# AttnComp: Attention-Guided Adaptive Context Compression for Retrieval-Augmented Generation

**Anonymous ACL submission**


## Abstract

Retrieval-augmented generation improves the factual accuracy of Large Language Models (LLMs) by incorporating external context, but often suffers from irrelevant retrieved content that hinders effectiveness. Context compression addresses this issue by filtering out irrelevant information from context before LLM generation. However, existing methods struggle to adaptively adjust compression rates for different context, maintain low latency and integrate information across multiple documents. To overcome these limitations, We introduce AttnComp, an adaptive, efficient and context-aware compression framework. By leveraging the attention mechanism of LLMs to identify relevant information, AttnComp employs a Top-P compression algorithm to retain the minimal set of documents whose cumulative attention weights exceeds a predefined threshold. In addition to compression, AttnComp estimates response confidence by assessing the overall relevance of the retrieved content, enabling users to gauge response reliability. Experiments demonstrate that AttnComp outperforms existing compression methods and uncompressed baselines, achieving higher accuracy with substantial compression rates and lower latency.


## 1 Introduction

Retrieval-Augmented Generation (RAG) enhances the factual accuracy and reliability of Large Language Models (LLMs) in knowledge-intensive tasks by integrating retrieved context into their generation process(Lewis et al., 2020; Borgeaud et al., 2022; Izacard et al., 2023; Ram et al., 2023; Xu et al., 2023b). However, practical RAG applications often grapple with retrieved content containing substantial irrelevant information, even entirely unrelated to the query(Sauchuk et al., 2022). This gives rise to three primary issues: first, LLMs can be misled by such noise, leading to incorrect answers(Shi et al., 2023; Jin et al., 2024a; Yoran et al., 2024; Wu et al., 2024a); second, LLMs struggle to identify and utilize key information effectively as context length increases(Liu et al., 2024); and third, irrelevant content unnecessarily inflates input sequences, escalating computational overhead.

To mitigate these issues, context compression has emerged as a promising solution to filter out irrelevant information before generation. Existing methods can be categorized into abstractive and extractive approaches. Abstractive methods leverage LLMs to summarize or rewrite retrieved content via autoregressive generation(Xu et al., 2023a; Yoon et al., 2024; Zhu et al., 2024). While achieving high compression rates, they incur significant latency due to token-by-token decoding. Extractive methods instead select relevant spans from the original content, offering greater efficiency(Jiang et al., 2024; Hwang et al., 2024; Chirkova et al., 2025). However, current extractive methods typically only assess the relevance of individual sentence or document to the query, limiting their ability to integrate information across broader context. Furthermore, many such approaches rely on fixed compression rates or target lengths(Xu et al., 2023a; Jiang et al., 2024), ignoring the variable proportion of relevant content and risking under- or over-compression.

Consequently, we posit that an effective context compression method should exhibit three key properties: (1) **Adaptive**: It should dynamically adjust the compression rates based on the proportion of relevant information within the context. (2) **Efficient**: It should maintain low computational cost and latency, ensuring rapid processing for real-time applications. (3) **Context-Aware**: It should integrate and synthesize information from the entire retrieved content to accurately identify relevant segments. However, to the best of our knowledge, no existing compression method simultaneously satisfies all three of these properties.

To bridge this gap, we introduce *AttnComp* (Attention-guided Context Compression), an adap-

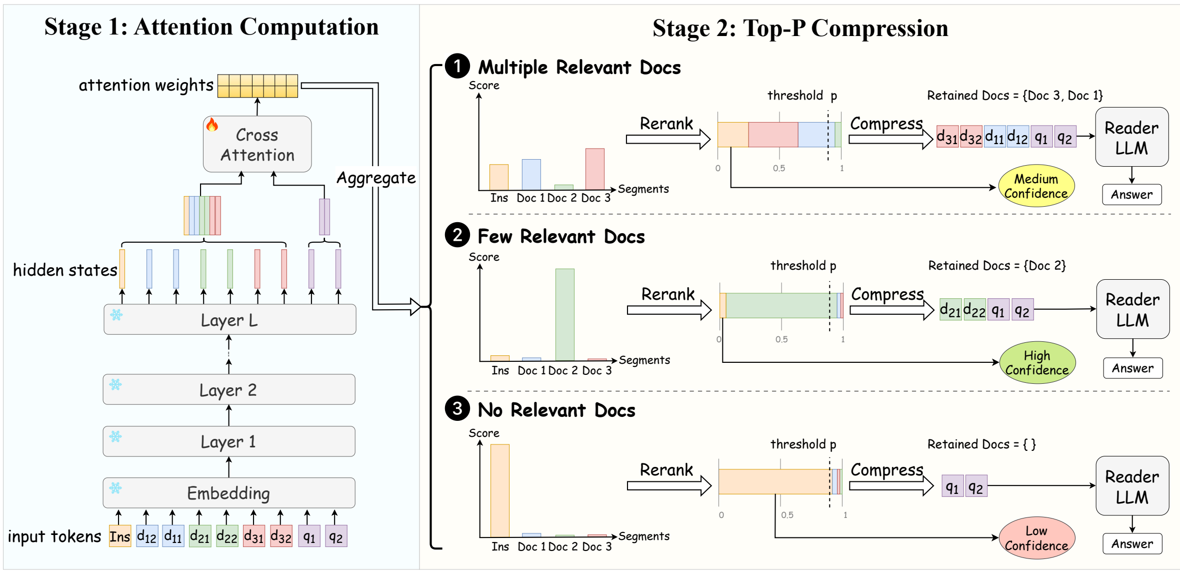



Figure 1: Illustration of the AttnComp Framework. AttnComp consists of two stages: Stage 1 involves attention computation, where attention weights are calculated from the query (q) to the context composed of the instruction (Ins) and documents (d); Stage 2 applies a Top-P compression algorithm to select the most relevant documents and generate a confidence score for the RAG response. Three cases are illustrated: (1) Documents 1 and 3 are relevant and retained; (2) Only Document 2 is relevant and retained; (3) All documents are irrelevant and filtered out.

tive, efficient and context-aware extractive compression method that leverages the inherent attention mechanisms of LLMs. As illustrated in Figure 1, the AttnComp pipeline consists of two stages: (1) Attention Computation. Given a prompt that combines the instruction, retrieved documents and query, we compute attention weights from middle layers of the LLM to quantify the relevance of each text segment to the query. (see Sec. 4.1 for details). (2) Top-P Compression. We aggregate the attention weights to compute scores for the instruction and each document. Documents are then ranked by score, and the top ones are retained until their cumulative score, combined with that of the instruction, reaches a predefined threshold. Compared to fixed-length compression, this approach adjusts the retained content based on attention distribution, allowing for flexible selection ranging from no documents to all documents (see Sec. 4.2 for details).

We observe that while the attention mechanisms in LLMs inherently capture relevance, they can still assign high attention to irrelevant content. This is particularly evident when all retrieved documents are irrelevant, as the model fails to shift attention away from the documents, with some irrelevant ones consistently receiving high attention. To address this, we fine-tune the cross-attention layer of the model to direct attention to relevant documents when present, or to the instruction when all documents are irrelevant (see Sec. 4.3 for details).

Experimental results on multiple QA datasets highlight the superior performance of AttnComp. It achieves a 1.9 point accuracy improvement over the uncompressed baseline, while other compression methods incur at least a 3 point decrease. This advantage is even more pronounced in multi-hop question answering, which requires integrating information from multiple documents and thus places higher demands on context-aware compression capabilities. Here, our method yields at least a 5.4 point improvement over other sentence-level compression methods. Beyond accuracy, AttnComp achieves a 17x compression rate, outperforming all other evaluated extractive methods, and significantly reduces the RAG system's end-to-end latency to 49% of the uncompressed baseline.

Beyond its primary role in compression, AttnComp also offers a valuable capability for estimating the confidence of RAG responses by leveraging the attention assigned to the instruction (see Sec. 4.4 for details). After training, the attention allocated to the instruction correlates with the quality of retrieved documents, serving as an indicator of answer reliability. Experiments show a strong positive correlation between the confidence score and actual answer accuracy, enabling users to assess response trustworthiness and mitigate risks from low-quality retrieval. Furthermore, this capability also suggests a possible avenue for future research on autonomous iterative RAG(Asai et al., 2023; Su et al., 2024; Yu et al., 2024).

In summary, our contributions are as follows:

1. We propose AttnComp, a novel extractive compression framework for RAG that is adaptive, efficient, and context-aware.

2. Our method enables confidence estimation for RAG responses, allowing users assess reliability and mitigate risks from low-quality retrieval.

3. Extensive experiments show that AttnComp outperforms existing compression methods and uncompressed retrieval baselines, delivering higher accuracy and lower end-to-end latency.

## 2 Related Work

**Context Compression.** The compress methods can be broadly categorized into abstractive and extractive approaches. For abstractive compression, RECOMP-abs (Xu et al., 2023a) trains a T5-based model to summarize the retrieved content. Zhu et al. (2024) leverage the Information Bottleneck principle to train LLMs for summarization. CompAct (Yoon et al., 2024) employs LLMs to summarize retrieved passages and introduces an iterative strategy that progressively updates the relevant context as new passages are incorporated. For extractive compression, RECOMP-ext (Xu et al., 2023a) performs sentence-level semantic matching by selecting the top-k sentences whose embeddings are most similar to the query. LongLLMLingua(Jiang et al., 2024) proposes a perplexity-based metric to assess the relevance between context and question. A critical limitation of these methods is their dependence on fixed compression ratios. To allow more flexible and adaptive compression, EXIT (Hwang et al., 2024) employs LLMs to conduct binary relevance classification for each sentence, enabling adaptive context reduction. Provence (Chirkova et al., 2025) trains a lightweight DeBERTa model(He et al., 2021) to predict sentence-level relevance scores and retains the sentences that exceed a predefined threshold.

**Confidence Estimation.** Estimating model confidence helps mitigate the risk of unreliable outputs from LLMs (Geng et al., 2024). Logit-based methods evaluate sentence-level uncertainty using token-level probabilities or entropy (Huang et al., 2023; Kuhn et al., 2023). Consistency-based methods estimate confidence by measuring the agreement across multiple generations (Manakul et al., 2023). However, these approaches focus solely on confidence estimation based on internal knowledge, without considering the integration of external knowledge under the retrieval-augmented generation (RAG) paradigm. Chen et al. (2024) highlight two key latent factors influencing confidence in RAG: the quality of the retrieved content and the manner in which it is incorporated into the generation process. To the best of our knowledge, there are currently no methods that estimate confidence in RAG outputs by explicitly evaluating retrieval quality.

## 3 Observations

In this section, we present our observations on the attention patterns within LLMs. Our analysis is conducted on QA datasets, where inputs are constructed by concatenating the context before the query. We then compute the attention score from the query to different context segments as follows:

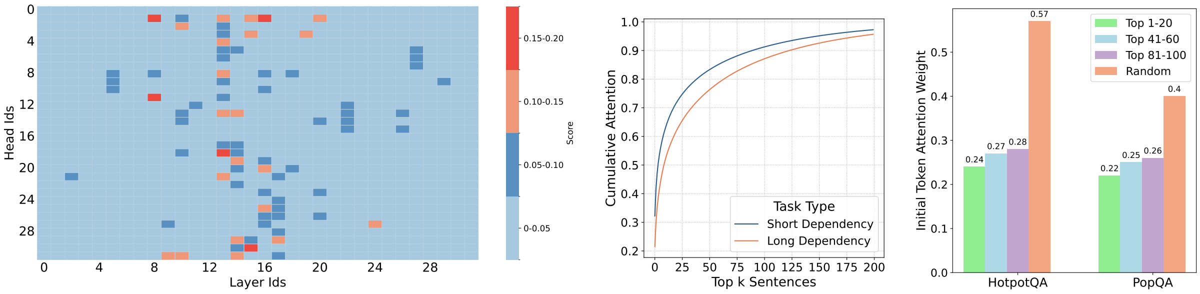
$$s = \frac{1}{|\mathcal{I}_q|} \sum_{i \in \mathcal{I}_q} \sum_{j \in \mathcal{I}_d} a_{ij} \tag{1}$$

where $\mathcal{I}_q$ and $\mathcal{I}_d$ denote the token indices of the query and context segment, respectively, and $a_{ij}$ is the attention weight from query token $i$ to context token $j$.

We present the experimental details in Appendix A. The key findings are summarized below:

- **Certain middle-layer attention heads effectively identify relevant information.** Using the LooGLE benchmark (Li et al., 2024a), a QA dataset with labeled evidence sentences, we analyze attention score assigned by each head to the evidence. Figure 2(a) visualizes the attention scores from each head in every LLM layer assigns to evidence sentences. It is observed that some attention heads in the middle layers consistently focus more on supporting evidence, suggesting their ability to capture relevance.

- **Attention pattern adapts to the density of relevant content.** The LooGLE benchmark divides tasks into short- and long-dependency types. Short-dependency tasks rely on a single sentence or paragraph, while long-dependency tasks require integrating information across multiple segments. We compute the cumulative attention score over the top-$k$ sentences with the highest attention for both task types. Figure 2(b) illustrates how the cumulative attention




(a) (b) (c)

Figure 2: Observations on attention allocation patterns. (a) Attention weights assigned by each attention head to the supporting evidence sentences. (b) Cumulative attention over the top-$k$ most attended sentences in short-dependency and long-dependency tasks. (c) Attention weights on the initial token under different context settings: top 1–20 retrieved documents, top 41–60 documents, top 81–100 documents, and 20 randomly sampled documents.

score changes as the sentence count $k$ varies. As shown, attention is more concentrated in short-dependency tasks and more spread out in long-dependency tasks. Figure 7 provides a more intuitive comparison.

- **Attention to the initial token of the context increases as context becomes less relevant.** To investigate how attention is allocated when the context is irrelevant, we sample questions from HotpotQA (Yang et al., 2018) and PopQA (Mallen et al., 2023), and construct context settings with varying levels of relevance. We then record the attention assigned to the initial token across these settings. As shown in Figure 2(c), the attention on the initial token increases as the relevance of the context decreases, consistent with prior findings on attention sinks (Xiao et al., 2023).

## 4 AttnComp

Inspired by our observations, we propose a novel compression framework *AttnComp*. In this section, we provide a comprehensive explanation of the framework.

**Problem Formulation** Given a query $q$, a RAG system retrieves a set of $k$ documents $D = \{d_1, d_2, \ldots, d_k\}$. A language model $M$ then generates an output $y$ conditioned on the retrieved documents and the query, i.e., $M(y \mid D, q)$. Our objective is to filter irrelevant documents from $D$, yielding a reduced subset $D' \subseteq D$ such that the size of $D'$ is minimized while maintaining or even improving the quality of generated answer $M(y \mid D', q)$.

### 4.1 Attention Computation

Building on the finding that attention heads in middle layers of LLMs identify relevant information, our compressor model comprises the first $L$ transformer layers from the original LLM, followed by an additional cross-attention layer. We first construct the context by prefixing a predefined instruction to the concatenated retrieved documents. The context and query are then concatenated and input into the model. After processing through the first $L$ layers, we obtain the hidden states $X_c \in \mathbb{R}^{n \times d_{\text{model}}}$ and $X_q \in \mathbb{R}^{m \times d_{\text{model}}}$ for the context and the query, respectively, where $n$ and $m$ denote their lengths, and $d_{\text{model}}$ is the hidden dimension. The cross-attention layer then computes query-context attention weights $A \in \mathbb{R}^{m \times n}$ as follows:

$$Q_i = X_q \cdot W_i^Q, \quad K_i = X_c \cdot W_i^K,$$
$$A = \frac{1}{H} \sum_{i=1}^{H} \text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_a}}\right) \tag{2}$$

where $H$ denotes the number of attention heads, $W_i^Q, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_a}$ are the query and key projection matrices for head $i$, and $d_a$ is the dimensionality of each attention head.

### 4.2 Top-P Compression

Motivated by our finding that the attention mechanism exhibits adaptive patterns across varying questions and contexts (as shown in Figures 2(b) and 2(c)), we propose a Top-P compression algorithm that leverages the computed query-context attention weights for adaptive context reduction.

The process commences by calculating attention scores for the instruction ($s_{\text{ins}}$) and each document ($s_{d_i}$), derived from aggregating attention weights $A$, as defined in Equation 1. These scores are then utilized to dynamically select critical documents. Initially, documents are sorted in descending order of their scores, thereby prioritizing candidates with higher attention. A cumulative sum, initialized with $s_{\text{ins}}$, is subsequently accumulated by incrementally adding the scores of these sorted documents. The selection process continues until either the cumulative score exceeds a predefined threshold $p$, or the current document's score is below a minimum threshold $\epsilon$. Algorithm 1 provides the pseudo-code for this procedure.

This strategy enables adaptive behavior: when many relevant documents disperse attention, more documents are required for their cumulative attention to reach the threshold $p$. Conversely, if relevant documents are few and attention is concentrated, a smaller subset is sufficient. If all documents are irrelevant, attention focused solely on the instruction can reach the threshold, filtering out all documents.

### 4.3 Attention Fine-Tuning

Since certain attention heads can inherently focus on relevant context, we initialize the cross-attention layer using selected attention heads from layer $L+1$ of the LLM. However, empirical results show that the untrained compressor still assign relatively high attention to irrelevant segments, particularly when all documents are irrelevant. To improve relevance discrimination, we fine-tune the model while freezing the first $L$ layers and updating only the cross-attention layer. This lightweight approach updates approximately 0.5% of the total parameters, reducing training cost while preserving generalization.

**Data Construction** We prepare training data where each instance comprises a query $q$, retrieved documents $D = \{d_1, \ldots, d_k\}$, and binary relevance labels $R = \{r_1, \ldots, r_k\}$, with each $r_i \in \{0, 1\}$ indicating the relevance of $d_i$ to $q$. Upon examining existing QA datasets, we observe that many contain incomplete relevance annotations, with only a small subset of relevant documents labeled. Directly training on such data yields sub-optimal performance, while manual annotation is resource-intensive. To address this, we propose an automated annotation pipeline based on question-answer pairs, comprising two stages: labeling and verification. In the labeling stage, we use an untrained compressor to perform multiple rounds of Top-P compression with different document permutations. Documents that are consistently retained across all rounds are labeled as relevant, while the rest are considered irrelevant. In the verification stage, the query and the labeled relevant documents are provided to an LLM to generate an answer. The annotation is accepted only if the generated answer is correct; otherwise, it is discarded. Furthermore, to enrich our training data, we also construct negative instances where all retrieved documents are irrelevant to the query. A detailed description of this annotation pipeline is provided in Appendix C.

**Training** Our training objective incorporates two complementary forms of supervision: document-level and instruction-level.

*Document-level Supervision*: This component enhances discrimination between relevant and irrelevant documents through binary cross-entropy:

$$L_{doc} = -\sum_{i=1}^{k} [r_i \log s_{d_i} + (1 - r_i) \log(1 - s_{d_i})] \tag{3}$$

*Instruction-level Supervision*: This component directs attention on the instruction if no documents are relevant, and suppresses it otherwise:

$$L_{ins} = -\Big[r_{\text{ins}} \log s_{\text{ins}} + (1 - r_{\text{ins}}) \log(1 - s_{\text{ins}})\Big] \tag{4}$$

where $r_{\text{ins}} \triangleq \mathbb{I}\big(\sum_{i=1}^{k} r_i = 0\big)$ indicates whether none of the retrieved documents are relevant, with $\mathbb{I}(\cdot)$ representing the indicator function.

The final objective combines both components with a balancing hyperparameter $\lambda$:

$$L = L_{doc} + \lambda L_{ins} \tag{5}$$

### 4.4 Confidence Estimation

The fine-tuned model tends to pay more attention to the instruction when the overall relevance of retrieved content is low. We leverage this behavior by using the instruction attention score $s_{\text{ins}}$ as a proxy for retrieval quality. Specifically, a higher $s_{\text{ins}}$ suggests that the retrieved content is less relevant to the query. In such cases, the LLM relies more on its internal knowledge, which can lead to less reliable responses. Motivated by this insight, we define the confidence score $p$ of a RAG response as:

$$p = 1 - s_{\text{ins}} \tag{6}$$

## 5 Experiments

### 5.1 Experimental Setup

**Implementation Details** We use Llama-3.1-8B-Instruct(Grattafiori et al., 2024) as the backbone architecture for AttnComp, retaining $L = 13$ transformer layers and $H = 16$ attention heads in the cross-attention layer. To train AttnComp, we construct a training dataset from the HotpotQA training

| Methods | HotpotQA | | | 2WikiMQA | | | MuSiQue | | | NQ | | | PopQA | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Comp. | F1 | Acc | Comp. | F1 | Acc | Comp. | F1 | Acc | Comp. | F1 | Acc | Comp. | F1 | Acc |
| ***No Retrieval*** | | | | | | | | | | | | | | | |
| Direct | - | 26.5 | 23.6 | - | 26.2 | 34.1 | - | 11.4 | 8.8 | - | 27.1 | 23.6 | - | 24.1 | 31.3 |
| ***Retrieval without Compression*** | | | | | | | | | | | | | | | |
| All Documents | 1x | 46.3 | 42.7 | 1x | 31.9 | 34.7 | 1x | 19.3 | 15.5 | 1x | **49.9** | 53.9 | 1x | **43.9** | 64.7 |
| Top 5 Documents | 18.2x | 40.4 | 37.9 | 18.3x | 25.7 | 30.4 | 18.4x | 16.5 | 13.9 | 18.3x | 49.0 | 54.8 | 18.4x | 38.1 | 60.9 |
| Top 10 Documents | 9.6x | 42.6 | 40.0 | 9.6x | 28.7 | 31.8 | 9.6x | 18.4 | 15.5 | 9.6x | 48.3 | **55.5** | 9.6x | 39.9 | 64.4 |
| ***Retrieval with Compression*** | | | | | | | | | | | | | | | |
| RECOMP-ext | 8.0x | 40.4 | 37.5 | 8.0x | 27.5 | 30.1 | 8.1x | 18.6 | 14.5 | 8.4x | 47.5 | 48.7 | 9.0x | 31.3 | 51.8 |
| LongLLMLingua | 9.7x | 42.5 | 39.1 | 9.7x | 30.2 | 31.9 | 9.7x | 17.2 | 13.8 | 9.7x | 42.6 | 48.1 | 9.7x | 40.1 | 62.1 |
| CompAct | 80.0x | 45.1 | 40.2 | 82.4x | 29.2 | 33.2 | 71.8x | 18.0 | 16.5 | 84.2x | 47.5 | 48.7 | 98.0x | 39.8 | 58.4 |
| Provence | 10.2x | 42.5 | 39.8 | 10.7x | 26.8 | 29.3 | 8.7x | 19.8 | 17.8 | 6.8x | 41.9 | 50.3 | 6.9x | 34.5 | 58.7 |
| AttnComp (w/o SFT) | 14.1x | 45.5 | 42.5 | 17.0x | 29.7 | 32.4 | 13.8x | 20.9 | 19.5 | 16.1x | 49.1 | 54.8 | 24.0x | 39.8 | 62.3 |
| AttnComp (Ours) | 12.6x | **48.3** | **45.2** | 18.4x | **32.9** | **38.1** | 16.3x | **21.4** | **19.6** | 13.5x | 48.0 | 53.0 | 23.9x | 41.3 | **65.1** |

Table 1: Main results. We use LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024) as the reader model and retrieve 100 documents for each query. Since our training data includes only a subset of HotpotQA, we perform zero-shot evaluation on the remaining datasets. Comp. denotes the compression rate, calculated as: $\frac{\text{\# of tokens in retrieved documents}}{\text{\# of tokens in compressed text}}$.

split, consisting of 8,000 examples. Each example includes a question and 100 documents. For 2,000 of these examples, all documents are irrelevant to the question. We train the model with the Adam optimizer(Kingma, 2014), using a learning rate of $2 \times 10^{-4}$ and a batch size of 8 for 8 epochs. The balancing coefficient $\lambda$ is set to 0.8. During inference, we apply the Top-P Compression algorithm with a threshold of $p = 0.95$ and $\epsilon = 10^{-2}$. Further information is provided in Appendix D.

**Datasets and Retrieval Corpus** We evaluate AttnComp on both single-hop and multi-hop question answering (QA) benchmarks. For single-hop QA, we use Natural Questions (NQ)(Kwiatkowski et al., 2019) and PopQA(Mallen et al., 2023). For multi-hop QA, we evaluate on HotpotQA (Yang et al., 2018), 2WikiMultiHopQA(Ho et al., 2020) and MuSiQue(Trivedi et al., 2022). Following Jin et al. (2024b), we use the Wikipedia dump from December 2018 as the retrieval corpus(Karpukhin et al., 2020), where articles are truncated into non-overlapping documents of 100 words each. For each query, we retrieve the top 100 documents using the E5-base-v2 retriever (Wang et al., 2022).

## 5.2 Baseline

We evaluate AttnComp against several baseline methods. To ensure a fair comparison, all baselines employ Llama-3.1-8B-Instruct(Grattafiori et al., 2024) as the reader model for answer generation, while results using other reader models are presented in Appendix G. The baselines are as follows: (1) *No Retrieval*: The reader model generates answers directly from the input query, without any retrieved context. (2) *Retrieval without Compression*: All retrieved documents are concatenated and fed to the reader model, serving as an uncompressed baseline. For a more fine-grained comparison, we also report results using only the top-5 and top-10 retrieved documents. (3) *Compression Methods*: We compare AttnComp against four compression methods: RECOMP-ext (Xu et al., 2023a), LongLLMLingua (Jiang et al., 2024), CompAct (Yoon et al., 2024) and Provence (Chirkova et al., 2025). Additionally, we also compare against AttnComp without fine-tuning. Detailed descriptions of these baselines are provided in Appendix E.

## 5.3 Main Results

We evaluate the performance of AttnComp using three metrics: compression rate (Comp.), F1 score, and accuracy (Acc), with the results presented in Table 1. The results demonstrate that, even without fine-tuning, AttnComp consistently outperforms all compression baselines across all benchmarks in terms of both F1 score and accuracy, while achieving a high compression rate. Moreover, after fine-tuning, AttnComp further extends its advantage, yielding an average accuracy improvement of 1.9 points over the uncompressed baseline. Notably, AttnComp is the only evaluated method that enhances accuracy, whereas all other compression baselines lead to a decrease of at least 3 points. Furthermore, our method maintains a 17x compression rate, which is higher than that of Provence (8.7x), another adaptive extractive compression method.

# 6 Analysis

We evaluate AttnComp for its adaptiveness (Sec. 6.1), efficiency (Sec. 6.2), context-awareness (Sec. 6.3), and robustness (Sec. 6.4). We also present an

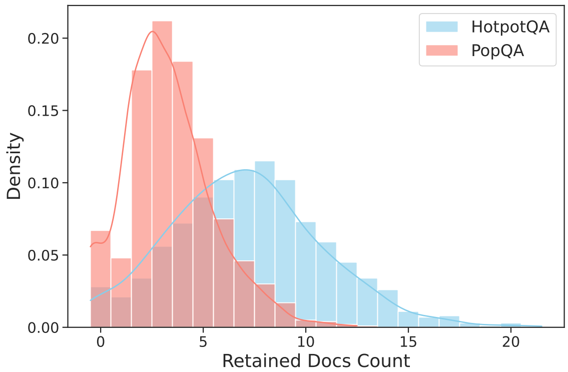



Figure 3: Distribution of documents retained by AttnComp on HotpotQA and PopQA.

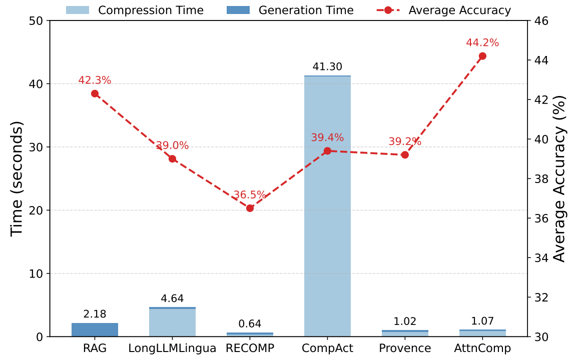



Figure 4: Comparison of end-to-end latency, and average accuracy across baselines and AttnComp.

ablation study (Sec. 6.5) and validate the reliability of its confidence estimation (Sec. 6.6).

### 6.1 Adaptive Compression Analysis

To validate the adaptive compression capability of AttnComp, we analyze the number of documents retained after compression on both the HotpotQA and PopQA datasets. As illustrated in Figure 3, the quantity of retained documents varied dynamically, ranging from 0 to 23. For the multi-hop QA dataset HotpotQA, our model tends to preserve a greater number of documents, averaging 7.5 per query. Conversely, on the simpler PopQA dataset, the number of retained documents was considerably smaller, with an average of 3.7. These results demonstrate that our method can dynamically adjust the compression rate based on the retrieval context and the complexity of the question.

### 6.2 Efficiency Analysis

We evaluate the end-to-end latency of the RAG system, including both the compression and generation stages, to demonstrate the efficiency of AttnComp. All methods are tested under the same hardware conditions: one NVIDIA RTX 4090 GPU for compression and two for generation. We report average compression and generation times, excluding retrieval latency as its impact is negligible.

As illustrated in Figure 4, although most compression methods significantly decrease generation time, the compression stage itself introduces considerable latency that cannot be overlooked. For example, while methods like CompAct achieve high compression rates (up to 80x), their reliance on multiple LLM calls during compression incurs substantial latency. This leads to an overall latency (41.30s) that markedly exceeds the uncompressed baseline (2.18s). In contrast, extractive compression methods such as RECOMP and Provence offer lower latency but at the cost of degraded performance. AttnComp achieves efficiency comparable to Provence while delivering better accuracy. With an average compression latency of 0.91 seconds and a generation latency of 0.16 seconds, AttnComp reduces the total end-to-end latency to 49% of the uncompressed baseline while simultaneously improving answer quality.

### 6.3 Context-Aware Compression Analysis

On multi-hop datasets requiring the integration of information from multiple documents, AttnComp exhibits particularly notable improvements, achieving an average accuracy increase of 3.3 points over the uncompressed baseline. Furthermore, on the 2WikiMultiHopQA dataset, it surpasses the sentence-level compression method Provence by a significant 8.8 points in accuracy. This underscores the context-aware capabilities of our approach. A case study is provided in Appendix H to demonstrate AttnComp's context-aware capability.

### 6.4 Robustness Analysis

We evaluate AttnComp across various settings, including different numbers of retrieved documents, top-p thresholds, and context granularities, to demonstrate its effectiveness in diverse scenarios. Experimental details are provided in Appendix F, and the main conclusions are as follows:

**Varying Number of Retrieved Documents**: We conduct experiments by varying the number of retrieved documents $k$. As shown in Figure 8(a), our approach consistently achieves accuracy comparable to or superior to the uncompressed baseline across different values of $k$. Notably, the superiority of our approach over the baseline becomes more substantial as $k$ increases.

| Arch | w/o Fine-tuning | | Fine-tuning | |
|---|---|---|---|---|
| | Acc | Comp. | Acc | Comp. |
| 7 Layers | 12.4 | 37.68x | 41.8 | 7.8x |
| 13 Layers* | **44.2** | 6.2x | **44.2** | 17.0x |
| 14 Layers | 44.1 | 10.4x | 44.0 | 17.4x |
| 23 Layers | 26.2 | 214.8x | 43.5 | 7.5x |
| 31 Layers | 26.1 | 216.2x | 41.8 | 5.8x |

Table 2: Comparison of accuracy and compression rate across layers and training settings. Default settings are marked with "*".

**Varying Top-p Thresholds**: We evaluate AttnComp with different top-p compression thresholds. As depicted in Figure 8(b), the value of $p$ serves as a parameter to balance accuracy against compression rate. Our findings indicate that AttnComp consistently delivers stable and strong performance when $p$ is set to 0.9 or higher.

**Varying Context Granularities**: We evaluate AttnComp beyond its standard document-level compression by assessing sentence-level compression performance. Results in Table 3 show that sentence-level compression maintained comparable accuracy to document-level, while achieving a superior compression rate. Additionally, visualizing the attention distribution revealed that, despite being trained with document-level annotations, the model effectively focuses attention on relevant sentences and words, demonstrating its adaptability to different context granularities.

### 6.5 Ablation Study

To investigate the impact of layer selections and training strategies, we conduct comprehensive ablation studies across varying layer depths (7, 14, 23, and 31), comparing them against our primary $L = 13$ layer setup. Table 2 presents the comparative results in terms of accuracy and compression rate. Our analysis reveals two key findings: (1) Without fine-tuning, only the middle layer configuration ($L = 13, 14$) achieves optimal accuracy, while others ($L = 7, 23, 31$) perform significantly worse, supporting our hypothesis that middle layers naturally develop effective filtering mechanisms during pretraining; (2) Supervised fine-tuning substantially improves accuary and compression rate across all layer configurations, demonstrating the effectiveness of our training approach. It also indicates that the hidden states of LLMs retain rich linguistic information, which can be effectively leveraged for downstream tasks.

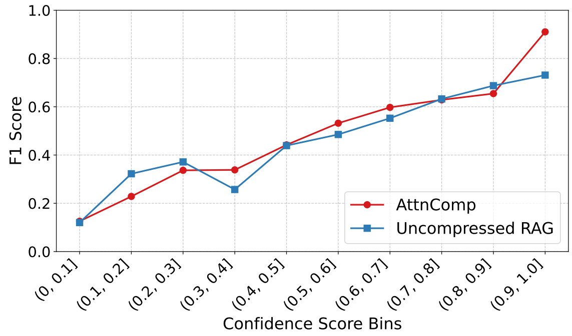



Figure 5: Average F1 score of AttnComp and Uncompressed RAG across confidence score bins on HotpotQA.

### 6.6 Reliability of Confidence Estimates

To assess the reliability of our method's confidence estimates, we compute a confidence score via Equation 6 for each test instance in the HotpotQA dataset. We then stratify these instances into ten decile groups based on their confidence scores. For each bin, we calculate the average F1 score of responses generated by both the AttnComp method and the uncompressed baseline. As shown in Figure 5, the results demonstrate a clear positive correlation between confidence and average F1 score for both methods. Instances with confidence scores below 0.1 yield an average F1 score of just 0.13, while those with confidence scores above 0.9 achieve a substantially higher F1 score of 0.91. Further supporting this observation, the Pearson correlation coefficient between confidence and F1 score is 0.35 for AttnComp and 0.32 for the uncompressed baseline, confirming the utility of the confidence scores as an indicator of RAG response reliability.

We believe the confidence score can be valuable for future work on iterative RAG systems. By leveraging confidence estimates, we can assess the sufficiency of the current retrieval and set the conditions for further iterations. We leave the full exploration of such an iterative framework to future work.

## 7 Conclusion

We introduce AttnComp, a novel framework that leverages the attention mechanism to adaptively compress retrieved documents. Additionally, AttnComp provides a confidence estimation capability for evaluating RAG responses. Extensive experiments demonstrate that AttnComp outperforms existing compression methods and uncompressed baseline, offering higher accuracy with significant compression rates and lower end-to-end latency.

## Limitations

Our study has several limitations. First, all observations and experiments are conducted on LLMs with up to 8 billion parameters due to computational constraints, and we do not evaluate the effectiveness of our method on larger models. Investigating AttnComp's performance across a broader range of model sizes may yield valuable insights. Second, the focus of this work is on the attention mechanisms of dense model architectures, leaving the applicability of our approach to other architectures, such as Mixture-of-Experts (MoE) models, unexplored. Third, our automated annotation strategies relied on Llama-3.1-8B-Instruct for data validation. Given the potential for hallucinations in LLMs, some errors may still exist in the constructed dataset. Finally, although the quality of retrieved content is critical for answer generation in RAG systems, other factors—such as the inherent parameter knowledge in the LLM and the way it integrates retrieved information—also affect response quality(Chen et al., 2024). Our proposed confidence estimation method focuses solely on the quality of retrieved documents, which may lead to inaccurate assessments when other influential factors are at play.

## References


Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, and 1 others. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Lu Chen, Ruqing Zhang, Jiafeng Guo, Yixing Fan, and Xueqi Cheng. 2024. Controlling risk of retrieval-augmented generation: A counterfactual prompting framework. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2380–2393.

Nadezhda Chirkova, Thibault Formal, Vassilina Nikoulina, and Stéphane Clinchant. 2025. Provence: efficient and robust context pruning for retrieval-augmented generation. *arXiv preprint arXiv:2501.16214*.

Yu Fu, Zefan Cai, Abedelkadir Asi, Wayne Xiong, Yue Dong, and Wen Xiao. 2024. Not all heads matter: A head-level kv cache compression method with integrated retrieval and reasoning. *arXiv preprint arXiv:2410.19258*.

Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. 2024. A survey of confidence estimation and calibration in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6577–6595.

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

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.

Yuheng Huang, Jiayang Song, Zhijie Wang, Shengming Zhao, Huaming Chen, Felix Juefei-Xu, and Lei Ma. 2023. Look before you leap: An exploratory study of uncertainty measurement for large language models. *arXiv preprint arXiv:2307.10236*.

Taeho Hwang, Sukmin Cho, Soyeong Jeong, Hoyun Song, SeungYoon Han, and Jong C Park. 2024. Exit: Context-aware extractive compression for enhancing retrieval-augmented generation. *arXiv preprint arXiv:2412.12559*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.

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

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677.

Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan O Arik. 2024a. Long-context llms meet rag: Overcoming challenges for long inputs in rag. *arXiv preprint arXiv:2410.05983*.

Jiajie Jin, Yutao Zhu, Guanting Dong, Yuyao Zhang, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, Zhicheng Dou, and Ji-Rong Wen. 2024b. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. *arXiv preprint arXiv:2405.13576*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2024a. Loogle: Can long-context language models understand long contexts? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16304–16333.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. 2024b. Snapkv: Llm knows what you are looking for before generation. *Advances in Neural Information Processing Systems*, 37:22947–22970.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Artsiom Sauchuk, James Thorne, Alon Halevy, Nicola Tonellotto, and Fabrizio Silvestri. 2022. On the role of relevance in natural language processing tasks. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1785–1789.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. Dragin: Dynamic retrieval augmented generation based on the information needs of large language models. In *62nd Annual Meeting of the Association for Computational Linguistics, ACL 2024*, pages 12991–13013. Association for Computational Linguistics (ACL).

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Siye Wu, Jian Xie, Jiangjie Chen, Tinghui Zhu, Kai Zhang, and Yanghua Xiao. 2024a. How easily do irrelevant inputs skew the responses of large language models? *arXiv preprint arXiv:2404.03302*.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2024b. Retrieval head mechanistically explains long-context factuality. *arXiv preprint arXiv:2404.15574*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023a. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *arXiv preprint arXiv:2310.04408*.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2023b. Retrieval meets long context large language models. In *The Twelfth International Conference on Learning Representations*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 45 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, and 1 others. 2025. Qwen2. 5-1m technical report. *arXiv preprint arXiv:2501.15383*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024. Compact: Compressing retrieved documents actively for question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21424–21439.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context. In *The Twelfth International Conference on Learning Representations*.

Tian Yu, Shaolei Zhang, and Yang Feng. 2024. Auto-rag: Autonomous retrieval-augmented generation for large language models. *arXiv preprint arXiv:2411.19443*.

Kun Zhu, Xiaocheng Feng, Xiyuan Du, Yuxuan Gu, Weijiang Yu, Haotian Wang, Qianglong Chen, Zheng Chu, Jingchang Chen, and Bing Qin. 2024. An information bottleneck perspective for effective noise filtering on retrieval-augmented generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1044–1069.

# A Detailed Observations

In this section, we present detailed observations of attention behavior in LLMs, along with the corresponding experimental procedures. Through three carefully designed experiments, we arrive at the following key findings:

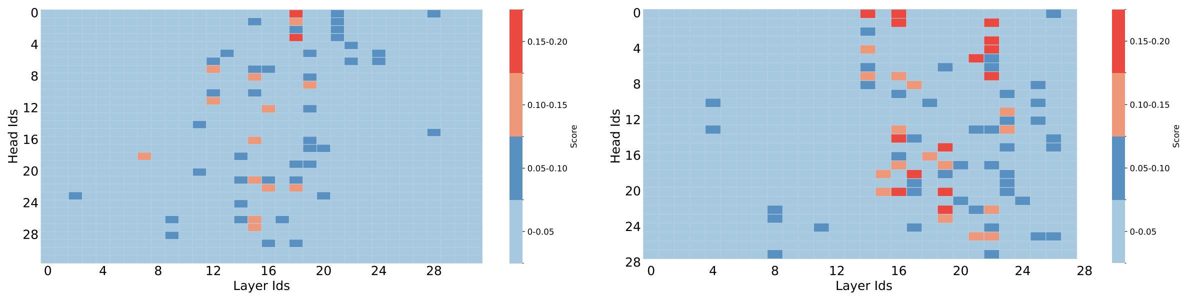
1. Certain middle-layer attention heads effectively identify relevant information. (Section A.1)
2. Attention patterns adapt to the density of relevant content. (Section A.2)
3. Attention to the initial token of the context increases as the overall context becomes less relevant. (Section A.3)

## A.1 Attention Heads Capture Relevance

Prior research has revealed that LLMs exhibit retrieval heads capable of focusing on task-relevant information during text generation (Wu et al., 2024b; Fu et al., 2024). However, these studies primarily focus on copy-and-paste behaviors occurring during the generation phase of LLMs. Other work indicates that LLMs' attention mechanisms can identify relevant information in context before generation, yet these analyses often lack granularity—such as attention-head-level insights into how relevance is determined(Li et al., 2024b; Wu et al., 2024b). Therefore, we address the following research question: *How do LLMs leverage their attention mechanisms to identify question-relevant information before text generation?*

**Experiment** We conduct our analysis using the LooGLE benchmark (Li et al., 2024a), a long-context QA dataset in which each instance comprises an article, a question, and labeled supporting evidence sentences. The concatenated input of the article and question is processed by three models—Llama-3.1-8B-Instruct(Grattafiori et al., 2024), Mistral-7B-Instruct-v0.2(Jiang et al., 2023), and Qwen2-7B-Instruct(Yang et al., 2024)—to compute attention weights across all layers and heads. Using Equation 1, we quantify the attention each head allocates to the supporting evidence sentences. Higher scores indicate stronger focus on question-relevant information.

**Results & Insights** We visualize the attention scores assigned by each head to the supporting evidence sentences. As shown in Figure 2(a) for Llama-3.1-8B-Instruct, and Figure 6 for Mistral-7B-v0.2 and Qwen2-7B-Instruct, these models consistently exhibit certain middle-layer attention heads that assign noticeably higher attention to relevant evidence. In contrast, attention heads in lower and upper layers tend to show weaker focus. These findings suggest that middle-layer attention heads in LLMs are particularly effective at capturing question-relevant information within the




(a) Mistral-7B-v0.2 (b) Qwen2-7B-Instruct

Figure 6: Attention weights assigned by each attention head to supporting evidence sentences are illustrated for two LLMs: Mistral-7B-v0.2 (Figure a) and Qwen2-7B-Instruct (Figure b).

context.

### A.2 Adaptive Attention Patterns

The relevance of information within a context varies depending on the question and task, motivating our investigation into the model's cognitive flexibility: *Does the proportion of relevant information in the context trigger distinct attention allocation strategies in LLMs?*

**Experiment** The LooGLE benchmark categorizes tasks into two types: short-dependency and long-dependency. Short-dependency tasks can be answered using a single sentence or paragraph, while long-dependency tasks require integrating information spread across multiple sentences within the article. In this experiment, we focus on 16 attention heads from the 14th layer of the Llama-3.1-8B-Instruct model. We sample test cases from each task category, ensuring that the input documents have a similar average number of sentences. For each sentence, we compute the attention scores and analyze the proportion of attention allocated to the top-$k$ ranked sentences.

**Results & Insights** Figure 2(b) presents the experimental results. For short-dependency tasks, the attention distribution is more concentrated—only a few sentences receive a disproportionately high share of attention. In contrast, attention is more evenly distributed across sentences in long-dependency tasks. Using a cumulative attention threshold of 0.8 as a reference point, we find that, on average, the top 39 sentences account for 80% of total attention in short-dependency cases, whereas long-dependency tasks require the top 63 sentences to reach the same threshold. This indicates that the model adapts its attention distribution based on the proportion and dispersion of relevant information in the context. To provide a more intuitive illustration, we sample a representative example from each task and visualize the corresponding attention distribution over the context, as shown in Figure 7.

### A.3 Attention on Initial Token

In practical applications of RAG, retrieved content may sometimes be entirely irrelevant to the given question. This leads to a key research question: *How do LLMs allocate attention when the retrieved context is completely irrelevant to the question?*

**Experiment** Inspired by prior work on attention sinks (Xiao et al., 2023), which reveals that initial tokens often collect significant attention scores. We hypothesize that the attention allocated to initial token increases with decreasing context relevance. To test this, we sample questions from HotpotQA (Yang et al., 2018) and PopQA (Mallen et al., 2023), and construct four context settings: (1) top 1–20 retrieved documents, (2) top 41–60 documents, (3) top 81–100 documents, and (4) 20 randomly sampled documents from the 2018 Wikipedia corpus (Karpukhin et al., 2020). For each question, we pair it with these different context sets and measure the attention scores allocated to the initial token. The analysis is conducted using 16 attention heads from the 14th layer of the Llama-3.1-8B-Instruct model.

**Results & Insights** The results are shown in Figure 2(c). We observe a consistent increase in attention scores for the initial token as context relevance decreases. Across different retrieved document sets, the attention to the initial token remains relatively stable. However, a substantial rise is observed when completely irrelevant documents (i.e., random samples) are used as context. These findings suggest that attention on initial token may serve as a useful signal for estimating the relevance of retrieved content.

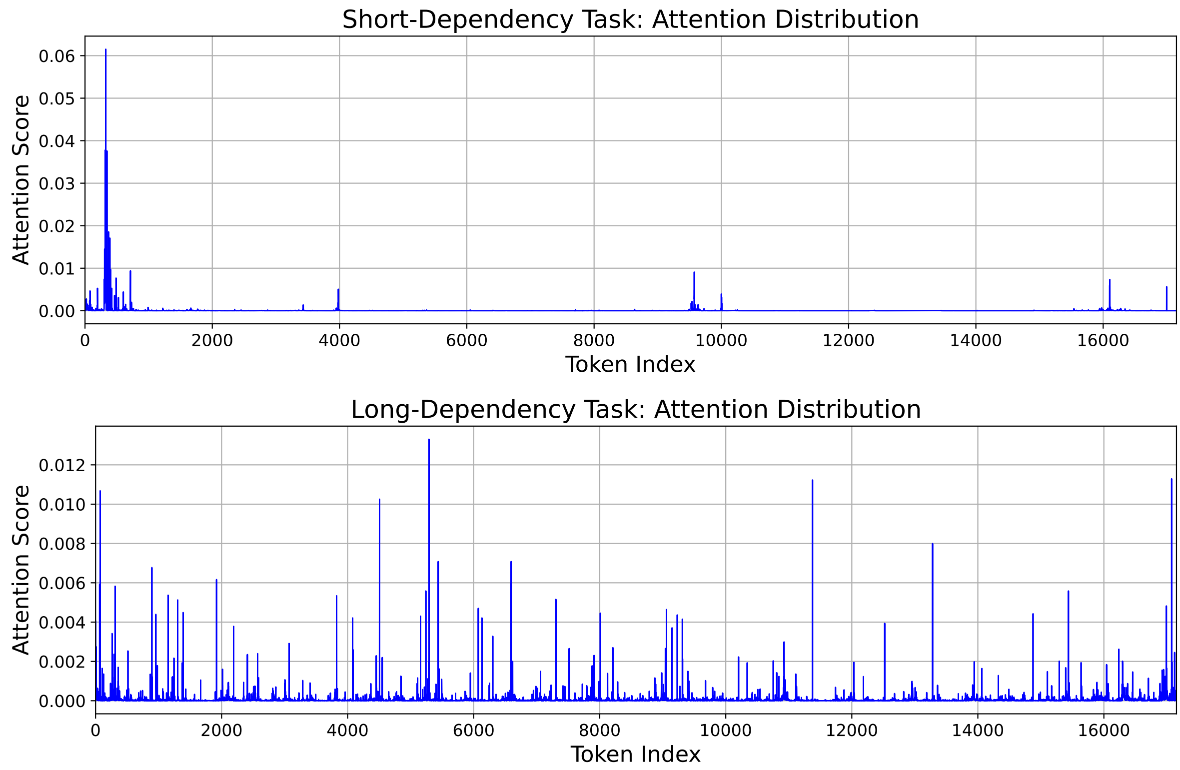



Figure 7: Examples of attention distribution for short- and long-dependency tasks. Attention is more concentrated for short-dependency tasks (top), while it is more dispersed across the input for long-dependency tasks (bottom).

**Algorithm 1** Top-P Compression Algorithm

**Input:** Instruction score $s_{\text{ins}}$, document scores $\{s_{d_1}, s_{d_2}, \ldots, s_{d_k}\}$, top-p threshold $p$, and minimum score threshold $\epsilon$.
**Output:** Compressed document set $D'$.
$\{d_{(1)}, \ldots, d_{(k)}\} \leftarrow \text{argsort}(\{s_{d_i}\}_{i=1}^{k}, \text{desc.})$
Initialize $sum \leftarrow s_{\text{ins}}$, $D' \leftarrow \emptyset$.
**for** $i = 1$ to $k$ **do**
  **if** $sum \geq p$ **or** $s_{d_{(i)}} < \epsilon$ **then**
    **break**
  **end if**
  $sum \leftarrow sum + s_{d_{(i)}}$
  $D' \leftarrow D' \cup \{d_{(i)}\}$
**end for**
**Return:** $D'$.

## B Pseudo-code for Top-P Compression

The pseudo-code for Top-P Compression is shown in Algorithm 1.

## C Details of Data Construction

This section introduces an automated annotation pipeline based on question-answer pairs, with the detailed procedure outlined in Algorithm 2. Given a query and a set of retrieved documents and the corresponding answer, our method utilizes an untrained document compressor to identify relevant documents. However, since such compressors are sensitive to the input document order and may assign high attention to irrelevant content, we perform multiple rounds of compression with different permutations of the document order. Only documents consistently retained across all rounds are labeled as relevant, while the rest are considered irrelevant. The specific model and number of iterations used in our final experiments are detailed in Appendix D.

In each round, the model iteratively applies the Top-P compression algorithm with a high threshold (e.g., $p = 0.95$), continuing until no further reduction in document count is possible, thereby minimizing the impact of individual compression errors.

To verify the correctness of the annotated relevant documents, we feed the query and the selected relevant documents to an LLM to generate an answer. If the generated answer matches the ground truth, the annotation is accepted. If there is a mismatch, we task the LLM with generating an answer using the full set of retrieved documents. If using all retrieved documents yields the correct answer,

**Algorithm 2** Relevance Annotation

**Input:** Compressor $M_c$, Generator $M_g$, Corpus $C$, Top-P threshold $p$, number of shuffles $N$, input tuple$(q, D, a)$ where $q$ is query, $D$ is retrieved documents, $a$ is ground truth answer.
**Output:** Data Sample $(q, D^+, D^-)$ where $q$ is query, $D^+$ is relevant documents, $D^-$ is irrelevant documents.
**for** $i = 1$ to $N$ **do**
  Shuffle documents: $D_i \leftarrow \text{permute}(D)$
  **while** True **do**
    Compute scores: $\{s_d\} \leftarrow M_c(q, D_i)$
    $D'_i \leftarrow$ Top-P Compression$(\{s_d\}, p)$
    **if** $|D'_i| = |D_i|$ **then**
      **break**
    **end if**
    $D_i \leftarrow D'_i$
  **end while**
**end for**
Get relevant docs: $D^+ \leftarrow \bigcap_{i=1}^{N} D_i$
Get irrelevant docs: $D^- \leftarrow D \setminus D^+$
Generate answer: $a' \leftarrow M_g(q, D^+)$
**if** Acc$(a', a) = 1$ **then**
  **Return:** $(q, D^+, D^-)$
**else**
  Generate answer: $a'' \leftarrow M_g(q, D)$
  **if** Acc$(a'', a) = 0$ **then**
    $D_{\text{sample}} \leftarrow$ RandomSample$(C, |D^+|)$
    $D^- \leftarrow D^- \cup D_{\text{sample}}$
    **Return:** $(q, \emptyset, D^-)$
  **else**
    **Return:** $\emptyset$  ▷ Discard the sample
  **end if**
**end if**

the annotation derived from the compressed set is deemed faulty and discarded. If even the full set does not lead to a correct answer, we infer that none of the retrieved documents are relevant to the query and use this instance to construct a negative example, where all documents are considered irrelevant. To mitigate potential errors introduced by the LLM in this negative example construction process, we replace the labeled relevant documents with randomly sampled ones and then label the entire document set as irrelevant.

## D Implementation Details

We construct query-document relevance annotations using question–answer pairs from the HotpotQA training set. For each QA pair, 100 documents are initially retrieved. We then apply Algorithm 2, setting number of shuffles set to $N = 3$, and using Llama-3.1-8B-Instruct as the generator $M_g$ to validate the annotations.

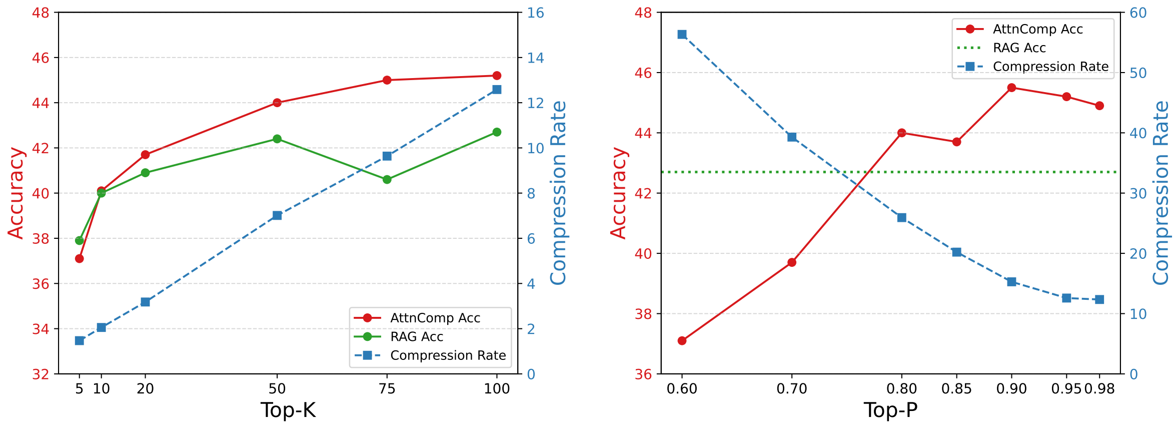
AttnComp is trained on four NVIDIA RTX 4090 GPUs with 24 GB for 4 hours. We use the Adam optimizer with a learning rate of $2 \times 10^{-4}$ and a batch size of 8. The training runs for 8 epochs. We shuffle the input document order in each epoch to mitigate mitigate potential positional bias in the attention mechanism.

## E Baselines Details

The details of the baseline methods are as follows:
(1) RECOMP-ext (Xu et al., 2023a) performs sentence-level semantic matching by selecting the top-k sentences whose embeddings are most similar to the query. We use the model trained on HotpotQA for experimen ts on HotpotQA, 2WikiMultiHopQA, and MuSiQue, and the model trained on NQ for experiments on NQ and PopQA. For all these experiments, we select 50 sentences from documents to ensure a fair comparison at similar text lengths.
(2) LongLLMLingua (Jiang et al., 2024) removes unimportant tokens based on the perplexity scores generated by LLMs. We implement LongLLMLingua using the FlashRAG(Jin et al., 2024b), and set the compression ratio to 10%.
(3) CompAct (Yoon et al., 2024) is an abstractive compression method that leverages LLMs fine-tuned on the HotpotQA dataset to generate summaries of retrieved documents. We use the publicly available implementation and model released by the authors, keeping all configurations consistent with the original setup.
(4) Provence (Chirkova et al., 2025) trains a lightweight DeBERTa model(He et al., 2021) to predict sentence-level relevance scores and retains only the sentences that exceed a predefined threshold. We use the publicly available implementation and model released by the authors, keeping all configurations consistent with the original setup.
(5) AttnComp (w/o SFT) is our proposed method without supervised fine-tuning. We set the threshold $p$ to 0.5 to ensure a fair comparison at a similar compression rate.

## F Details of Robustness Analysis

We assess the model's robustness through experiments that vary the number of retrieved documents,




(a) Top-k Variants Analysis

(b) Top-p Threshold Analysis

Figure 8: Performance of AttnComp with varying top-k and top-p values on HotpotQA.

top-p thresholds, and context granularities. The specific experimental settings and results are detailed as follows:

**Experiment Settings for Number of Retrieved Documents**: We conduct experiments on HotpotQA by varying the number of retrieved documents $k \in 5, 10, 20, 50, 75, 100$, while keeping the top-p threshold $p$ constant at $0.95$. The results are shown in Figure 8(a).

**Experiment Settings on Top-P Threshold**: We conduct experiments on HotpotQA by varying the top-p threshold $p \in 0.1, 0.3, 0.5, 0.7, 0.9, 0.95$, while keeping the number of retrieved documents $k$ constant at 100. The results are shown in Figure 8(b).

**Experiment Settings for Context Granularities**: We evaluate AttnComp by varying the context granularity by varying the context granularity, including document-level and sentence-level compression. For sentence-level compression, we split the retrieved documents into sentences following Provence(Chirkova et al., 2025) and then apply the Top-P compression algorithm with the threshold $p$ set to 0.95, while the minimal score threshold $\epsilon$ is set to $10^{-3}$. The results are shown in Table 3.

## G Additional Results

We conduct experiments using Qwen2.5-7B-Instruct-1M (Yang et al., 2025) as the reader model, which has stronger long-context capabilities. The results are shown in Table 4. Compared to the uncompressed baseline, our method still achieves improvements in accuracy and outperforms other compression baselines.

| Dataset | Document-level | | Sentence-level | |
|---|---|---|---|---|
| | **Comp.** | **Acc** | **Comp.** | **Acc** |
| HotpotQA | 12.6x | 45.2 | 14.5x | 43.2 |
| 2WikiMQA | 18.4x | 38.1 | 21.3x | 34.4 |
| MuSiQue | 16.3x | 19.6 | 18.5x | 20.1 |
| NQ | 13.5x | 53.0 | 16.8x | 51.8 |
| PopQA | 23.9x | 65.1 | 34.3x | 65.9 |
| **Average** | 17.0x | 44.2 | 21.1x | 43.1 |

Table 3: Comparison of compression ratio (Comp.) and answer accuracy (Acc) between document-level and sentence-level granularity across five QA datasets using AttnComp.

## H Case Study

In Table 5, we present a representative example from the HotPotQA dataset. The query is: "*Who was the eldest brother of the Mexican drug trafficker born 12 March 1952?*" Two of retrieved documents provide the necessary evidence. Document A states, "*Benjamín Arellano Félix (born 12 March 1952) is a Mexican drug trafficker*" (see the first document in the AttnComp compressed context), while Document B indicate that, "*Francisco Rafael Arellano Félix is the eldest brother of Benjamín Arellano Félix*" (see the third document in the AttnComp compressed context). Importantly, the relevance of Document B is not evident in isolation, as it requires the contextual link provided by Document A. Without this cross-document connection, Document B is prone to being mistakenly filtered out as irrelevant.

AttnComp addresses this issue by jointly processing all retrieved documents, allowing it to capture semantic dependencies across documents and

| Methods | HotpotQA | | | 2WikiMQA | | | MuSiQue | | | NQ | | | PopQA | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Comp.** | **F1** | **Acc** | **Comp.** | **F1** | **Acc** | **Comp.** | **F1** | **Acc** | **Comp.** | **F1** | **Acc** | **Comp.** | **F1** | **Acc** |
| ***No Retrieval*** | | | | | | | | | | | | | | | |
| Direct | - | 26.5 | 23.6 | - | 26.2 | 34.1 | - | 11.4 | 8.8 | - | 27.1 | 23.6 | - | 24.1 | 31.3 |
| ***Retrieval without Compression*** | | | | | | | | | | | | | | | |
| All Documents | 1x | 46.4 | 42.3 | 1x | 38.5 | **38.8** | 1x | 22.4 | **20.0** | 1x | 42.6 | 51.1 | 1x | 27.5 | **65.0** |
| Top 5 Documents | 18.2x | 42.3 | 36.5 | 18.3x | 35.4 | 33.1 | 18.4x | 17.9 | 15.0 | 18.3x | 48.5 | 51.3 | 18.4x | 37.4 | 60.2 |
| Top 10 Documents | 9.6x | 44.3 | 38.5 | 9.6x | 38.4 | 36.8 | 9.6x | 19.8 | 16.9 | 9.6x | **49.0** | **52.7** | 9.6x | 37.5 | 62.2 |
| ***Retrieval with Compression*** | | | | | | | | | | | | | | | |
| RECOMP-ext | 8.0x | 40.6 | 35.5 | 8.0x | 37.5 | 35.1 | 8.1x | 19.9 | 16.3 | 8.4x | 41.8 | 44.8 | 9.0x | 29.9 | 51.0 |
| LongLLMLingua | 9.7x | 45.1 | 39.5 | 9.7x | 35.7 | 33.2 | 9.7x | 18.2 | 14.1 | 9.7x | 39.0 | 41.5 | 9.7x | 33.4 | 58.6 |
| CompAct | 80.0x | 45.8 | 40.4 | 82.4x | 34.9 | 35.7 | 71.8x | 18.0 | 16.6 | 84.2x | 44.7 | 47.7 | 98.0x | 39.7 | 57.4 |
| Provence | 10.2x | 44.3 | 39.4 | 10.7x | 33.7 | 31.4 | 8.7x | 19.5 | 16.6 | 6.8x | 43.2 | 46.8 | 6.9x | 30.2 | 55.7 |
| AttnComp (Ours) | 12.6x | **50.8** | **45.4** | 18.4x | **40.5** | 38.1 | 16.3x | **23.4** | 19.6 | 13.5x | 48.4 | 50.1 | 23.9x | **39.8** | 63.9 |

Table 4: Results with Qwen2.5-7B-Instruct-1M(Yang et al., 2025) as the reader model; all other experimental settings are kept the same as in the main results.

retain both supporting facts. In contrast, methods such as RECOMP(Xu et al., 2023a), LongLLMLingua(Jiang et al., 2024) and Provence(Chirkova et al., 2025) process each document independently, preventing them from integrating cross-document information and often leading to the erroneous exclusion of relevant content. Although CompAct(Yoon et al., 2024) adopts an iterative integration mechanism, it often halts the iteration prematurely before gathering sufficient evidence, ultimately missing the key facts needed to answer the query.

**Question**: Who was the eldest brother of the Mexican drug trafficker born 12 March 1952?
**Answer:** Francisco Rafael Arellano Félix

---

**Method**: AttnComp (Ours)
**Compressed Context**:
Doc 1(Title: "Benjamín Arellano Félix") Benjamín Arellano Félix (born 12 March 1952) is a Mexican drug trafficker and former leader of the Mexican criminal organization known as the Tijuana Cartel or ""Arellano-Félix Organization"". Benjamín Arellano Félix, who worked closely with his brothers, was one of Mexico's most powerful drug lords and the supplier of one-third of the U.S.'s cocaine. Benjamín had six brothers: He also has four sisters. Two of them, Alicia
...
Doc 3(Title: "Francisco Rafael Arellano Félix") Francisco Rafael Arellano Félix Francisco Rafael Arellano Félix (24 October 1949 – 18 October 2013) was a Mexican drug lord and former leader of the Tijuana Cartel, a drug trafficking organization. He was the oldest of seven brothers and headed the criminal organization early in the 1990s alongside them. Through his brother Benjamín, Francisco Rafael joined the Tijuana Cartel in 1989 following the arrest of Miguel Ángel Félix Gallardo
**Predict**: Francisco Rafael Arellano Félix (**Correct**)

---

**Method**: RECOMP
**Compressed Context**:
(Title: "Eduardo Arellano Félix") Eduardo Arellano Félix Eduardo Arellano Félix (born October 11, 1956) is a Mexican drug trafficker, brother of Benjamín, Ramón, Javier and sister Enedina, all drug traffickers.
...
(Title: "Jorge Luis Ochoa Vásquez") Jorge Luis Ochoa Vásquez Jorge Luis Ochoa Vásquez (September 30, 1950) is a Colombian drug trafficker who was one of the key founding members of the notorious Medellín Cartel in the late 1970s.
...
(Title: "Ramón Arellano Félix") Ramón Arellano Félix Ramon Arellano Félix (August 31, 1964 – February 10, 2002) was a Mexican drug trafficker whom authorities linked to the Tijuana drug cartel
**Predict**: Jorge Luis Ochoa Vásquez (**Wrong**)

---

**Method**: LongLLMLingua
**Compressed Context**:
Doc(Title: "amín Arellano Félix display at Museo del Enervante Mexico City. currently incarcerated at United States Penitentiary Canaan In the217 Netflix andivision series,El Chapo"", Hern Rom Benjam Avendañoa fictionalized portrayal of Benjamín Arellano Félix)
...
Doc 2(Title: "Benjamín Arellano Féli Benjamín Arellano Félix Benjamín Arellano Féli (born 12 March 1952) is a Mexican drug traff and former leader of Mexican criminal organization known as the Tijuana Cartel or ""Arellano-Félix Organization" Benjamín Arellano Féli, who worked closely with his brothers, was one of Mexico's most powerful lords andlier of one-third of the U..'s cocaine. Benjamín had brothers: also has four sisters....
...
He has several aliases, including El 85, Saúl Ulloa Cuevas, Gerardo Sánchez Espinosa, Érick Valencia Cornelio, Ochenta y Cinco, and Mono. His criminal profile says he is tall and weighs
**Predict**: This information is not available in the given documents (**Wrong**)

---

**Method**: CompAct
**Compressed Context**:
Benjamín Arellano Félix, born on 12 March 1952, is a Mexican drug trafficker and former leader of the Mexican criminal organization known as the Tijuana Cartel or the Arellano-Félix Organization. Benjamín had six brothers, including He also has four sisters.
**Predict**: Benjamín Arellano Félix (**Wrong**)

---

**Method**: Provence
**Compressed Context**:
(Title: "Eduardo Arellano Félix") Eduardo Arellano Félix Eduardo Arellano Félix (born October 11, 1956) is a Mexican drug trafficker, brother of Benjamín, Ramón, Javier and sister Enedina, all drug traffickers.
...
Juan David was the elder brother of Jorge Luis and Fabio Ochoa Vásquez, powerful figures inside Born in a small town in the state of Sinaloa, Torres Félix began working for the Sinaloa Cartel in the 1990s and later ascended to the apex of the cartel after his brother Javier Torres Félix was arrested in 2004. He reportedly has five brothers: Nemesio, Juan, Miguel, Marín, and Abraham.
...
(Title: "Enedina Arellano Félix") brother Eduardo Arellano Félix in 2008. Benjamín Arellano Félix, who worked closely with his brothers, was one of Mexico's most powerful drug He formed the Beltrán Leyva Cartel along with his brothers Héctor, Carlos and Arturo.
**Predict**: Juan David Ochoa Vásquez (**Wrong**)

---

Table 5: Case study comparing compressed contexts and answers generated by baseline methods and AttnComp. Relevant content and correct answers are highlighted in green, while misleading content and incorrect answers are highlighted in red.