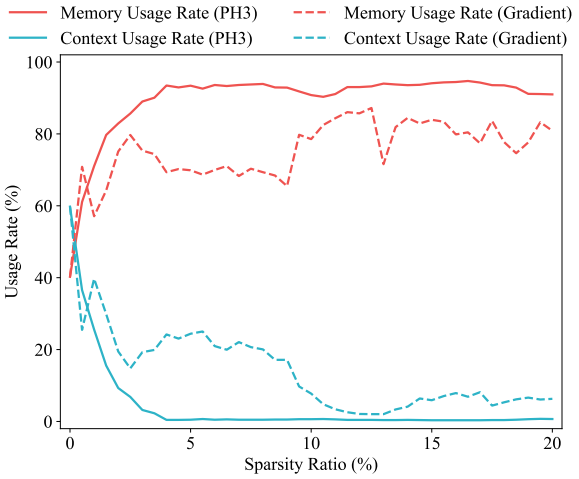


Figure 35: Impact of OPT-2.7B's sparsity ratio on improving internal memory usage rate.

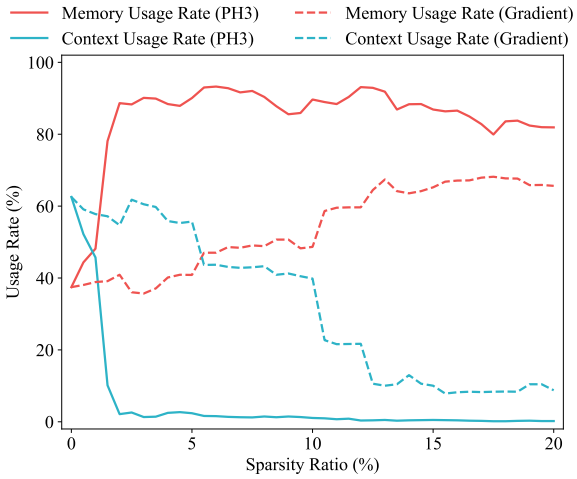


Figure 33: Impact of GPT-J's sparsity ratio on improving internal memory usage rate.

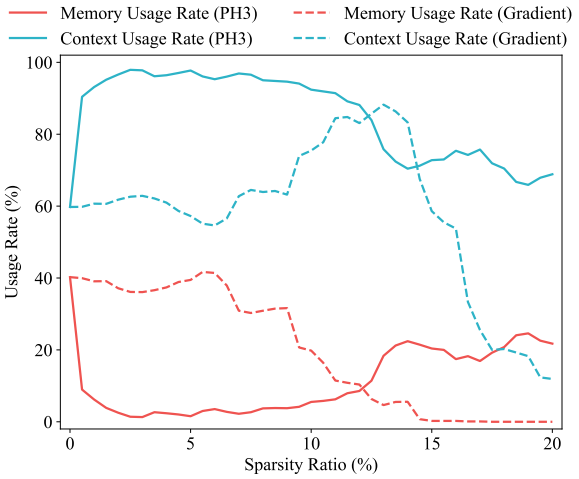


Figure 36: Impact of OPT-2.7B's sparsity ratio on improving external context usage rate.

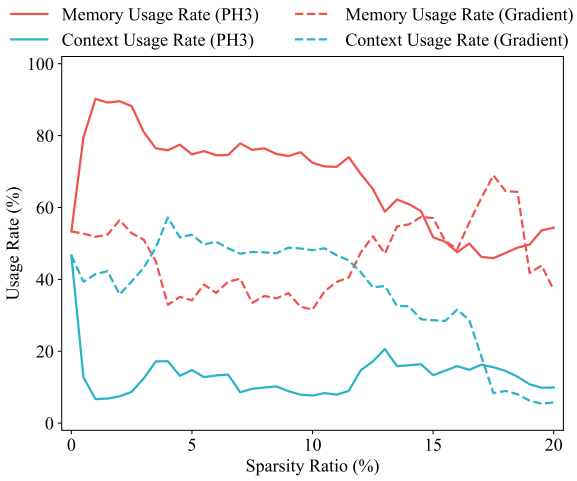


Figure 37: Impact of Pythia-6.9B's sparsity ratio on improving internal memory usage rate.

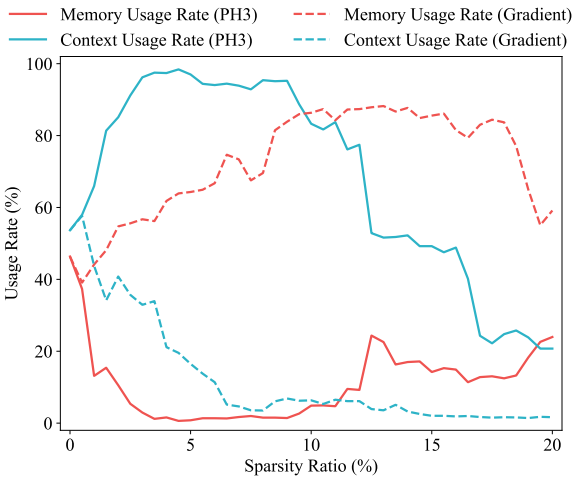


Figure 40: Impact of LLaMA2-7B's sparsity ratio on improving external context usage rate.

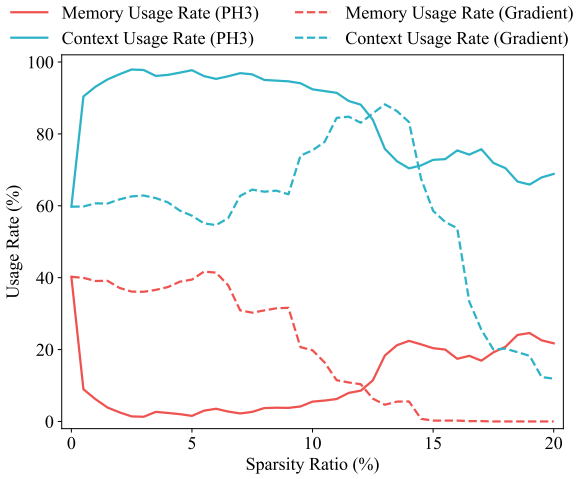


Figure 38: Impact of Pythia-6.9B's sparsity ratio on improving external context usage rate.

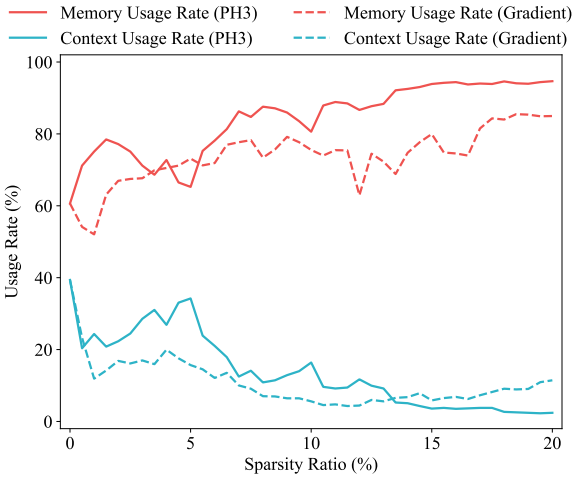


Figure 41: Impact of LLaMA2-13B's sparsity ratio on improving internal memory usage rate.

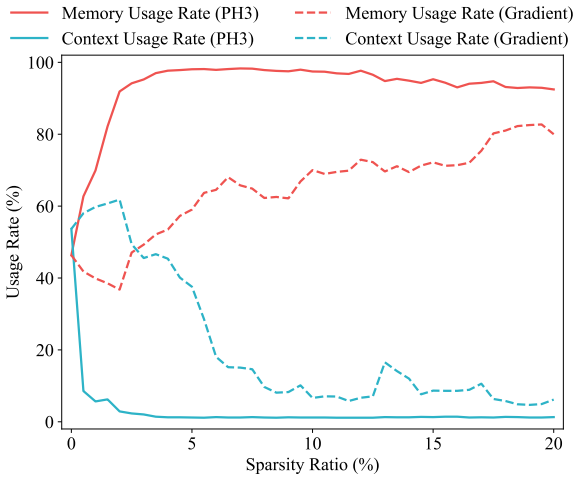


Figure 39: Impact of LLaMA2-7B's sparsity ratio on improving internal memory usage rate.

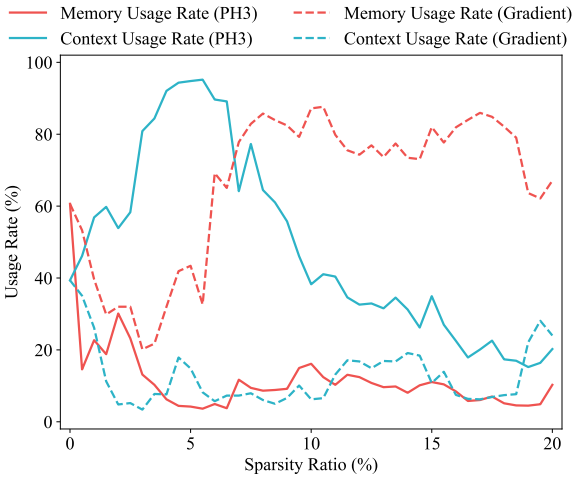


Figure 42: Impact of LLaMA2-13B's sparsity ratio on improving external context usage rate.

Model	Method		World Capital		World Capital D		Official Language		Country		Continent	
			RM	RC	RM	RC	RM	RC	RM	RC	RM	RC
OPT-1.3B		Base	40.5	59.5	36.3	63.7	20.3	79.7	26.8	73.2	19.2	80.8
	↑ Memory	Prompt	19.7	78.4	37.5	57.2	7.9	91.4	16.9	82.6	7.9	90.1
		Gradient	82.7	12.2	56.4	21.2	37.1	50.4	38.0	61.1	20.5	55.9
		PH3 (Ours)	**95.0**	**0.2**	**87.2**	**1.9**	**70.3**	**16.6**	**47.7**	**49.1**	**43.4**	**51.7**
	↑ Context	Prompt	17.0	81.4	38.5	57.7	9.3	89.5	15.9	83.8	6.3	93.3
		CAD	8.1	86.5	31.6	60.0	**3.2**	**89.6**	**0.1**	**99.5**	5.1	89.1
		Gradient	22.9	73.7	35.7	63.8	12.4	82.5	16.3	82.9	17.6	80.2
		PH3 (Ours)	**0.2**	97.0	**7.9**	**69.6**	12.8	84.1	4.0	85.4	**1.6**	44.1
		+ Prompt	0.4	**99.2**	10.8	68.8	9.3	88.3	2.4	92.5	2.5	**92.1**
OPT-2.7B		Base	40.2	59.8	46.6	53.4	8.8	91.2	26.5	73.5	4.3	95.7
	↑ Memory	Prompt	17.7	80.3	24.2	71.6	3.4	96.3	12.7	87.2	1.6	98.1
		Gradient	75.4	19.3	10.3	79.7	13.3	57.7	42.9	56.3	5.3	94.1
		PH3 (Ours)	**93.4**	**0.5**	**2.8**	**87.6**	**75.6**	**1.5**	**56.3**	**38.9**	**29.3**	**55.5**
	↑ Context	Prompt	4.7	94.9	11.1	86.9	3.2	96.3	13.2	86.5	0.7	99.0
		CAD	10.8	72.2	28.8	46.3	2.7	87.5	9.0	89.1	0.5	99.0
		Gradient	36.1	62.9	43.3	53.5	7.3	91.6	23.7	76.3	4.1	95.9
		PH3 (Ours)	1.3	97.8	3.4	81.6	4.1	94.9	9.0	90.9	0.9	98.6
		+ Prompt	**0.8**	**98.3**	**1.3**	**94.6**	**1.5**	**98.2**	**6.9**	**93.1**	**0.0**	**99.5**
Pythia-6.9B		Base	53.3	46.6	74.8	25.2	49.7	50.3	41.3	58.7	39.1	60.9
	↑ Memory	Prompt	44.7	51.2	41.8	37.8	16.5	81.4	12.8	87.1	36.2	61.6
		Gradient	56.5	35.8	72.8	22.4	56.7	36.3	37.9	62.1	40.1	58.9
		PH3 (Ours)	**90.2**	**6.7**	**88.4**	**10.2**	**71.5**	**11.1**	**41.8**	**57.3**	**66.0**	**31.0**
	↑ Context	Prompt	32.7	63.7	32.0	44.8	8.9	90.5	10.4	89.5	31.7	75.6
		CAD	14.3	55.1	**22.0**	27.6	**3.3**	78.1	8.5	91.2	12.3	82.2
		Gradient	41.4	53.7	61.8	35.6	48.2	51.3	34.3	65.6	41.7	53.0
		PH3 (Ours)	6.7	81.7	34.4	30.0	16.4	70.0	3.4	96.3	3.3	94.5
		+ Prompt	**0.6**	**98.6**	24.4	**60.4**	**3.3**	**95.6**	**0.3**	**99.5**	**0.7**	**98.8**
Pythia-12B		Base	59.7	40.3	64.6	35.4	34.3	65.7	35.0	65.0	43.0	57.0
	↑ Memory	Prompt	5.8	94.1	43.7	53.3	11.1	85.8	2.4	97.5	10.1	88.5
		Gradient	62.9	27.7	63.1	30.2	39.3	37.5	46.2	53.3	42.1	56.9
		PH3 (Ours)	**95.0**	**0.6**	**82.4**	**2.2**	**69.9**	**6.9**	**57.1**	**35.9**	**70.1**	**9.6**
	↑ Context	Prompt	6.2	93.6	34.1	62.0	21.1	77.6	**1.7**	**98.3**	6.3	92.5
		CAD	3.1	65.7	13.6	42.9	2.0	89.4	3.7	94.7	11.3	80.0
		Gradient	59.3	19.6	33.3	25.0	21.3	54.6	28.5	71.3	40.2	52.1
		PH3 (Ours)	18.6	76.1	56.9	33.3	10.9	80.4	16.9	76.7	26.9	67.9
		+ Prompt	**1.9**	**97.6**	**17.7**	**75.8**	**3.8**	**95.5**	2.2	97.7	**2.4**	**97.0**
LLaMA2-13B		Base	60.6	39.4	74.6	25.0	1.6	98.4	26.2	73.7	5.6	93.3
	↑ Memory	Prompt	0.4	99.6	78.5	21.0	0.5	99.5	22.3	77.6	10.7	89.3
		Gradient	77.7	10.0	89.9	9.5	26.1	68.3	**48.1**	**51.4**	**30.7**	49.0
		PH3 (Ours)	**86.3**	**12.5**	**91.5**	**8.1**	**71.2**	**11.6**	47.7	51.6	11.9	**45.4**
	↑ Context	Prompt	**0.0**	**100.0**	70.9	28.7	**0.0**	**100.0**	7.1	92.7	3.6	96.4
		CAD	6.2	91.0	**1.1**	**98.8**	**0.0**	**100.0**	0.5	99.5	**0.0**	99.6
		Gradient	46.3	33.9	74.1	25.4	16.5	83.3	20.2	79.5	3.6	96.4
		PH3 (Ours)	6.2	92.1	24.1	75.9	1.3	98.7	5.6	94.4	0.4	99.6
		+ Prompt	**0.0**	**100.0**	31.0	68.0	**0.0**	**100.0**	**0.0**	**100.0**	**0.0**	**100.0**

Table 2: Experimental results of OPT-1.3B, OPT-2.7B, Pythia-6.9B, Pythia-12B and LLaMA2-7B on five datasets. Bolds denote the best results.

Method	GPT-2 XL	GPT-J	OPT-2.7B
Base	45.6	54.8	51.4
Prompt	47.6	57.4	54.1
CAD	44.5	55.0	50.2
Gradient	45.3	55.0	50.8
PH3 ($k = 1$)	47.1	57.5	53.5
PH3 ($k = 3$)	52.2	58.6	55.4
+ Prompt	**54.0**	**59.6**	**56.7**
PH3 ($k = 5$)	49.4	56.3	54.0

Table 3: Experimental results (Recall) of GPT-2 XL, GPT-J and OPT-2.7B on the NQ dataset. Bolds denote the best results.

Model	#Layer L	#Head M
GPT-2 XL	48	25
GPT-J	28	16
OPT-1.3B	24	32
OPT-2.7B	32	32
Pythia-6.9B	32	32
Pythia-12B	36	40
LLaMA2-7B	32	32
LLaMA2-13B	40	40

Table 4: Model details.