
Retrieval Head Mechanistically Explains Long-Context Factuality

Wenhao Wu^λ Yizhong Wang^δ Guangxuan Xiao^σ Hao Peng^π Yao Fu^μ

^λPeking University ^δUniversity of Washington ^σMIT ^πUIUC ^μUniversity of Edinburgh
 waynewu@pku.edu.cn haopeng@illinois.edu yao.fu@ed.ac.uk
https://github.com/nightdessert/Retrieval_Head

Abstract

Despite the recent progress in long-context large language models (LLMs), it remains elusive how these transformer-based language models acquire the capability to retrieve relevant information from arbitrary locations within the long context. This paper aims to address this question. Our systematic investigation across 4 model families, 6 model scales, and 3 types of finetuning reveals that a special type of attention heads are largely responsible for retrieving relevant information from long context, which we dub *retrieval heads*. We identify important and intriguing properties of retrieval heads: (1) *universal*: all the explored models with long-context capability have a set of retrieval heads; (2) *sparse*: only a small portion (less than 5%) of the attention heads are retrieval. (3) *intrinsic*: retrieval heads already exist in models pretrained with short context. When extending the context length to 32-128K by continual pretraining, it is still the same set of heads that perform information retrieval. (4) *dynamically activated*: take Llama-2 7B for example, 12 retrieval heads always attend to the required information no matter how the context is changed. The rest of the retrieval heads are activated in different contexts. (5) *causal*: completely pruning retrieval heads leads to failure in retrieving relevant information and results in hallucination, while pruning random non-retrieval heads does not affect the model’s retrieval ability. We further show that retrieval heads strongly influence chain-of-thought (CoT) reasoning, where the model needs to frequently refer back the question and previously-generated context. Conversely, tasks where the model directly generates the answer using its intrinsic knowledge are less impacted by masking out retrieval heads. These observations collectively explain which internal part of the model seeks information from the input tokens. We believe our insights on retrieval heads foster future research on reducing hallucination, improving reasoning, and compressing the KV cache.

1 Introduction

This work studies the internal mechanism of how long-context language models can utilize information at arbitrary locations within the input. Recent advances in long-context language modeling [1, 20, 6] show inspiring results, particularly on the Needle-in-a-Haystack test [14], which asks the model to precisely retrieve the information of a short sentence (the needle) within a long context (the haystack). Such capability is the basis of more advanced long-context tasks, which usually interleaves retrieval and reasoning in a multi-step fashion [17]. Based on extensive experiments across 4 model families, 6 model scales, and 3 types of finetuning, we show that across the models’ attention layers, there exist a small number of retrieval heads that search the information being asked, and redirect the relevant tokens from the input to the output. Activation of retrieval heads explains

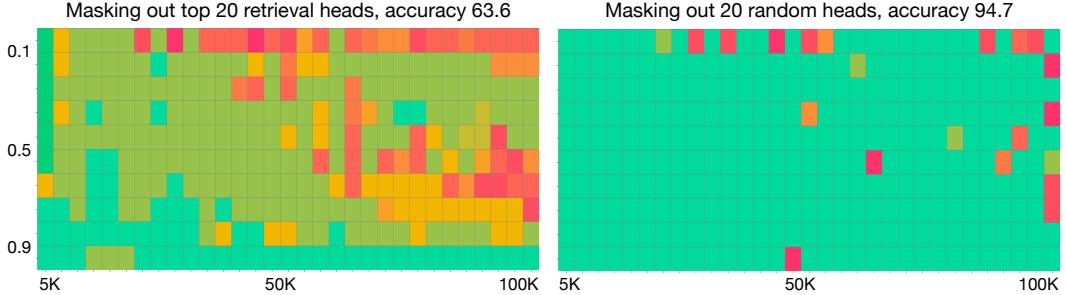


Figure 1: Retrieval heads are the heads that redirect information from the input to the output. Left: masking out the top retrieval heads of LLaMA 2 7B 80K, its Needle-in-a-Haystack performance drops significantly, and the model hallucinates during decoding. Right: masking out random non-retrieval heads does not influence the model’s needle-in-a-haystack behavior. We further note that the retrieval head only influence factuality but not the language capability: when they are masked, the model hallucinates by saying “go to the beach”, which is still a fluent sentence, but not factual (as the correct answer is “eating a sandwich at Dolores park.”).

whether the output is factual or hallucinated. When such heads are activated, the model behaves faithful to the input document. When they are not activated, or intentionally masked out in controlled experiments (Fig. 1), the model cannot find the relevant information and hallucinate instead.

The discovery of the retrieval head is motivated by the question of what the attention mechanism is doing when the model can or cannot find the given needle. We take important inspiration from two existing works: the CopyNet [10] and the Induction Head [19]. The CopyNet is essentially a single-layer, single-head attention mechanism in the age of RNNs that copy-paste tokens from the input to the output. Induction Heads [19] are special heads within a multi-layer, multi-head attention network that implements an implicit program induction algorithm. Combining the observation from the two works, we naturally hypothesize that, just like induction heads are accountable for in-context learning, there might exist special heads that are accountable for information retrieval and implement a conditional copy-paste algorithm.

We design algorithms to detect retrieval heads within the transformer architecture (Sec. 2), and conduct large-scale experiments to demonstrate important properties of them (Sec. 3): (1) retrieval heads are universal and sparse: for any model family (LLaMA [21], Yi [25], QWen [2] and Mistral [12]), at any scale (6B, 14B, and 34B and 8×7 B), either base or chat, either dense or MoE, as long as the model can precisely recite the input information, they have a small number of retrieval heads (Fig. 1); (2) they are intrinsic: the base model (e.g., LLaMA2 base) already contains retrieval heads (as a consequence of large-scale pretraining). Subsequent derivations, such as the long-context continue pretraining (LLaMA2 7B 80K), chat fine-tuning (Qwen Chat), or even sparse upcycling [16, 13] uses the same retrieval heads as the base model (Fig. 5); (3) they are dynamically activated according to the context: the strongest retrieval heads (e.g., 13 for LLaMA 2 7B) are always activated no matter what the required information is, while weaker retrieval heads are activated on different parts of the required information; consequently these heads compensate each other’s functionality: removing a subset of the heads, the model at least retrieve part of the required information; (4) the retrieval heads are causal: say we put a needle “the best thing to do in San Francisco is to eat a sandwich in Dolores Park on a sunny day”, completely masking out retrieval heads, the model hallucinates (by saying the best thing is to visit Golden Gate bridge); partially masking out the heads, the model retrieves part of the needle (e.g., it gets the sandwich but forget the Dolores Park); masking out random non-retrieval heads, the model still find full needle; when we do not mask the head yet the model still hallucinate in some cases, the retrieval heads are not activated. We further note that chain-of-thought reasoning also heavily relies on retrieval heads because the model needs to refer back the input information, indicating a complex relationship between the model’s retrieval and reasoning capability.

The discovery of retrieval head has profound implications on long-context modeling: (1) it marks a significant step forward in the field of mechanistic interpretability [3, 19] because for the first time we pin point a particular subnet implementing the conditional retrieval algorithm; (2) it explains why certain context-compression methods fail to keep factuality (because they removes the retrieval head, e.g., in Xiao et al. 24), and suggests future research on KV cache compression [7, 15], a key problem for deploying long-context models, should consider the influence of retrieval heads.

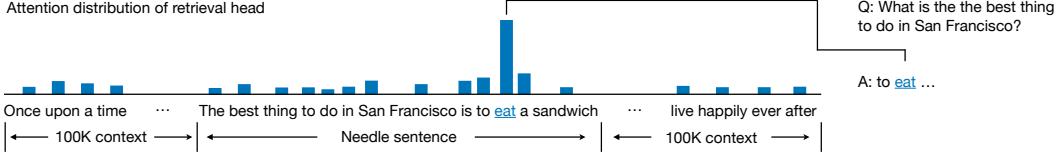


Figure 2: An attention head that performs a copy-paste operation if the token that the head attend to is the same the token being generated. The retrieval score of a head is defined as the frequency of this head’s copy-paste behavior when answering questions that asks for raw information from the input.

Table 1: We consider a wide range of language model families and show that the basic properties of retrieval heads are universal and consistent across all language models we study.

Base Model	Variant	Variation Type
Llama-2-7B	Llama-2-7B-80K	Length Extension via Continue Pretrain
	Llama-2-13B-64K	Model Scaling and Length Extension
Mistral-7B-v0.2	Mistral-7B-Instruct-v0.2	SFT and RLHF
	Mixtral-8x7B-v0.1	Sparse Upcycling to Mixture of Experts
Yi-6B	Yi-6B-200K	Length Extension via Continue Pretrain
	Yi-34B-200K	Model Scaling and Length Extension
Qwen1.5-14B	Qwen1.5-14B-Chat	SFT and RLHF

2 Detecting Retrieval Head

To detect which head is implementing the retrieval algorithm, we introduce a *retrieval score* to measures the frequency of a head’s copy-paste behavior during autoregressive decoding. An attention head with high retrieval score suggests that statistically across various contexts, this head is frequently copying the input tokens from the input to the output.

Needle-in-a-Haystack Our retrieval head detection algorithm roots from the needle-in-a-Haystack test, which asks the model to copy-paste the input tokens to the output. Given a question q and its corresponding answer k (the needle), we insert k in a given context x (the haystack) at a random position index range i_q . The language model is then tasked with answering q based on the haystack with the inserted needle. We set q and k unique and irrelevant with the given long context, ensuring that if an answer is correctly generated, it is indeed copied from the context, not from the model’s internal knowledge.

Retrieval Score for Attention Heads We define the retrieval score as the frequency of a head’s copy-paste operations. Specifically, during auto-regressive decoding (we use greedy decoding by default), denote the current token being generated as w and the attention scores of a head as $a \in \mathcal{R}^{|x|}$. As demonstrated in Fig. 2, we say an attention head h copies and pastes a token from the needle to the output sentence if it follows two criteria: (1) $w \in k$, i.e., w is a token within the needle sentence. (2) $x_j = w, j = \arg \max(a), j \in i_q$, i.e., the input token that receives the most attention probability mass by this head is a token within the needle and is the same token as the currently generated token. Let g_h be the set containing all tokens copy and pasted by a given head h , we define:

$$\text{Retrieval score for head } h = \frac{|g_h \cap k|}{|k|}, \quad (1)$$

Intuitively, retrieval score represents a token-level recall rate of the most attended tokens by an attention head. For example, when retrieving a needle of 10 tokens, a retrieval score of 0.9 indicates that the attention head has copies and pasted 9 tokens in the 10-token target answer.

Retrieval Head Detection Algorithm We calculate the retrieval score for all attention heads under a diverse set of input contexts. For each language model we consider, we compile three sets of Needle-in-a-Haystack samples, each consisting of a unique tuple (q, k, x) . For each sample, we make sure (q, k) is semantically irrelevant with x and that q cannot be answered using the model’s existing knowledge by manually inspecting the model output. Then for each (q, k, x) sample, we

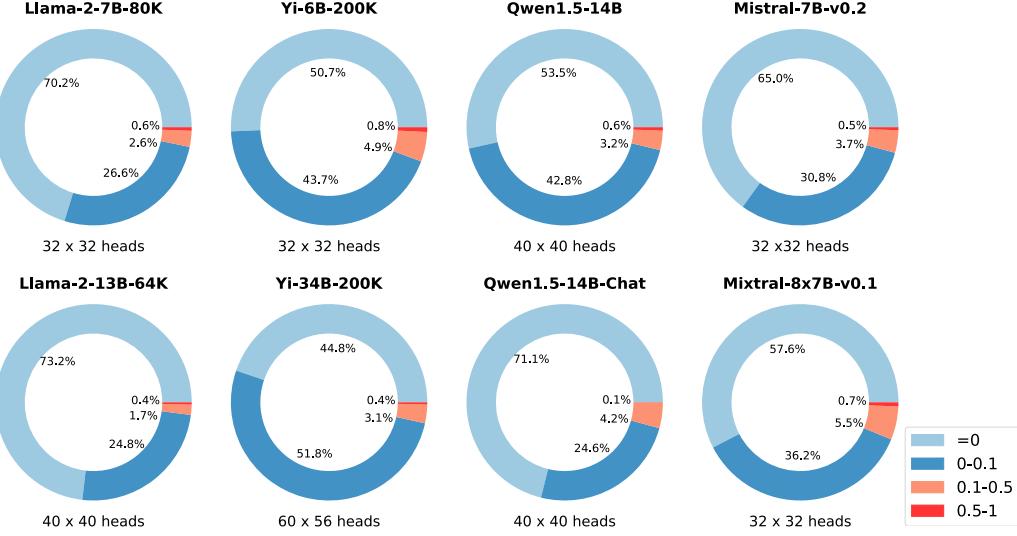


Figure 3: Retrieval heads are universal and sparse across model family and scale. For all models we consider, less than 5% of the attention heads are activated more than 50% of the time (with a retrieval score higher than 0.5) when retrieval is required.

perform Needle-in-a-Haystack on 20 different length values uniformly sampled from 1K-50K, where in each length, q is inserted in 10 different depth uniformly ranging from the start to the end of x . We note that this scale of tests gives stable outputs as the average retrieval score converges after just a few samples. In total, each language model is subjected to approximately 600 instances of retrieval testing. We calculate the retrieval score for each attention head in each test and use the average of these scores as the head’s final retrieval score. The attention heads with relatively larger retrieve score can be considered as retrieval head. In our case (Fig. 3), we set the threshold as 0.1, meaning that as long as the head performs copy-paste in 10% of the times, we consider it a retrieval head.

3 Basic Properties of Retrieval Heads

This section discusses important properties of retrieval heads: (1) universal and sparse: any model that exhibits long-context capability has a small set of retrieval heads; (2) dynamic: most of retrieval heads are activated under different contexts; (3) intrinsic: retrieval heads are already within the base model as a consequence of large-scale pretraining. Subsequent models reuse the same set of heads. Our results are supported by extensive experiments on a large spectrum of models (Table 1). To investigate the influence of continued pretraining for context length extension, we compare Llama-2-7B 4K to Llama-2-7B-80K and Llama-2-13B-60K [6]. To examine the effect of alignment, we have study Mistral-7B-Instruct-v0.2 and Qwen-1.5-14B-Chat [2] and compare them to their base versions. We further choose Mixtral-8x7B-v0.1 [13], a mixture of expert versions derived from Mistral-7B-v0.2, presumably via sparse upcycling [16], to study retrieval heads in different architectures.

3.1 Universal and Sparse

Figure 3 demonstrate that a sparse set of retrieval heads exist in all models we consider, regardless of the various pertraining, fine-tuning recipes and the underlying architecture. Between 25% and 52% of the heads exhibit copy-paste behaviors at a very low frequency, with a score of between 0 and 0.1. Approximately 45% to 73% of attention heads have 0 retrieval score, meaning that they have other functionality than retrieval. Only about 3% to 6% of the attention heads have a retrieval score larger than 0.1 (recall that a retrieval score 0.1 means a head retrieves at least 10% of the tokens being asked). It is also intriguing that although all model parameters, particularly the total number to attention heads, are largely different from each other, their ratio of retrieval heads lays in the same interval (about 5%).

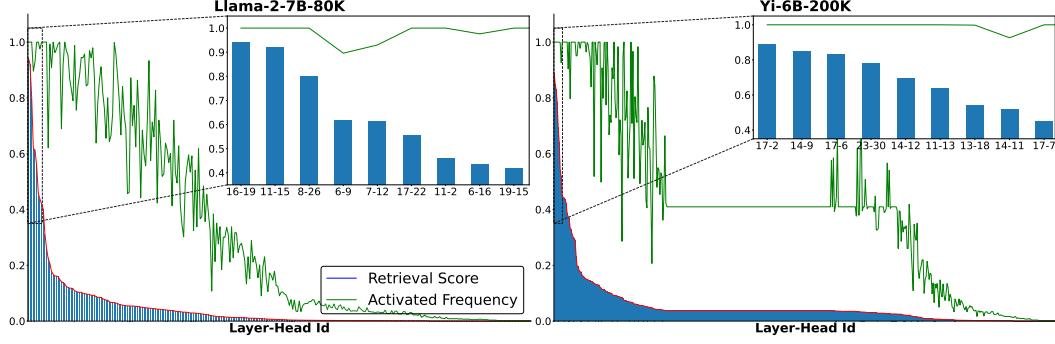


Figure 4: Retrieval score (blue): average number of activated tokens; Activation Frequency (green): frequency of at least activated on one token. The *gap* between the two curves shows *context-sensitivity*: a head of high activation frequency but low retrieval score means it is only activated on certain tokens and contexts; a head of high retrieval score means it is activated on almost any context. For both LLaMA and Yi, there exist strong heads that are not sensitive to context and always activated.

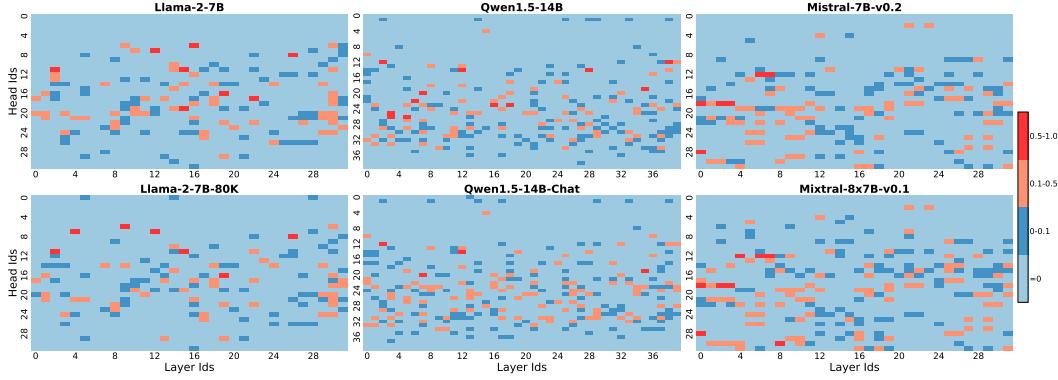


Figure 5: Retrieval head is intrinsic and already within the base model. The subsequent model derivations, either by continue pretraining (LLaMA 2 7B 80K) or chat finetuning (Qwen 1.5 14B Chat) or sparse upcycling (Mixtral 8 \times 7B), use the same set of retrieval head as the base model, as demonstrated by a high level of similarity between the heatmap patterns.

3.2 Dynamically Activated Based on Tokens and Contexts

Now we study how sensitive a retrieval head is to its input context, i.e., whether a head is consistently activated no matter what the context is, or if a head is activated only on specific contexts. For the needle sentences "the best thing to do in San Francisco is eating a sandwich in Dolores park in a sunny day", some heads are activated on the full sentence, whereas other heads only activated on certain tokens like "eating a sandwich" or "in Dolores park". We define *activation frequency*, the frequency of a head being activated on *at least one token* (v.s., the retrieval score measures the *average* number of activated tokens). A head of high activation frequency but low retrieval score means it is only activated on certain tokens and contexts. As is shown in Fig. 4, Llama-2-7B-80K and Yi-6B-200K have 12 and 36 strongest retrieval heads, respectively, that are always activated (activation frequency equal to 1) under all the contexts we consider. Weaker heads only activate on certain tokens and contexts.

3.3 Intrinsic

We show that the retrieval heads, thus the ability of utilizing information at arbitrary location of the input, is an intrinsic property [6] of the base model as a consequence of large-scale pretraining, with subsequent small-scale training exerting only minor alterations to these head activation patterns. In Figure 5, we present the retrieval score distributions for a range of base models in the initial row, followed by their corresponding variants in the subsequent row. We see that regardless of the models

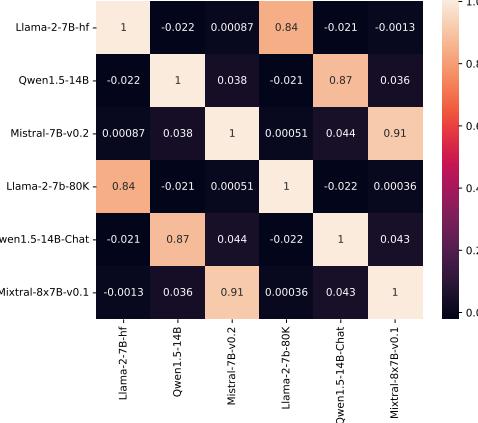


Figure 6: The retrieval heads of models of the same family are strongly correlated, i.e., the chat model and base model typically uses the same set of retrieval heads. The retrieval heads of models of different families are clearly different.

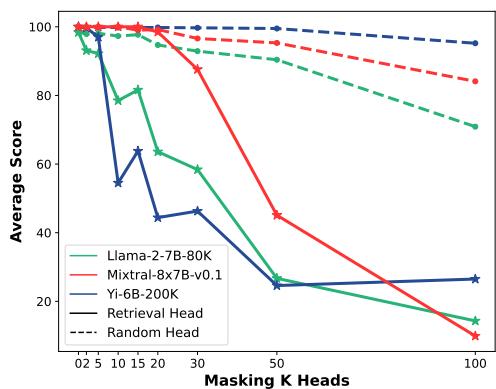


Figure 7: Masking out top K retrieval heads vs K random heads. For all models are consider, the removal of retrieval heads clearly reduces the Needle-in-a-Haystack performance, versus the removal of non-retrieval heads have much weaker influence.

being continuously pre-trained, chat fine-tuned, or sparsely upcycled, there is a notable consistency in their retrieval scores heatmaps. Figure 6 offers a more direct and strict examination, where we compute the statistical correlations between different models. The data reveal a high degree of correlation in the retrieval score distributions between base models and their respective variants, with a Pearson correlation coefficient exceeding 0.8. Models from different families exhibit a correlation coefficient of less than 0.1, indicative of their distinct pretraining recipes.

4 Influence on Downstream Tasks

This section examines how retrieval heads influence downstream tasks. Across the experiments we use Mistrial-7B-Instruct-v0.2 [18] as it is a popular and strong open language model with 32K context length. We first show that retrieval heads explains the factuality of Needle-in-a-Haystack test. When the model can retrieve the needle, retrieval heads are always activated. When the model cannot retrieve the needle and hallucinate instead, retrieval heads are either partially activated or not activated. Then we show that retrieval heads significantly influence question answering that requires extracting the information from the input, but does not strongly influence tasks where the model directly produce answers based on its internal knowledge. We further explore how retrieval heads influence more sophisticated reasoning behaviors like chain-of-thought [23].

4.1 Retrieval Heads Explains Factuality in Needle-in-a-Haystack

We start with a closer look at Needle-in-a-Haystack and construct an additional set of needle tests that are different from the three sets used for retrieval head detection. We gradually mask out the number of retrieval/ random heads and see how the model’s behavior changes. As is shown in Fig. 7, masking out retrieval heads severely damages the model’s Needle-in-a-Haystack performance, while masking out random heads shows much smaller performance impact. Notably, when increasing the number of masked heads K to 50 (about 5% of the full number of heads), all models’ needle test performance drop to below 50, showing that the top retrieval heads are responsible for most of the needle retrieval behavior.

We observe three types of error: (1) Incomplete retrieval, where models only captured partial information of the target and omitted key details; (2) Hallucination, where models generated hallucinated sentences; and (3) Wrong extraction, where models incorrectly retrieved irrelevant content from the haystack. Without any masking, instances of wrong extraction occurred when retrieval heads were active but focused on incorrect sections. During hallucinated generation, retrieval heads predominantly

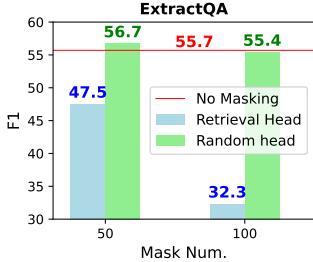


Figure 8: Masking out retrieval heads severely damages ExtractQA performance.

attended to the initial token of the input, which is often known as "attention sink" [24], presumably dummy tokens that provide less meaningful signal.

As we increase the number of masked heads, initially, a small subset of the most robust retrieval heads are masked, and incomplete retrieval began to appear. In the absence of the strongest retrieval heads, the remaining weaker heads only managed to retrieve a fraction of the target information. Metaphorically, each retrieval head holds a small piece of the "needle," yet these pieces cannot form a complete one, resulting in an incomplete final output. This phenomenon typically begins when the mask out heads of retrieval score larger than 0.4. As we further increase the number of mask, hallucinations become more prevalent, signaling a complete failure of the retrieval capability.

4.2 Influence on Extractive QA

Now we study how retrieval heads influence more realistic tasks beyond Needle-in-a-Haystack. We use extractive QA as a test bed, as common usecase of long-context model where the user typically upload a pdf (research papers, finantial reports, legal documents, .etc) and ask questions about specific information within the document. To make sure the knowledge being asked does not exist in the model's internal knowledge, we synthesize an extractive QA dataset by selecting a set of *up-to-date* news articles, extract a paragraph from it, and asking GPT-4 to produce a question-answer pair based on the extracted paragraph, similar to the evaluation conducted in Anthropic [1]. As illustrated in Figure 8, randomly masking out non-retrieval heads demonstrated no significant impact on performance. Masking out retrieval heads led to a substantial decrease in F1 scores, with reductions of 9.2% and 23.1%. These observations demonstrate that real-world document QA tasks heavily rely on the functionality of retrieval heads.

4.3 Chain-of-Thought Reasoning also Requires Retrieval Heads

We test Mistrial-7B-Instruct-v0.2's performance on MMLU [11], MuSiQue and GSM8K [4], with and without chain-of-thought reasoning. As is shown in Fig. 10, if we use answer-only prompting (without CoT), masking out either retrieval or random heads do not really influence the performance, presumably because the model's generation is based on its internal knowledge primarily stored in the FFN layers [8]. For CoT styled reasoning, masking out retrieval heads signifantly influence the model's performance. Upon inspecting typically error cases (Fig. 11), we find out the model becomes "blind" to important input information and hallucinate instead. We find the relationship between CoT and retrieval heads particularly intriguing as it may offers deeper insights into model's complex reasoning performance. We leave more in-depth studies to future research.

5 Discussions

General Functionalities of Attention Heads

For transformer language models, we tend to view the functionality of FNNs layers to be the place for storing knowledge [8], and the attention layers to be the place for implementing algorithms [19]. The induction head discussed in Olsson et al. [19] typically searches repeated patterns of the input, which is at a certain level similar to the retrieval heads (as it also searches and repeats information). Different than the induction heads, the

Figure 9: When the model fails to retrieve the full needle, there are two typical errors: (1) incomplete retrieval, where the retrieval heads miss part of the information "eat a sandwich"; (2) hallucination, where the retrieval heads attend to the initial tokens.

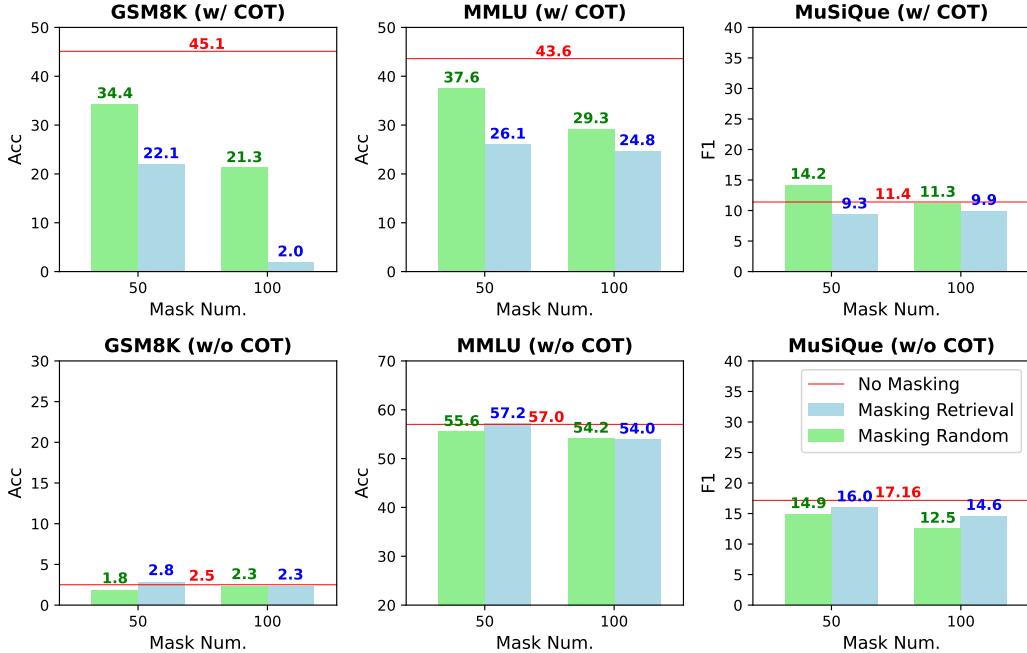


Figure 10: Retrieval heads significantly influence tasks that require chain-of-thought reasoning. This is because typically in a reasoning chain, the next step reasoning requires the model to refer to previous information. See Fig. 11 for examples.

<p>Question 1: Mishka bought 3 pairs of shorts, 3 pairs of pants, and 3 pairs of shoes. One pair of shorts costs \$16.50. One pair of pants costs \$22.50 and one pair of shoes costs \$42. How many dollars did Mishka spend on all the clothing items?</p> <p>Masking Top 100 Retrieval Heads: Let's calculate the cost of each type of clothing for Mishka: 1. For shorts, there are 3 pairs, so the total cost is $3 * \\$16.50 = \\59.10. 2. For pants, the cost is the same, so 3 pairs cost $3 * \\$22.50 = \\67.50. 3. In total, Mishka spent $\\$59 + \\$67 = \\$125$. So, the answer is \$125.</p>	<p>Question 2: James decides to run 3 sprints 3 times a week. He runs 60 meters each sprint. How many total meters does he run a week? Masking Top 100 Retrieval Heads: Let's think through this step by step: 1. James runs 3 times a week for each sprint session. 2. During each session, he covers a distance of 60 meters (3 times the length of a standard football field goal post). 3. Therefore, the total distance James covers in a week is: $6 \text{ (sessions in a week)} \times 1 \text{ (number of sprints in session)}$ $\times 60 \text{ (meters in a sprint)} = 360 \text{ meters}$. So, James runs a total of 360 meters every week.</p>
---	--

Figure 11: When we mask out retrieval heads, the model becomes “blind” to important information in the question description resulting in incorrect reasoning chains.

retrieval heads are typically responsible for redirecting the information according to the context, but do not for inferring programs. We tend to believe that there exist more algorithm and functionalities implemented by other types of attention heads to be discovered by future research.

Relationship to Local and Linear Attention and State-Space Models Although there exist numerous works about local [24] / linear [22] attention, state space models [9], and hybrid architectures [5] achieving inspiring efficiency in long-context modeling, so far there is no linear attention / SSM architecture that passes the Needle-in-a-Haystack test to the best of our knowledge, suggesting that the full attention might be a must for long-context information retrieval. One example is that the Mistral v0.1 [12] uses sliding window attention but cannot pass needle-in-a-haystack, and their authors changes the attention to full in v0.2 [18], then it can pass the needle test. Our results showing strong evidence why full attention is a must. For the model to precisely utilize input information at arbitrary location, it is crucial for the retrieval heads to work on the full KV cache.

Applications to KV Cache Compression The problem that the KV cache is too large and occupies a large chunk of the GPU memory severely hinders the deployment of long-context models. For example, for LLaMA 2 7B, the KV cache of 100K tokens requires more than 50GB memory, while 2K context requires less than 1GB memory. If we serve this model on one 80G A100, then the

concurrency of 100K context can be 50 times less than 2K context queries, which is prohibitively expensive. The results from this work indicates that we might be possible to radically prune out the KV cache corresponding to the non-retrieval heads (recall in Fig. 3 shows only 5% of the heads are retrieval) and significantly reducing the deployment cost of long-context models. We leave this study to future research.

6 Conclusions

This paper discovers retrieval heads, a special set of attention heads that are responsible for implementing the conditional copy algorithm and redirect information from the input to the output. Retrieval heads are the primarily reason why a successful long-context model can pass the Needle-in-a-Haystack test, and their activation explains why a language model is faithful to the input or hallucinate. Compared to non-retrieval heads, retrieval heads have a stronger influence on downstream tasks that require the model to precisely recall the input information, either in extractive question answering or chain-of-thought reasoning. We believe this work will foster future research on reducing hallucination, improving reasoning, and compressing the KV cache.

References

- [1] Anthropic. Model card and evaluations for claude models, July 2023. URL <https://www.anthropic.com/product>.
- [2] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.
- [3] Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. <https://transformer-circuits.pub/2023/monosemantic-features/index.html>.
- [4] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.
- [5] Soham De, Samuel L Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*, 2024.
- [6] Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*, 2024.
- [7] Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*, 2023.
- [8] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.
- [9] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.
- [10] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. Incorporating copying mechanism in sequence-to-sequence learning. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1154. URL <https://aclanthology.org/P16-1154>.

- [11] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2020.
- [12] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- [13] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- [14] Greg Kamradt. Needle in a haystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2023.
- [15] Hao Kang, Qingru Zhang, Souvik Kundu, Geonhwa Jeong, Zaoxing Liu, Tushar Krishna, and Tuo Zhao. Gear: An efficient kv cache compression recipefor near-lossless generative inference of llm. *arXiv preprint arXiv:2403.05527*, 2024.
- [16] Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. Sparse upcycling: Training mixture-of-experts from dense checkpoints. *arXiv preprint arXiv:2212.05055*, 2022.
- [17] Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. In search of needles in a 10m haystack: Recurrent memory finds what llms miss. *arXiv preprint arXiv:2402.10790*, 2024.
- [18] Mistral. Model card for mistral-7b-instruct-v0.2, April 2024. URL <https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2>.
- [19] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.
- [20] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.
- [21] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [22] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.
- [23] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.
- [24] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.
- [25] Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*, 2024.