

DioR: Adaptive Cognitive Detection and Contextual Retrieval Optimization for Dynamic Retrieval-Augmented Generation

Hanghui Guo^{1,2}, Jia Zhu^{1*}, Shimin Di^{3*}, Weijie Shi^{4*}, Zhangze Chen¹, Jiajie Xu⁵

¹Zhejiang Key Laboratory of Intelligent Education Technology and Application, Zhejiang Normal University

²School of Computer Science and Technology, Zhejiang Normal University

³School of Computer Science and Engineering, Southeast University

⁴Department of Computer Science and Engineering, Hong Kong University of Science and Technology

⁵School of Computer Science and Technology, Soochow University

Abstract

Dynamic Retrieval-augmented Generation (RAG) has shown great success in mitigating hallucinations in large language models (LLMs) during generation. However, existing dynamic RAG methods face significant limitations in two key aspects: **1) Lack of an effective mechanism to control retrieval triggers, and 2) Lack of effective scrutiny of retrieval content.** To address these limitations, we propose an innovative dynamic RAG method, DioR (Adaptive Cognitive Detection and Contextual Retrieval Optimization), which consists of two main components: *adaptive cognitive detection* and *contextual retrieval optimization*, specifically designed to determine when retrieval is needed and what to retrieve for LLMs is useful. Experimental results demonstrate that DioR achieves superior performance on all tasks, demonstrating the effectiveness of our work.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities across various generative tasks, such as text creation, dialogue generation, and content summarization (Raffel et al., 2020; Brown et al., 2020; Achiam et al., 2023; Chan et al., 2024). However, LLMs remain inherently static and lack the ability to learn in real-time (Mallen et al., 2023; Xing et al., 2024; Kumar, 2024). As a result, they cannot incorporate up-to-date information, which may generate inaccurate or even fabricated content when encountering unfamiliar scenarios (Maynez et al., 2020). This phenomenon is commonly referred to as “hallucination”.

Retrieval-Augmented Generation (RAG) has gained attention as an innovative approach to reduc-

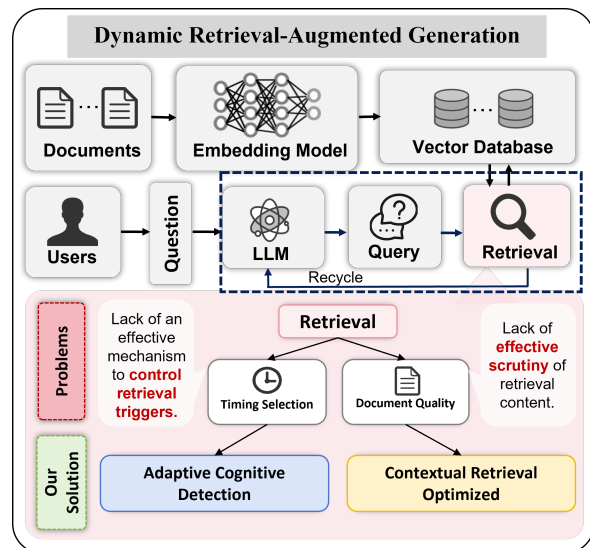


Figure 1: Dynamic RAG and its limitations about “Timing Selection” and “Document Quality”, as well as our solution to address the “Retrieval” limitation.

ing hallucinations in LLM. By integrating external knowledge bases during the generation process and leveraging the contextual learning capabilities of LLMs, RAG effectively minimizes the generation of erroneous information, enhancing the reliability and precision of LLM generation (Jiang et al., 2022, 2024b; Niu et al., 2023; Zhu et al., 2025).

Conventional RAG methods are a single-turn retrieval process, where relevant documents are extracted based on the LLM’s input query and used for content generation (Ye et al., 2024; Roy et al., 2024; Zhang et al., 2024). While effective for simpler tasks, this method struggles with complex tasks or extensive text generation (He et al., 2024). These highlight the need for advanced retrieval mechanisms to handle these complex scenarios.

In contrast, Dynamic RAG effectively alleviates this issue by enhancing the quality of LLM outputs through multi-turn retrieval during the generation process (Tayal and Tyagi, 2024; Yang et al., 2024b; Su et al., 2024a). Specifically, it consists of two

First author.
ghh1125@zjnu.edu.cn

*Corresponding author.
jiazhu@zjnu.edu.cn
shimin.di@seu.edu.cn
wshiah@connect.ust.hk

main steps: 1) Selecting the appropriate retrieval timing, and 2) Constructing the appropriate query once retrieval is triggered. This approach effectively addresses the shortcomings of conventional RAG in handling more complex problems.

However, as shown in Figure 1, existing dynamic RAG methods exhibit significant limitations from two perspectives, specifically:

1. **Lack of an effective mechanism to control retrieval triggers:** Current dynamic RAG strategies typically rely on static, predefined rules to trigger the retrieval module, such as when the generation probability of a token falls below a predefined threshold. However, a low generation probability only indicates low confidence in the token, which does not necessarily imply hallucination after LLM generation text. Moreover, existing RAG strategies often trigger retrieval only after hallucinations occur during the LLM text generation process. If the existence mechanism could predict whether the LLM is capable of answering a question in advance and trigger retrieval accordingly before generating, hallucinations might be avoided when generating.

2. **Lack of effective scrutiny of retrieval content:** The current dynamic RAG method performs single-batch retrieval in each round and also relies on the confidence scores of the most recent sentences' tokens generated by the LLM to determine the final retrieval keyword, without fully considering the overall contextual requirements of the task. This can lead to the retrieval of documents that are not entirely relevant, as well as the introduction of noisy data. In addition, single-batch retrieval often focuses on limited aspects of the context, and the retrieved information has high redundancy, leading to repeated retrievals and increased computational costs. Moreover, retrieving documents with excessively long content can significantly hinder the LLM's ability to understand, leading to information overload and impacting the model's ability to extract key information from documents and perform effective reasoning.

To bridge these gaps, we propose a novel dynamic RAG method, named **DioR** (Adaptive Cognitive Detection and Contextual Retrieval Optimization). DioR consists of two main components: *adaptive cognitive detection*, and *contextual retrieval optimization*. Specifically:

In terms of *adaptive cognitive detection*, we utilize the Wiki dataset to construct two hallucination detection datasets and train two classifiers: Early Detection and Real-time Detection. The Early Detection Classifier assesses the model's ability to answer questions independently, while Real-time Detection monitors for hallucinations during the generation process. Once Early Detection finds LLM has no confidence in its response or Real-time Detection identifies hallucinations, we present *contextual retrieval optimization* to step-by-step retrieve documents. We optimize the priority of query keywords through contextual analysis for retrieval. Retrieved documents are then used to capture new concepts to refine query keywords for more relevant and precise retrieval in later steps. Additionally, we use a sentence-level chunking module to reduce the impact of long texts, enhancing model understanding and inference performance.

The main contributions are listed as follows:

- We propose DioR, a novel dynamic RAG method that addresses two key limitations of existing methods: 1) Lack of an effective mechanism to control retrieval triggers, and 2) Lack of effective scrutiny of retrieval content.
- We constructed and trained both early detection and real-time detection classifiers to determine the optimal retrieval trigger timing based on the internal state of the LLM before and during the text generation process. Additionally, we adopted contextual retrieval optimization to enhance both the retrieval process and the quality of the documents, thereby improving the LLM's reasoning generation capabilities.
- Our experiments demonstrate that DioR surpasses popular methods in four knowledge-intensive generation datasets, achieving superior performance across the board.

2 Related Work

Large language models (LLMs) have shown outstanding performance in solving downstream tasks (Kumar, 2024); however, due to their inability to integrate new information, they are prone to generating hallucinations during generation (Su et al., 2024b; Ramprasad et al., 2024). To address this issue, Retrieval-Augmented Generation (RAG) is an effective strategy, enhancing the model's ability to

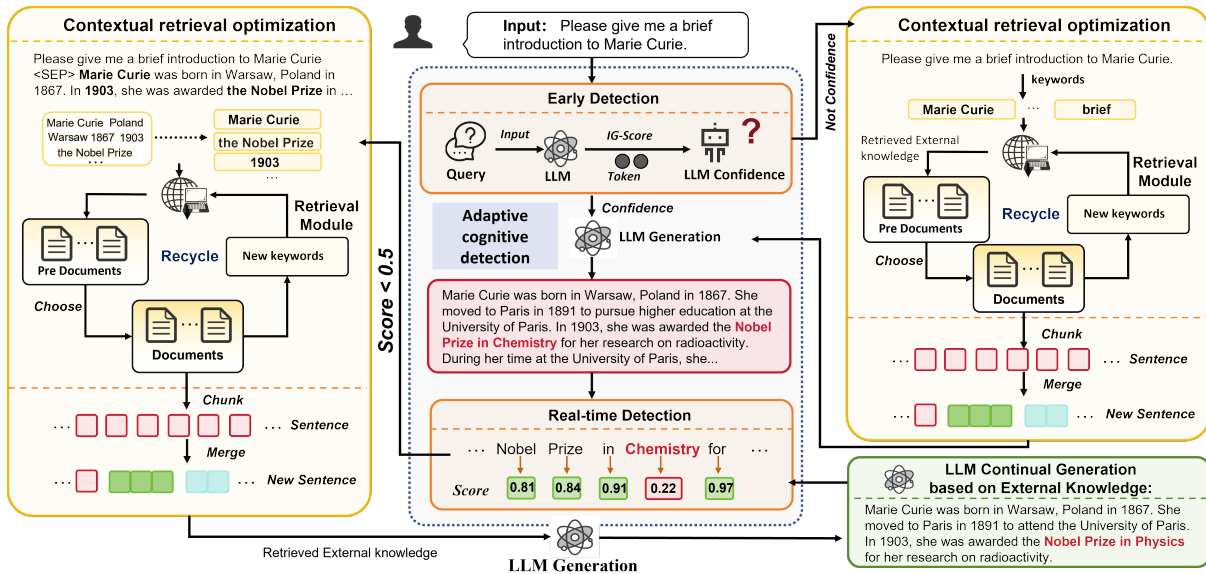


Figure 2: Detailed technical framework of DioR. It is mainly divided into two technical aspects: 1) Adaptive cognitive detection; and 2) Contextual retrieval optimization.

leverage non-parametric knowledge by retrieving external data resources (Jiang et al., 2022, 2024b).

The initial RAG paradigm primarily focused on single-turn retrievals, such as Single-round RAG (SR-RAG) methods like KNN-LM (Wang et al., 2023), ReAtt (Jiang et al., 2022), REPLUG (Shi et al., 2024), and UniWeb (Li et al., 2023), which retrieve relevant passages from external corpora based on the initial query and append them to the LLM’s input. Single-turn retrieval indeed shows significant improvements in the context of simple tasks. However, when confronted with more complex tasks, such as multi-step reasoning, long-form question answering, and chain-of-thought reasoning (Ayala and Béchar, 2024a; Su et al., 2024a), relying solely on the initial input for RAG often fails to provide sufficient effective external knowledge, resulting in limited improvements in addressing hallucinations (Wu et al., 2024).

To address this issue, researchers have explored multi-turn retrieval-augmented strategies (Dynamic RAG). Fixed Length RAG (FL-RAG) methods, such as RETRO (Borgeaud et al., 2022) and ICRLM (Ram et al., 2023), trigger the retrieval module after every n token, using the generated tokens from the previous token window as the query. Fixed Sentence RAG (FS-RAG) methods, such as IRCot (Trivedi et al., 2023), trigger retrieval after each sentence, treating each generated sentence as a new query. However, these multi-turn retrieval approaches do not fully consider the real-time needs of the LLM during the generation process, which

may lead to suboptimal results. In contrast, FLARE (Jiang et al., 2023) triggers retrieval when uncertain tokens are encountered (i.e., when the probability of any token in the text falls below a certain threshold), and inspired by this, Dragin (Su et al., 2024a), a novel dynamic RAG method, determines the timing and content of retrieval based on the LLM’s attention entropy and token relevance.

However, current dynamic RAG methods fail to predict whether the LLM has the capability to answer a question prior to generation, thereby triggering retrieval in advance. Moreover, most methods often rely on static rules, leading to ineffective timing for retrieval triggers during the generation process. Additionally, existing dynamic RAG approaches lack optimization strategies for the document retrieval process and the quality of retrieved documents, such as relevance and length, which significantly hampers the LLM’s performance

3 Proposed Method: DioR

In this section, we present the **DioR**, an innovative dynamic Retrieval-Augmented Generation (RAG) method, as illustrated in Figure 2. DioR consists of two main components: *adaptive cognitive detection* and *contextual retrieval optimization*.

3.1 Adaptive cognitive detection

In view of the limitation of existing dynamic RAG methods that *Lack an effective mechanism to control retrieval triggers*, we propose an innovative approach, **Adaptive cognitive detection**, which aims

to effectively judge the optimal *timing* of retrieval based on the cognitive characteristics of large language models (LLMs). We specifically explain the method from two perspectives.

3.1.1 Is LLM confident in answering this?

We mentioned before that existing dynamic RAG methods all determine whether hallucinations occur after LLM generates content. However, if an LLM exhibits a lack of sufficient confidence when confronted with a particular question before generating, does it mean the model is more likely to generate incorrect information in uncertain situations, potentially leading to hallucinations? Therefore, can we intervene in the model’s output too early, before it generates such hallucinations?

Algorithm 1 Early Detection Dataset Construction and Training

```

1: Input: Wikipedia
2: Output: Trained RNN classifier  $RNN_C$ 
3: Step 1: Data Preparation
4: for each question-answer pair  $Q, R$  do
5:   Generate answer  $A$  using LLaMA2-7B based on question  $Q$ 
6:   if  $A$  matches  $R$  or contains  $R$  (i.e.,  $R \subseteq A$ ) then
7:     Label as Non-Hallucination
8:   else
9:     Label as Hallucination
10:  end if
11:  Extract IG Attribution Entropy (IG features) from question  $Q$ 
12:  Store  $(Q, A, R, \text{IG}, \text{Label})$ 
13: end for
14: Step 2: Model Training
15: Initialize RNN-based classifier  $RNN_C$ 
16: for each data point  $(Q, A, R, \text{IG}, \text{Label})$  do
17:   Feed  $(Q, A, R, \text{IG})$  into the RNN
18:   Compute predicted label  $\hat{y}$  and loss  $\mathcal{L}(\hat{y}, \text{Label})$ 
19:   Backpropagate and update model parameters to minimize  $\mathcal{L}$ 
20: end for
21: Step 3: Output Trained classifier  $RNN_C$ 

```

Therefore, we plan to train a detector to determine whether early intervention in retrieval **timing** is necessary to optimize the subsequent generation process. Specifically, **Early Detection** aims to assess the LLM’s confidence level regarding a given

question before generation, and based on this assessment, decide whether retrieval should occur prior to generation. The specific dataset construction and training process is detailed in Algorithm 1 and Appendix A.1 (Algorithm Explanation).

Based on this, we trained an RNN classifier on a large Wikipedia dataset to determine whether to trigger RAG based on the LLM’s confidence level in responding to a given question.

3.1.2 Does LLM have accurate output?

Algorithm 2 Real-time Detection Dataset Construction and Training

```

1: Input: Wikipedia  $W = \{w_1, w_2, \dots, w_n\}$ 
2: Output: Trained MLP classifier  $MLP_C$ 
3: Step 1: Data Preparation
4: for each article  $w_i \in W$  do
5:   Truncate article based on entity positions
6:   Generate text  $t_i$  using LLaMA2-7B from truncated article
7: end for
8: Step 2: Entity Extraction and Comparison
9: Extract entities  $e_i$  from original article  $w_i$ 
10: Extract entities  $n_i$  from generated text  $t_i$ 
11: for each entity  $n_j \in n_i, e_j \in e_i$  do
12:   Compare  $e_j$  and  $n_j$  for cosine similarity
13:   if similarity between  $e_j$  and  $n_j$  is high then
14:     Consider  $e_j$  and  $n_j$  as the same entity (no hallucination)
15:   else
16:     Mark  $t_i$  as hallucination (0)
17:   end if
18: end for
19: Step 3: Dataset Construction
20: Construct data point  $D_i = \{w_i, e_i, t_i, n_i, H_i\}$ , where  $H_i$  is hallucination label (0 for hallucination, 1 for non-hallucination)
21: Store data point  $D_i$  in dataset  $D_n$ 
22: Step 4: Model Training
23: Train MLP classifier  $MLP_C$  on dataset  $D_n$ 
24: for each data point  $(w_i, e_i, t_i, n_i, H_i) \in D_n$  do
25:   Feed  $w_i, e_i, t_i, n_i$  into the MLP classifier
26:   Compute predicted hallucination label  $\hat{H}_i$  and loss  $\mathcal{L}(\hat{H}_i, H_i)$ 
27:   Backpropagate and update classifier parameters to minimize  $\mathcal{L}$ 
28: end for
29: Step 5: Output Trained classifier  $MLP_C$ 

```

As mentioned earlier, most existing dynamic RAG frameworks rely on static, predefined rules to trigger the retrieval module. For example, retrieval is initiated when the generation probability of a token during the LLM’s output process falls below a predefined threshold, treating such low-probability tokens as potential hallucinations.

However, hallucination detection should not rely on generation probability. While the generation probability could indicate model confidence (Farquhar et al., 2024; Ayala and Béchar, 2024b), it does not directly reflect the accuracy of the LLM-generated text, and therefore may not be a reliable criterion for the occurrence of hallucinations.

Thus, the **Real-time Detection** aims to monitor the generation process of the LLM and detect in real time whether the output tokens from LLMs are likely to be hallucinations, as the standard of the retrieval timing. The specific dataset construction and training process is described in Algorithm 2 and Appendix A.2 in detail.

Therefore, we have trained an MLP-based hallucination detector on a large corpus of Wikipedia data to ensure its effectiveness in accurately identifying hallucinations.

3.2 When to retrieve is needed?

The appropriate retrieval *timing* helps LLM to avoid hallucinations in the generation process with maximum efficiency. So we define the detection process of the above two classifiers as follows:

Definition 1. Let $C(Q)$ denote the LLM’s confidence in question Q . $IG(Q)$ represents the Attribution Entropy (IG features) (more ref. A.1.1).

$$IG(Q) = \sum_{j=1}^N \frac{-IG_j}{\sum_{k=1}^N IG_k} \log \left(\frac{\sum_{k=1}^N IG_k}{IG_j} \right). \quad (1)$$

$$C(Q) = \mathbb{I}(\text{Softmax}(f_{\text{RNN}}(IG(Q))) > 0.5). \quad (2)$$

If $C(Q)$ is 1, the retrieval trigger. We calculate the IG attribution value IG_i separately for each token t_i of Q , and the mean of all token is IG_{mean} . For selecting candidate keywords for retrieval (the model tends to focus on), we following below:

$$\text{If } IG_i > IG_{\text{mean}} \Rightarrow t_i \text{ as candidate.} \quad (3)$$

Definition 2. Let t_j as the generating token of LLMs. P_{t_j} as hallucination probability of t_j .

$$P_{t_j} = \sigma(f_{\text{MLP}}(t_j)). \quad (4)$$

If P_{t_j} (the sigmoid layer score σ) is below 0.5, the token t_j is flagged as a hallucination and the retrieval trigger. Then, use spaCy to extract all entities from the generated text, convert them into tokens, filter out the tokens labeled as hallucinations, and ultimately attain valid candidate keywords.

3.3 Contextual retrieval optimization

Once the LLM is determined to lack confidence in answering a question or when hallucinations are detected, triggering RAG becomes necessary. The next step in the RAG framework is to generate queries and retrieve relevant information from external databases to support the LLM in continuing text generation. However, existing dynamic RAG methods suffer from the limitation of **Lack of effective scrutiny of retrieval content**.

To address the limitation, we propose a novel method called **Contextual retrieval optimization**, consisting of two key steps: pre-retrieval and post-retrieval. Specifically, as follows.

3.3.1 Pre-retrieval

Existing dynamic RAG frameworks rely solely on the token confidence scores (e.g., attention weights) of the most recent sentences generated by the LLM to select the final retrieval keyword. This approach, which selects keywords based on recent sentences or token confidence, struggles to accurately capture subtle semantic relationships between contextual information, potentially leading to suboptimal retrieval results. Therefore, we performed the following optimizations in the selection of keywords.

We analyzed the entire text generated by the LLM using the two detectors (sec. 3.2, Early and Real-time Detection) and identified a set of candidate keywords. Then, we assess the importance of each candidate keyword (token) through four evaluation metrics. This helps the model choose the most important keyword (token) to retrieve.

We first compute the attention scores of each token using a multi-head attention mechanism. By representing attention across different heads, the model can focus on various semantic and syntactic features of the text, offering a more comprehensive understanding of which parts of the text are most important for retrieving external information. Let A_i represent the attention score for token i . Next, we incorporate TF-IDF scores to evaluate the information density of each token. We assign higher priority to tokens with greater information density. We then calculate a positional score based on the

relative position of each token in the text, taking into account its contextual role in text generation. The positional score for token i is computed as: $P_i = \frac{\text{Pos}(i)}{N}$, where $\text{Pos}(i)$ and N are the position of the token and the total number of tokens respectively. Finally, we assess the semantic relevance between each token and the query term by calculating the cosine similarity S_i between the word embedding vectors of the token i and the query \vec{q} .

The overall importance score I_i for each token is computed as a weighted sum of four components, which enables more accurate identification of relevant tokens and enhances subsequent retrieval.

DioR ranks tokens in descending order of importance based on I_i , it prioritizes those most relevant to the retrieval task. These top-ranked tokens facilitate the extraction of the most relevant external knowledge with the question. Advanced retrieval models, such as BM25, are then used to retrieve relevant documents from an external knowledge base, enhancing the precision of the retrieval process. Then, we put the retrieved documents in the candidate area for the next step of optimization.

3.3.2 Post-retrieval

After retrieving relevant documents from the knowledge base based on token priority, we need to design a dynamic optimization strategy to enhance the accuracy and relevance of documents for real-time retrieval needs. Specifically, since current dynamic RAG frameworks retrieve all documents in a single batch per query trigger, lacking flexibility and failing to fully understand the task context, we discard this strategy in favor of stepwise retrieval.

For example, if five documents need to be retrieved, we first choose the top two most relevant from the candidate area ($n/2$, where n is the number of remaining documents to be retrieved). Next, we perform keyword extraction on these two documents, focusing on newly emerging keywords or concepts. The newly emerged concepts are merged with the original keywords, and a new round of retrieval is conducted based on the updated keyword set. This process is repeated until the required number of documents is selected, providing accurate information support for subsequent generation.

Once the document selection is complete, the next step is to input the selected documents into the LLM. If any individual document is too long, it may hinder the model’s understanding and reasoning. To address this, we design a sentence-level segmentation approach for the document. However,

simple sentence segmentation may lead to semantic breaks, potentially causing misinterpretations by the model. To solve this problem, we reassemble the sentences into shorter, semantically coherent segments, ensuring that the split content maintains logical continuity. Specifically, for each sentence x , we first split it into sub-clauses x_1, x_2, \dots, x_n , then evaluate the scores for various sub-clause combinations. For instance, we compare the score of the combination of x_1 and x_2 with that of x_1 alone. If the combination score improves, we proceed by adding x_3 , comparing the score of x_1, x_2, x_3 with that of x_1, x_2 , and repeat this process until the score decreases. At that point, the sub-clauses are grouped into a block. This method enables us to break each document d_i into several semantically coherent and shorter blocks (e.g., $d_{i1}, d_{i2}, \dots, d_{ik}$), mitigating the negative impact of long texts on the model’s comprehension and optimizing its reasoning performance. This ensures that each input is semantically clear and concise.

We integrate all the segmented blocks from the retrieved documents into the LLM prompt template, please refer to the following for details:

External Knowledge After Chunk:

[1] $d_1[(1).d_{11}, (2).d_{12}, \dots, (u).d_{1u}]$

[2] $d_2[(1).d_{21}, (2).d_{22}, \dots, (t).d_{2t}]$

...

[3] $d_i[(1).d_{i1}, (2).d_{i2}, \dots, (k).d_{ik}] \dots$

Using the external knowledge provided above, please answer the following question:

Question: [Ques.]

Answer: Insert truncated output [] and additional relevant details here.

This integration method effectively addresses the knowledge gaps during the LLM generation process. Specifically, when hallucinations occur in the content generated by the LLM, we make it a truncation point, and we introduce externally retrieved knowledge to allow the LLM to continue generating more accurate and comprehensive content from the truncation point. This enhances the model’s understanding of the documents and ensures that the generated content is more precise and relevant.

4 Experiment

4.1 Experimental Setups

The comprehensive and detailed descriptions of the experimental setups, including the *datasets*, *evaluation metrics*, and *implementation specifics*, can be found in Appendix B.

Table 1: Comparison of EM, F1, Precision, and Recall scores between the Base method and DioR across various datasets and retrieval strategies. The highest scores for each metric in each dataset are highlighted in underline.

Matrix	2WikiMultihopQA		HotpotQA		IIRC		StrategyQA	
	Base	DioR	Base	DioR	Base	DioR	Base	DioR
EM (BM25)	0.214	0.254	0.219	0.274	0.156	0.201	0.639	0.659
EM (SGPT)	0.209	0.226	0.202	0.212	0.125	0.178	0.604	0.616
EM (SBERT)	0.231	0.266	0.165	0.205	0.142	0.153	0.645	0.654
F1 (BM25)	0.282	0.335	0.314	0.379	0.188	0.245	0.639	0.659
F1 (SGPT)	0.278	0.292	0.301	0.307	0.153	0.224	0.604	0.616
F1 (SBERT)	0.294	0.328	0.244	0.303	0.172	0.179	0.645	0.654
Pre. (BM25)	0.288	0.342	0.331	0.399	0.195	0.255	0.639	0.659
Pre. (SGPT)	0.284	0.298	0.319	0.333	0.159	0.229	0.604	0.616
Pre. (SBERT)	0.298	0.334	0.256	0.324	0.176	0.185	0.645	0.654
Rec. (BM25)	0.285	0.345	0.316	0.381	0.196	0.243	0.639	0.659
Rec. (SGPT)	0.281	0.299	0.305	0.312	0.156	0.219	0.604	0.616
Rec. (SBERT)	0.299	0.332	0.255	0.303	0.180	0.189	0.645	0.654

Table 2: Comparison of DioR and other RAG methods for LLaMA2-7B-CHAT across four different datasets

Dataset	Metric	DioR	SEAKR	RaDIO	Dragin	wo-RAG	SR-RAG	FL-RAG	FS-RAG	FLARE
2WikiMultihopQA	EM	0.266	0.264	0.254	0.231	0.146	0.169	0.112	0.189	0.143
	F1	0.335	0.330	0.317	0.294	0.223	0.255	0.192	0.265	0.213
HotpotQA	EM	0.274	0.261	0.246	0.219	0.184	0.164	0.146	0.214	0.149
	F1	0.379	0.365	0.351	0.314	0.275	0.150	0.211	0.304	0.221
IIRC	EM	0.201	0.195	0.196	0.156	0.139	0.187	0.172	0.178	0.136
	F1	0.245	0.235	0.239	0.188	0.173	0.226	0.203	0.216	0.164
StrategyQA	Pre.	0.659	0.650	0.654	0.645	0.659	0.645	0.634	0.629	0.627

4.2 Experimental Results

In this subsection, we mainly discuss the most important experimental results, which aim to comprehensively evaluate DioR’s performance across four datasets with various competitors. The additional experimental results, which provide further insights into the performance of our method, are discussed in greater detail in Appendix C.

4.2.1 Overall Results

In this part, we present results comparing the performance of the Base (Dragin (Su et al., 2024a)) and DioR (Using LLaMA2-7B). As shown in Table 1 in detail.

In 2WikiMultihopQA, the DioR method consistently outperforms the Base method across all retrieval methods and evaluation metrics. Specifically, DioR achieves a notable increase in EM, with a score of 0.254 (BM25) compared to the Base’s 0.214. Similarly, F1, Precision, and Recall scores

show advantages for DioR, surpassing the Base.

In HotpotQA. For EM, DioR reaches 0.274 (BM25), an improvement over the Base’s 0.219. The F1 score is also better for DioR, achieving 0.379 (BM25) versus the Base’s 0.314. Pre. and Rec. follow a similar trend. It is worth noting that under the SBERT retrieval strategy, the Pre. is improved by 0.068, the largest increase compared with BM25 and SGPT.

In the IIRC dataset, DioR demonstrates a consistent advantage across metrics. The EM score increases from 0.156 (Base) to 0.201 (DioR, BM25). F1, Pre., and Rec. scores also show significant improvement, with DioR reaching a Pre. of 0.255 (BM25) and Rec. of 0.243 (BM25), higher than the Base method by a noticeable margin.

In the StrategyQA dataset, DioR outperforms the Base method across all retrieval methods. The EM score increases from 0.639 (Base) to 0.659 (DioR, BM25). F1, Pre., and Rec. show similar

Table 3: Ablation experiments. ED: Early-Detection, RD: Real-time Detection, Pre-R/Post -R: Pre/Post -retrieve.

Dataset	Metric	DioR	w/o ED	w/o RD	w/o Pre-R	w/o Post-R
2WikiMultihopQA	EM	0.266	0.258	0.239	0.249	0.260
	F1	0.335	0.327	0.301	0.306	0.322
HotpotQA	EM	0.274	0.266	0.223	0.237	0.249
	F1	0.379	0.363	0.319	0.334	0.356
IIRC	EM	0.201	0.191	0.169	0.172	0.197
	F1	0.245	0.233	0.197	0.206	0.240
StrategyQA	Pre.	0.659	0.652	0.646	0.648	0.655

improvements, with DioR achieving a Precision score of 0.659 and a Recall score of 0.659 (both BM25), marking a consistent performance boost.

Overall, results show that DioR outperforms the Base method across all datasets and evaluation metrics, providing a more robust solution for question-answering tasks and effectively addressing hallucinations while enhancing LLM reasoning.

4.2.2 Comparison with Other RAG methods

In this part, we present a comparison of the performance of DioR and various RAG-based methods (Using LLaMA2-7B). As shown in Table 2.

On the 2WikiMultihopQA dataset, DioR achieves the highest performance. Specifically, DioR scores 0.266 for EM, surpassing some popular dynamic RAG methods, such as SEAKR, RaDIO, and Dragin. Similarly, DioR leads in F1, with a score of 0.335. Other RAG methods, such as wo-RAG (EM: 0.146, F1: 0.223) and SR-RAG (EM: 0.169, F1: 0.255), perform notably worse.

In HotpotQA, DioR also outperforms other methods, achieving higher EM (0.274) and F1 (0.379) scores than popular dynamic RAG methods such as RaDIO (EM: 0.246, F1: 0.351), SEAKR (EM: 0.261, F1: 0.365), and Dragin (EM: 0.231, F1: 0.294). In contrast, wo-RAG and SR-RAG show significantly lower scores (EM: 0.184, F1: 0.275 for wo-RAG; EM: 0.164, F1: 0.150 for SR-RAG).

In the IIRC dataset, DioR achieves the highest EM (0.201) and F1 (0.245) scores, outperforming other RAG methods such as Dragin (EM: 0.156, F1: 0.188), FL-RAG (EM: 0.172, F1: 0.203) and FS-RAG (EM: 0.178, F1: 0.216).

On the StrategyQA dataset, DioR achieves the highest Precision score (0.659), surpassing the RaDIO (0.654), Dragin (0.645), SEAKR (0.650), and other RAG methods. While wo-RAG shows similar performance (0.659) due to the dataset is not

complex, DioR still leads in all datasets, which is indicative of its overall more reliable performance.

Thus, the experimental results demonstrate that DioR offers a robust solution for handling LLM-generated hallucinations, making it a superior choice for tasks requiring high factual accuracy, such as those involving complex, multi-hop reasoning or multi-turn question answering.

4.2.3 Ablation Experiment

In this part, we mainly test the effectiveness of the following two components of our proposed method: adaptive cognitive detection and contextual retrieval optimization. As shown in Table 3.

It is worth noting that if we remove the Real-time Detection module, retrieval is triggered directly when the generation probability of a token is below 0.5. If we remove the “Pre-retrieval” step, retrieval is performed sequentially based on the order of token appearances. The remaining two modules can be directly removed: removing Early Detection eliminates the need to assess the LLM’s ability to answer in advance while removing “Post-retrieval” leads to single-batch retrieval, where all documents are included in the prompt input to the LLM (the prompt for LLMs reference Appendix F).

Based on the results of the ablation experiments, we can observe the effectiveness of each component in enhancing DioR across four datasets. This further validates the efficacy of our proposed method in addressing the two limitations of existing dynamic RAG approaches.

5 Conclusion and Future work

In this paper, we investigate the effectiveness of Retrieval-Augmented Generation (RAG) techniques in mitigating hallucination issues in large language models (LLMs). However, existing dy-

dynamic RAG methods face significant limitations in two key aspects, 1) *Lack of an effective mechanism to control retrieval triggers*, and 2) *Lack of effective scrutiny of retrieval content*. To address these limitations, we propose an innovative dynamic RAG approach, **DioR** (Adaptive Cognitive Detection and Contextual Retrieval Optimization), achieving when retrieval is needed and what to retrieve for LLMs is useful. Compared to existing popular dynamic RAG methods, DioR demonstrates superior performance across four knowledge-intensive generation datasets, proving the effectiveness of our method in improving.

Looking ahead, we will refine DioR, especially in its ability to tackle complex problems in a step-by-step manner. By integrating more refined reasoning strategies, we aim to further elevate the model's performance on intricate tasks.

Limitations

We acknowledge the limitations of this paper. Specifically, we face the performance impact of retrieving long documents for model inference. While we have implemented chunking to shorten the length of individual knowledge pieces and improve the model's understanding of each, the total length of all input knowledge remains unchanged. In the future, we could introduce a model that summarizes the key points of each document, reducing the overall length and focusing on the core content for retrieval, thereby improving both the efficiency and the relevance of the retrieved knowledge.

On the other hand, we have identified that complex problems often pose significant challenges when approached through a single, direct reasoning process. For instance, in mathematical problems, attempting to solve them in one step can lead to increased computational complexity, higher error rates, and difficulty in identifying intermediate issues. To address these challenges, we plan to adopt a step-by-step reasoning approach in the future. Specifically, we will plan to break down complex problems into a series of smaller, more manageable sub-problems. By tackling each sub-problem sequentially, we can enhance the clarity and accuracy of the reasoning process, making it easier to identify and correct errors at each stage, and ultimately improve the overall effectiveness and reliability of large language models' reasoning.

Acknowledgment

This work was supported by the National Key R&D Program of China under Grant No. 2022YFC3303600, the Zhejiang Provincial Natural Science Foundation of China under Grant No. LY23F020010, and the National Natural Science Foundation of China under Grant Nos. 62337001.

References

- Mohamed Abdelsalam, Mojtaba Faramarzi, Shagun Sodhani, and Sarath Chandar. 2021. Iirc: Incremental implicitly-refined classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11038–11047.
- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Orlando Ayala and Patrice B  chard. 2024a. [Reducing hallucination in structured outputs via retrieval-augmented generation](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track, NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 228–238. Association for Computational Linguistics.
- Orlando Ayala and Patrice B  chard. 2024b. [Reducing hallucination in structured outputs via retrieval-augmented generation](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track, NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 228–238. Association for Computational Linguistics.
- Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yike Guo, and Jie Fu. 2024. Rq-rag: Learning to refine queries for retrieval augmented generation. *arXiv preprint arXiv:2404.00610*.
- Tyler A Chang, Katrin Tomanek, Jessica Hoffmann, Nithum Thain, Erin MacMurray van Liemt, Kathleen

- Meier-Hellstern, and Lucas Dixon. 2024. Detecting hallucination and coverage errors in retrieval augmented generation for controversial topics. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4729–4743.
- Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.
- Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.
- Bolei He, Nuo Chen, Xinran He, Lingyong Yan, Zhenkai Wei, Jinchang Luo, and Zhen-Hua Ling. 2024. Retrieving, rethinking and revising: The chain-of-verification can improve retrieval augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10371–10393.
- 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.
- Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2024. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.
- Xuhui Jiang, Yuxing Tian, Fengrui Hua, Chengjin Xu, Yuanzhuo Wang, and Jian Guo. 2024a. A survey on large language model hallucination via a creativity perspective. *arXiv preprint arXiv:2402.06647*.
- Zhengbao Jiang, Luyu Gao, Zhiruo Wang, Jun Araki, Haibo Ding, Jamie Callan, and Graham Neubig. 2022. Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2336–2349.
- Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992.
- Ziyan Jiang, Xueguang Ma, and Wenhua Chen. 2024b. Longrag: Enhancing retrieval-augmented generation with long-context llms. *arXiv preprint arXiv:2406.15319*.
- Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.
- Pranjal Kumar. 2024. Large language models (llms): survey, technical frameworks, and future challenges. *Artificial Intelligence Review*, 57(10):260.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.
- Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jingyuan Wang, Jian-Yun Nie, and Ji-Rong Wen. 2023. The web can be your oyster for improving language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 728–746.
- Ningke Li, Yuekang Li, Yi Liu, Ling Shi, Kailong Wang, and Haoyu Wang. 2024. Drowzee: Metamorphic testing for fact-conflicting hallucination detection in large language models. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA2):1843–1872.
- Yuanhua Lv and ChengXiang Zhai. 2011. Adaptive term frequency normalization for bm25. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1985–1988.
- Alex Troy 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 *The 61st Annual Meeting Of The Association For Computational Linguistics*.
- Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919.
- Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.
- Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, Kashun Shum, Randy Zhong, Juntong Song, and Tong Zhang. 2023. Ragtruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. *arXiv preprint arXiv:2401.00396*.

- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don’t know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789.
- 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.
- Sanjana Ramprasad, Elisa Ferracane, and Zachary C. Lipton. 2024. [Analyzing LLM behavior in dialogue summarization: Unveiling circumstantial hallucination trends](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 12549–12561. Association for Computational Linguistics.
- Nirmal Roy, Leonardo FR Ribeiro, Rexhina Billosmi, and Kevin Small. 2024. Learning when to retrieve, what to rewrite, and how to respond in conversational qa. *arXiv preprint arXiv:2409.15515*.
- Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. Replug: Retrieval-augmented black-box 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 8364–8377.
- Ben Snyder, Marius Moisesescu, and Muhammad Bilal Zafar. 2024. On early detection of hallucinations in factual question answering. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2721–2732.
- Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024a. [DRAGIN: dynamic retrieval augmented generation based on the real-time information needs of large language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 12991–13013. Association for Computational Linguistics.
- Weihang Su, Changyue Wang, Qingyao Ai, Yiran Hu, Zhijing Wu, Yujia Zhou, and Yiqun Liu. 2024b. [Un-supervised real-time hallucination detection based on the internal states of large language models](#). In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 14379–14391. Association for Computational Linguistics.
- Anuja Tayal and Aman Tyagi. 2024. Dynamic contexts for generating suggestion questions in rag based conversational systems. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 1338–1341.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037.
- Bin Wang and C-C Jay Kuo. 2020. Sbert-wk: A sentence embedding method by dissecting bert-based word models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2146–2157.
- Shufan Wang, Yixiao Song, Andrew Drozdov, Aparna Garimella, Varun Manjunatha, and Mohit Iyyer. 2023. knn-lm does not improve open-ended text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15023–15037.
- Yike Wu, Yi Huang, Nan Hu, Yuncheng Hua, Guilin Qi, Jiaoyan Chen, and Jeff Pan. 2024. Cotkr: Chain-of-thought enhanced knowledge rewriting for complex knowledge graph question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3501–3520.
- Mingzhe Xing, Rongkai Zhang, Hui Xue, Qi Chen, Fan Yang, and Zhen Xiao. 2024. Understanding the weakness of large language model agents within a complex android environment. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6061–6072.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024a. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.
- Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, Lei Chen, Nicolas Scheffer, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Scott Yih, and Xin Dong. 2024b. [CRAG - comprehensive RAG benchmark](#). In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

- 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*. Association for Computational Linguistics.
- Zijun Yao, Weijian Qi, Liangming Pan, Shulin Cao, Linmei Hu, Weichuan Liu, Lei Hou, and Juanzi Li. 2024. Seagr: Self-aware knowledge retrieval for adaptive retrieval augmented generation. *arXiv preprint arXiv:2406.19215*.
- Linhao Ye, Zhikai Lei, Jianghao Yin, Qin Chen, Jie Zhou, and Liang He. 2024. Boosting conversational question answering with fine-grained retrieval-augmentation and self-check. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2301–2305.
- Yu Zhang, Kehai Chen, Xuefeng Bai, Zhao Kang, Qianjiang Guo, and Min Zhang. 2024. Question-guided knowledge graph re-scoring and injection for knowledge graph question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8972–8985.
- Jia Zhu, Hanghui Guo, Weijie Shi, Zhangze Chen, and Pasquale De Meo. 2025. Radio: Real-time hallucination detection with contextual index optimized query formulation for dynamic retrieval augmented generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24):26129–26137.

A Adaptive cognitive detection algorithm details

In Section 3, we build two hallucination detection classifiers, as shown in Figure 3. We describe the algorithm described above in detail as follows and the effectiveness analysis of these two classifiers is shown in the Appendix C.6:

A.1 Early Detection

We choose the Wikipedia dataset and used LLaMA2-7B-CHAT for generative question answering. For a randomly selected question Q , we generate an answer A with the model, where each Q contains a ground truth answer R , as described in lines 3 to 5 of Algorithm 1. We consider that if the generated answer A is inconsistent with the factual details of R , then it is factually incorrect, and the model’s response is a hallucination (Jiang et al., 2024a; Ji et al., 2023). This definition aligns with previous generative question-answering studies, which view factually incorrect statements as hallucinations. However, since LLM outputs can be quite long, merely matching the generated answer A with the reference answer R via exact string comparison is insufficient for judging the correctness of A . For example, assume $Q = [\text{Where is the capital of America?}]$ and $R = [\text{Washington}]$. Whether LLaMA generates the answer A as "Washington" or "Washington D.C." both represent correct answers. Therefore, we consider that the reference answer R is contained within the generated answer A , i.e., $R \subseteq A$, and the answer is correct, as described in lines 6 to 10 of Algorithm 1.

To effectively determine whether the LLM has enough information to answer a question, we can assess the model’s focus on certain words within the question, i.e., whether the LLM can capture the key points of the question. We use the Integrated Gradients (IG) method to generate feature attribution, which indicates the importance of each feature in the specific prediction (Snyder et al., 2024). This method is typically used to inspect the behavior of the model and uncover potential problematic patterns. In simple terms, if we calculate the feature attribution based on the LLM’s attention, if the LLM’s output does not generate hallucinations, i.e., if it focuses on certain keywords, the attribution entropy will be low (focusing on keywords, keywords’ attribution is high). Otherwise, when hallucinations are generated, the model will focus on more input words, resulting in higher attribution

entropy (the focus is more dispersed, and all word attribution is relatively even). Specific proof of attribution entropy process reference A.1.1.

Thus, we construct an Early-Detection dataset consisting of Q , A , R , and the IG-generated feature attribution values, along with hallucination labels, as described in lines 11 and 12 of Algorithm 1.

Due to the sequential nature of the data, we use a Recurrent Neural Network (RNN) for training, as RNNs are adept at handling sequential data and effectively capture contextual dependencies within the input sequence, as described in lines 14 to 21 of Algorithm 1. During training, we use the standard binary cross-entropy loss function to predict the \hat{y}_i given Q , A , R , and IG , as follows:

$$L(\theta) = - \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)], \quad (5)$$

where $\hat{y}_i = \text{sigmoid}(f(Q_i, A_i, R_i, IG_i, \theta))$ and $f(Q_i, A_i, R_i, IG_i, \theta)$ is the model's output, calculated from the model parameters θ and the inputs Q_i, A_i, R_i, IG_i .

A.1.1 Proof of Attribution Entropy Behavior

Uniform distribution of attribution values: If the model pays almost equal attention to each input feature (for example, the attribution value of each feature is similar), the entropy value will be high because the model as a whole does not tend to favor a specific feature, but pays attention to multiple features. In this case, the system is more uncertain or chaotic, so the attribution entropy is large.

Concentrated attribution values: If the model's attention is mainly focused on a few features, while the attribution values of other features are small or close to zero (for example, the attribution values of some features are significantly higher than other features), the entropy value will be small because the model's attention is not scattered, but concentrated on a few features, which reduces the uncertainty of the system, so the attribution entropy is small.

The following proves the attribution entropy, specifically:

1. Definition of Attribution Entropy:

The attribution entropy $H(P)$ of a probability distribution $P = \{p_1, p_2, \dots, p_n\}$ is defined as:

$$H(P) = - \sum_{i=1}^n p_i \log p_i, \quad (6)$$

where $\sum_{i=1}^n p_i = 1$.

2. Uniform Distribution (Hallucination):

For a uniform distribution, where each probability $p_i = \frac{1}{n}$, the attribution entropy becomes:

$$H(P) = - \sum_{i=1}^n \frac{1}{n} \log \frac{1}{n} = \log n \quad (7)$$

3. Concentrated Distribution (Non-Hallucination):

Now, consider a distribution where most of the probability mass is concentrated on one element. For example, let $p_1 = 1 - \epsilon$ and $p_2 = p_3 = \dots = p_n = \frac{\epsilon}{n-1}$, where $\epsilon \rightarrow 0$. The attribution entropy is:

$$H(P) = - ((1 - \epsilon) \log(1 - \epsilon)) - (n - 1) \times \frac{\epsilon}{n - 1} \log \frac{\epsilon}{n - 1} \quad (8)$$

When ϵ is small, we can approximate $\log(1 - \epsilon) \approx -\epsilon$. Thus:

$$H(P) \approx - ((1 - \epsilon)(-\epsilon)) - (n - 1) \times \frac{\epsilon}{n - 1} \log \frac{\epsilon}{n - 1} \quad (9)$$

This simplifies to:

$$H(P) \approx \epsilon - \epsilon^2 - \epsilon \log \frac{\epsilon}{n - 1} \quad (10)$$

As $\epsilon \rightarrow 0$, $H(P) \rightarrow 0$, indicating that the attribution entropy is minimized for a concentrated distribution.

A.2 Real-Time Detection

Specifically, we consider that the part of the LLM output most prone to hallucinations typically involves changes in entities (e.g., incorrect dates or names) (Huang et al., 2024; Li et al., 2024; Chang et al., 2024). Therefore, effectively identifying the semantic differences of entities is an effective solution for detecting hallucinations. Based on this, we select the Wikipedia dataset, denoted as W ,

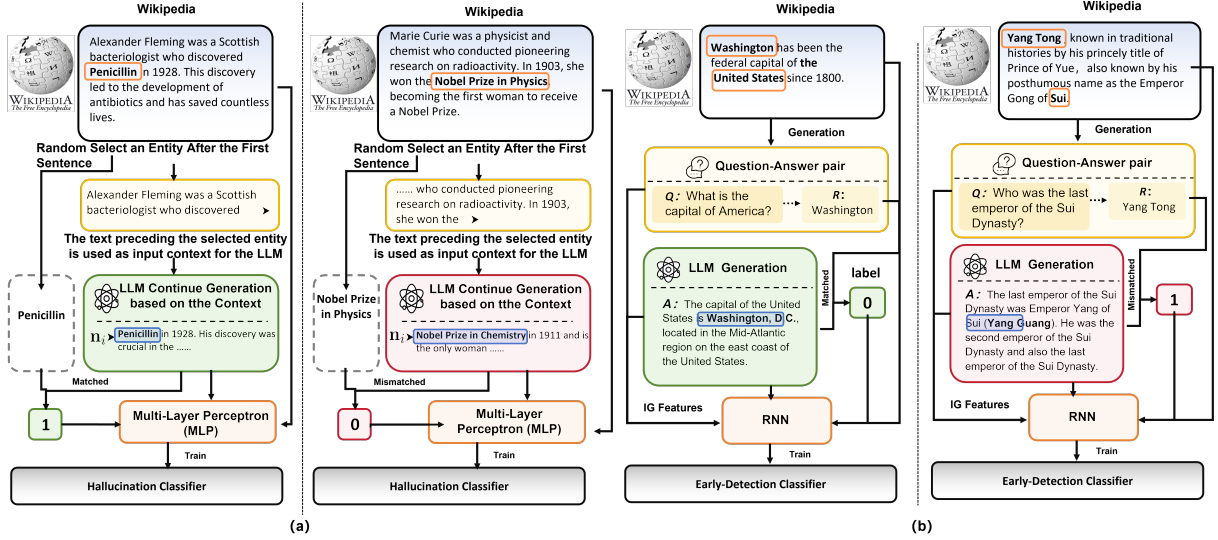


Figure 3: The construction process of Real-time Detection (a) and Early Detection (b) datasets

which contains several independent article paragraphs $\{w_1, w_2, \dots, w_n\}$. From each article w_i , we extract key entities (such as names, years, numbers, and other important terms), and truncate each article w_i based on the positions of the extracted entities e_i (excluding the beginning of each sentence). The truncated document is then fed into the LLM with the prompt: "Below is the opening sentence from a Wikipedia article titled [Title]. Please continue the passage from where the sentence ends. [First sentence of that article]." We choose LLaMA2-7b-CHAT to generate a new piece of text based on the prompt, which we define as t_i , as shown in lines 1-7 of Algorithm 2.

To detect whether the newly generated text contains hallucinations, we perform entity extraction on the same truncated portion of t_i and the original text w_i , resulting in n_i . We then compare the extracted entities n_i with the original entities e_i to determine if a hallucination has occurred. However, entities often exist in multiple forms, such as abbreviations, variations in naming conventions, or different word orders. Therefore, simple direct matching may not capture all entity-level errors. To address this, we combine manual evaluation with the cosine similarity comparison between the vector representations of the extracted entities and the originally selected entities. In this way, we can assess whether two entities are semantically equivalent. If the entities are semantically similar (e.g., "USA" and "United States of America" are considered the same), a hallucination is not detected. Otherwise, the LLM-generated text is labeled as a

hallucination. This is specifically described in lines 8-18 of step 2 in Algorithm 2.

Based on the above steps, we construct a real-time hallucination detection dataset D_i , which includes $(w_i, e_i, t_i, n_i, H_i)$, where H_i represents the hallucination label, with 0 indicating a hallucination and 1 indicating no hallucination. For D_i , we train a multi-layer perceptron (MLP) model.

$$P = \text{MLP}(D_i \{w_i, e_i, t_i, n_i, H_i\} + b), \quad (11)$$

where b is the bias vector.

A.3 How many training samples are generated from Wikipedia?

In this subsection, we mainly explain how much data is constructed to train these two classifiers: Early Detection and Real-time Detection.

A.3.1 The number of training samples of Early Detection

We used LLaMA2-7B to construct 5201 data from Wikipedia, with a training set and a test set ratio of 0.8:0.2, to train and validate Early Detection. We use an NVIDIA A100 GPU, which processes for about half an hour.

A.3.2 The number of training samples of Real-time Detection

We constructed the training set for Real-time Detection from Wikipedia by means of the LLaMA2-7B model. Our dataset comprises 6,000 training samples, 1,000 validation samples, and 1,304 test

samples. We use an NVIDIA A100 GPU, which processes for about one hour.

B Experimental Setups

B.1 Datasets

We evaluate our approach using four multi-hop benchmark datasets and three single-hop benchmark datasets, each designed to assess different aspects of model reasoning and performance:

a) 2WikiMultihopQA (Ho et al., 2020): This dataset is specifically designed to test a model’s ability to perform chain-of-thought (CoT) reasoning. It challenges the model to generate answers by integrating information from multiple Wikipedia articles, requiring the model to navigate through several hops of information to synthesize a final response. It has 1000 examples.

b) HotpotQA (Yang et al., 2018): Similar to 2WikiMultihopQA, this dataset focuses on questions that necessitate information retrieval from multiple documents. It evaluates the model’s capacity to engage in CoT reasoning, requiring it to identify and connect different pieces of evidence scattered across diverse passages to formulate a coherent and accurate answer. It has 1000 examples.

c) IIRC (Abdelsalam et al., 2021): The IIRC dataset is aimed at evaluating reading comprehension and advanced synthesis. Questions in this dataset require the model to integrate information from multiple documents, often spanning several passages, and demand a deep level of synthesis to derive accurate answers from complex, interconnected pieces of text. It has 954 examples.

d) StrategyQA (Geva et al., 2021): This dataset is designed to assess commonsense reasoning by posing questions that involve implicit strategies. To answer these questions, models must generate CoT reasoning steps and leverage abstract thinking, making it a strong test for evaluating a model’s commonsense knowledge and its ability to navigate reasoning processes that are not explicitly stated. It has 1000 examples.

e) NaturalQuestions (Kwiatkowski et al., 2019): NaturalQuestions is a large-scale open-domain question answering dataset constructed from real user queries issued to Google Search. Each question is paired with a Wikipedia page, and annotated with both long answers (typically paragraphs) and short answers (typically named entities or phrases). The dataset is challenging due to the natural, free-form nature of the questions and the need for deep

contextual understanding, making it a strong benchmark for evaluating information retrieval and reading comprehension models.

f) TriviaQA (Joshi et al., 2017): TriviaQA is a question answering dataset collected from trivia websites, featuring naturally occurring questions written by trivia enthusiasts. Each question is associated with multiple evidence documents retrieved from Wikipedia and the web, with answers annotated within these documents. The dataset covers a broad range of topics and requires models to perform reasoning across long and sometimes noisy contexts, providing a robust test for reading comprehension in real-world settings.

g) SQuAD (Rajpurkar et al., 2018): The Stanford Question Answering Dataset (SQuAD) is one of the most widely used benchmarks in QA research. In version 2.0, it combines answerable and unanswerable questions, requiring models to either extract a precise span from a given Wikipedia paragraph or determine that no answer is present. SQuAD emphasizes fine-grained language understanding, span-level prediction, and the ability to distinguish between answerable and unanswerable contexts.

B.2 Evaluation Metrics

We employed a comprehensive set of metrics to evaluate the performance of DioR and other RAG methods on different datasets and retrieval methods. These metrics can be broadly categorized into two groups: *effectiveness metrics* and *efficiency metrics*. Specifically as follows:

Effectiveness metrics aimed to measure the accuracy and quality of the generated answers. We evaluated the models using Exact Match (EM), F1 Score, Precision (Pre.), and Recall (Rec.). The EM metric measures the proportion of model-generated answers that exactly match the reference answers, providing a strict assessment of answer accuracy. The F1 Score calculates the harmonic mean of precision and recall, offering a balanced evaluation of answer quality. We assessed the precision and recall of the models to evaluate their ability to produce accurate responses.

Efficiency metrics focused on assessing the computational resources and time complexity associated with each model. We measured the Retrieve Count (Rc) for the number of documents retrieved during the retrieval process, which measures the ability of the model to efficiently gather relevant information. Furthermore, we calculated the Gen-

erate Count (Gc) for the number of times generated answers are produced, reflecting the model’s capacity for generating responses. The Hallucinated Count (Hc) metric assesses the number of hallucinated content occurrences, where the model generates responses that are not grounded in the input text or retrieved documents. Finally, we measured the Token Count (Tc) and Sentence Count (Sc) to evaluate the verbosity and response length of the models. Due to the fact that LLM is generated more than once, multiple RAGs may be performed, resulting in cumulative statistics of Tc and Sc indicators.

B.3 Implementation

We compare our method with seven RAG baselines, including SEAKR (Yao et al., 2024), RaDIO (Zhu et al., 2025), DRAGIN (Su et al., 2024a), wo-RAG (Su et al., 2024a), SR-RAG (Su et al., 2024a), FL-RAG (Borgeaud et al., 2022; Ram et al., 2023), FS-RAG (Trivedi et al., 2023), and FLARE (Jiang et al., 2023; Su et al., 2024a). wo-RAG represents a setting where no retrieval-augmented generation (RAG) is applied. Other methods are already introduced in the related work section. Notably, wo-RAG and SR-RAG are single-round RAG methods, while the remaining techniques employ a multi-round retrieval strategy.

For our experiments, DRAGIN serves as the baseline (we express it as Base), referenced throughout this work. We employ two fine-tuned large language models as the underlying backbone: LLaMA2-7B CHAT (Touvron et al., 2023) and Qwen2.5-7B (Yang et al., 2024a), both optimized for dialogue-oriented tasks.

Three retrieval strategies are employed: BM25 (Lv and Zhai, 2011), SBERT (Wang and Kuo, 2020), and SGPT (Muennighoff, 2022). Specifically as follows,

1) BM25: It is a classic information retrieval algorithm based on the idea of TF-IDF (Word Frequency Inverse Document Frequency), but has been improved to consider factors such as document length. It can quantitatively evaluate the relevance between documents and queries, help determine the most relevant documents, and thus improve retrieval performance.

2) SGPT: It is a semantic search sentence embedding method based on GPT model. It utilizes the powerful semantic understanding ability of the GPT model and proposes two architectures: dual encoder (SGPT-BE) and cross encoder (SGPT-CE).

SGPT-BE generates sentence representations by fine-tuning the bias tensor of the GPT model and using position weighted average pooling.

3) SBERT: It is a variant based on BERT, designed specifically for generating sentence embeddings, mainly used for tasks such as calculating semantic similarity, text clustering, and information retrieval. It optimizes the generation of semantic embeddings, making it more efficient to calculate the similarity between sentence pairs. SBERT solves the problem of low efficiency in semantic similarity calculation of the original BERT model. By introducing sentence embeddings, each sentence only needs to be encoded once, greatly improving efficiency.

The experiments are conducted on NVIDIA A100 80GB. For hyperparameter settings, we set the maximum generated sequence length to 256 tokens, with a retrieval top- k value of 3 and max retrieval times of 5, and selected the top 25 passages to ensure high-quality responses. We used datasets containing 1000 data points each to ensure robust and reliable results. We processed approximately 12,000 samples on four A100 GPUs. The total runtime was around 18 hours, with an average overall response time of 5.4 seconds per sample.

C More Experimental Results

C.1 Efficiency Comparison Experiment

As shown in Table 4, We discuss the effectiveness of DioR from the perspective of efficiency indicators.

DioR demonstrates a more efficient retrieval and generation process. For example, in 2WikiMulti-hopQA, DioR retrieves significantly more documents, reaching 3.006, while reducing hallucinated generations to just 1.003. The same findings can also be observed from SBERT.

In HotpotQA, DioR retrieves more documents with BM25 (3.033 vs. 2.832) but generates fewer responses (Gc 3.037 vs. Base 6.372), leading to more computationally efficient performance.

Additionally, DioR reduces hallucinated content across all retrieval methods. In 2WikiMulti-hopQA (BM25), DioR’s Hc is 1.003, much lower than Base’s 1.018, reflecting its improved ability to ground responses and minimize irrelevant information. This trend is consistent across datasets, with DioR generally exhibiting fewer hallucinations than the Base method.

In terms of verbosity, DioR produces slightly

Table 4: Compare the efficiency of various retrieval and generation methods in base and DioR across four different datasets.

Matrix	2WikiMultihopQA		HotpotQA		IIRC		StrategyQA	
	Base	DioR	Base	DioR	Base	DioR	Base	DioR
Rc (BM25)	1.018	3.006	2.832	3.033	3.696	3.019	4.422	3.173
Rc (SGPT)	3.602	1.932	3.002	2.053	3.002	2.023	4.305	2.213
Rc (SBERT)	1.804	4.006	0.974	4.060	1.534	4.018	2.390	4.258
Gc (BM25)	2.038	3.006	6.372	3.037	7.431	3.022	8.849	3.206
Gc (SGPT)	7.208	2.021	6.007	2.092	6.009	2.074	8.613	2.280
Gc (SBERT)	3.982	2.033	2.725	2.098	3.206	2.062	5.601	2.206
Hc (BM25)	1.018	1.003	2.832	2.017	3.696	2.010	4.422	2.093
Hc (SGPT)	3.602	1.005	3.002	1.046	3.002	1.035	4.305	1.134
Hc (SBERT)	1.804	1.014	0.974	1.046	1.534	1.030	2.390	1.098
Tc (BM25)	523.763	772.585	694.887	391.962	959.080	776.529	894.252	323.321
Tc (SGPT)	1852.543	519.481	775.129	270.201	775.328	533.008	870.086	229.660
Tc (SBERT)	537.053	522.566	216.537	270.896	497.527	529.970	325.277	222.434
Sc (BM25)	36.530	47.589	34.372	21.491	47.099	42.073	59.893	23.066
Sc (SGPT)	123.930	32.572	40.494	14.887	36.826	29.014	60.191	16.984
Sc (SBERT)	30.908	32.544	10.629	14.575	25.094	29.376	23.592	16.293

longer responses in some cases, such as 2WikiMultihopQA (Tc 772.585 vs. Base 523.763), but this is balanced by its ability to generate more precise answers with fewer sentences (Sc 47.589 vs. 36.530 with BM25). In datasets like StrategyQA, DioR achieves a more concise output (Tc 323.321 vs. 894.252 for Base) without sacrificing accuracy, showing its flexibility in adjusting verbosity based on the task requirements.

DioR outperforms the Base method by retrieving more relevant documents, generating fewer responses, and reducing hallucinations, all while maintaining a balanced verbosity. These improvements result in a more efficient and effective approach to complex question-answering tasks.

C.2 Compare With Single-hop Datasets

As shown in Table 5, we conducted experiments on three single-hop datasets: NaturalQuestions, TriviaQA, and SQuAD. From the results, we observed that DioR achieves certain advantages in terms of EM and F1 scores in most cases. Compared to other dynamic RAG methods such as FLARE, DRIGIN, and RaDIO, our approach consistently demonstrates a certain level of improvement.

Table 5: Performance of various models on NaturalQuestions, TriviaQA, and SQuAD using LLaMA2-7B and BM25. Metrics shown are Exact Match (EM) and F1 scores.

Model	NQ		TriviaQA		SQuAD	
	EM	F1	EM	F1	EM	F1
DioR	26.2	35.9	52.3	61.9	21.5	27.8
CoT	13.4	18.7	42.6	48.6	8.7	13.6
Self-RAG	32.3	40.2	21.2	37.9	5.1	18.3
FLARE	25.3	35.9	51.5	60.3	19.4	28.3
DRIGIN	23.2	33.2	42.0	62.3	18.7	28.7
RaDIO	24.6	34.0	48.3	61.8	20.4	27.5

C.3 Compare with other LLM backbone

We have supplemented our results with experiments using Qwen2.5-7B, comparing performance from both effectiveness and efficiency perspectives. Our results on four knowledge-intensive datasets further validate the effectiveness of DioR. Specifically see Table 6.

C.4 More efficiency ablation study

Our efficiency analysis further demonstrates that, compared to the baseline (DRIGIN), DioR effectively minimizes hallucinations, improves the conciseness of generated content, and optimizes infer-

Table 6: Performance Metrics for Multiple Datasets using Qwen2.5-7B

Metric	2WikiMultihopQA			HotpotQA			IIRC			StrategyQA		
	Base	RaDIO	DioR	Base	RaDIO	DioR	Base	RaDIO	DioR	Base	RaDIO	DioR
EM (BM25)	0.165	0.178	0.214	0.111	0.124	0.163	0.1488	0.1594	0.1719	0.768	0.776	0.789
F1 (BM25)	0.246	0.2641	0.2902	0.1951	0.2236	0.2548	0.1851	0.2039	0.2086	0.768	0.776	0.789
Precision (BM25)	0.259	0.2704	0.2913	0.1931	0.2261	0.2605	0.1899	2.0645	0.2145	0.768	0.776	0.789
Recall (BM25)	0.245	0.2556	0.3032	0.3003	0.3023	0.3052	0.1939	0.2036	0.2153	0.768	0.776	0.789

Metric	2WikiMultihopQA			HotpotQA			IIRC			StrategyQA		
	Base	RaDIO	DioR	Base	RaDIO	DioR	Base	RaDIO	DioR	Base	RaDIO	DioR
Rc (BM25)	1.866	1.015	2.036	2.904	3.024	2.524	1.9004	2.1962	2.0922	1.776	3.026	2.638
Gc (BM25)	3.737	2.301	2.04	4.386	3.985	2.553	3.8050	3.136	2.0953	4.06	2.897	2.703
Hc (BM25)	1.866	1.135	1.019	2.094	1.765	1.263	1.9004	1.693	1.0471	1.776	1.471	1.321
Tc (BM25)	856.514	521.979	442.793	302.894	200.398	194.395	868.561	697.563	515.927	239.286	210.397	171.608
Sc (BM25)	54.754	36.474	28.751	19.146	13.562	12.273	48.010	36.854	30.769	17.845	14.672	12.487

Table 7: Comparison of Gc and Hc Metrics Across Datasets and Retrieval Methods (w/o ED vs DioR)

Metric	2WikiMultihopQA		HotpotQA		IIRC		StrategyQA	
	w/o ED	DioR	w/o ED	DioR	w/o ED	DioR	w/o ED	DioR
Gc (BM25)	3.165	3.006	3.574	3.037	3.566	3.022	3.406	3.206
Gc (SBERT)	3.356	2.021	2.357	2.092	2.431	2.074	2.862	2.280
Gc (SGPT)	2.833	2.033	2.114	2.098	3.012	2.062	2.232	2.206
Hc (BM25)	1.015	1.003	2.041	2.017	2.209	2.010	2.131	2.093
Hc (SBERT)	1.035	1.014	1.075	1.046	1.084	1.030	1.268	1.098
Hc (SGPT)	1.016	1.005	1.304	1.046	1.242	1.035	1.165	1.134

ence time. Consequently, DioR not only achieves accuracy improvements over existing state-of-the-art RAG methods while maintaining answer quality but also significantly reduces computational overhead. The table 7 below presents an efficiency experiment that validates the effectiveness of our approach in improving computational efficiency. (Early-Detection: ED)

C.5 Case Study

As shown in Table 8, we present a case study comparing the Base method and DioR on five example questions, evaluating their ability to generate accurate answers while minimizing hallucinations (Hc), reducing verbosity (Sc), and optimizing response generation (Gc). For each example, we assess the correctness of the predictions and the associated metrics.

1. Does Santa Claus work during summer?

The Base method incorrectly asserts that Santa Claus works during summer based on flawed reasoning, leading to a hallucination (Hc = 2) and generating 16 sentences (Sc = 16). In contrast, DioR correctly deduces that Santa Claus works during winter, producing a correct answer with only one hallucination (Hc = 1) and 15 sentences (Sc = 15), but with a lower number of generations (Gc = 2), indicating a

more efficient response.

2. Would an uninsured person be more likely than an insured person to decline a CT scan?

The Base method fails to reason effectively, resulting in incorrect reasoning and an answer that was not grounded in the text, with 4 hallucinations (Hc = 4) and 46 sentences (Sc = 46). DioR, however, produces a grounded and correct response with minimal hallucination (Hc = 1) and more concise reasoning (Sc = 16) while generating only 2 responses (Gc = 2).

3. Are Christmas trees dissimilar to deciduous trees?

The Base method incorrectly concludes that Christmas trees are similar to deciduous trees, producing a large number of hallucinations (Hc = 6) and an extensive response (Sc = 59). DioR, however, correctly answers that Christmas trees are evergreen and not deciduous, with significantly fewer hallucinations (Hc = 1) and a more succinct response (Sc = 11), generating only 2 answers (Gc = 2), indicating better efficiency and correctness.

4. Is the language used in Saint Vincent and the Grenadines rooted in English?

The Base method produces a false response, relying on incorrect reasoning, resulting in 2 hallucinations ($H_c = 2$) and 15 sentences ($Sc = 15$). DioR provides the correct answer, grounded in more accurate reasoning, with only 1 hallucination ($H_c = 1$) and a more concise response ($Sc = 10$), generating 2 answers ($G_c = 2$).

5. Are more people today related to Genghis Khan than Julius Caesar?

Both methods provide a correct answer, but the Base method’s reasoning leads to a higher number of hallucinations ($H_c = 5$) and a larger response ($Sc = 33$). DioR correctly answers with more concise reasoning, fewer hallucinations ($H_c = 1$), and shorter response length ($Sc = 12$), generating only 2 answers ($G_c = 2$), demonstrating higher efficiency.

6. Does Hades appear in a Disney Channel musical movie?

The Base method incorrectly concludes that Hades does not appear in a Disney Channel musical movie, producing a large number of hallucinations and an extensive response. In contrast, DioR correctly answers, with significantly fewer hallucinations ($H_c = 2$) and a more succinct response ($Sc = 18$), indicating better efficiency and correctness

7. Could someone theoretically use an armadillo as a shield?

Although both the Base and DioR methods produce the correct answer that someone theoretically use an armadillo as a shield. However, DioR demonstrates a greater advantage in metrics.

8. Did a Mediterranean Sea creature kill Steve Irwin?

For this question, the Base method incorrectly believes that Steve Irwin was killed by Stingray, so the answer is incorrect. In contrast, DioR can generate fewer hallucination ($H_c = 1$) to answer correctly.

9. Does a Generation Y member satisfy NYPD police officer age requirement?

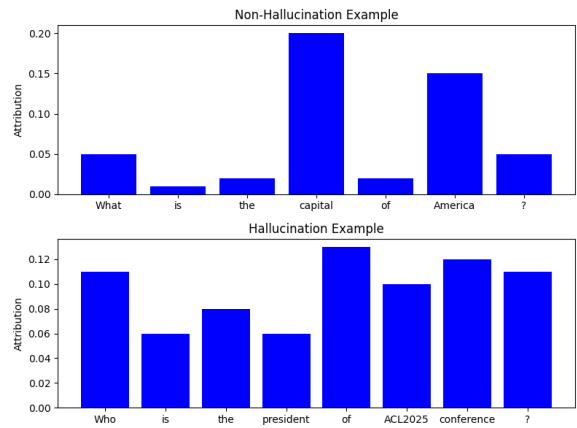


Figure 4: An example of an IG attribution score for hallucination and non-hallucination.

The logic of the Base method is confusing. The age requirement for the New York Police Department is 21 years old, while the age range for millennial members is from 26 to 41 years old (based on members born in 1994). Since the minimum age for members of the millennial generation is 26 years old, which is already higher than 21 years old, they clearly meet the age requirements of the New York Police Department. On the contrary, DioR can correctly derive the answer and all indicators are superior to Base.

10. Would someone with back pain enjoy picking strawberries?

The description of the Base method is incorrect. Strawberry picking is not a sedentary activity, but a physical activity that requires frequent bending, twisting, and weightlifting. This activity may cause discomfort for people suffering from back pain. And DioR can output the correct answer, that strawberry picking does require these actions, which may cause discomfort for people with back pain.

DioR consistently outperforms the Base method across all five examples by producing correct answers with fewer hallucinations, more efficient responses (lower G_c), and more concise text (lower Sc). These results demonstrate DioR’s superior ability to ground its answers while minimizing unnecessary verbosity, highlighting its advantage in handling complex reasoning tasks.

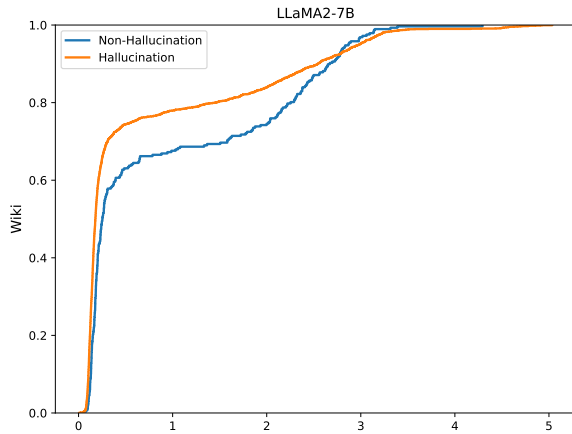


Figure 5: The difference in the distribution of the output entropy values when the model generates the correct answer (non-hallucination) and the wrong answer (hallucination).

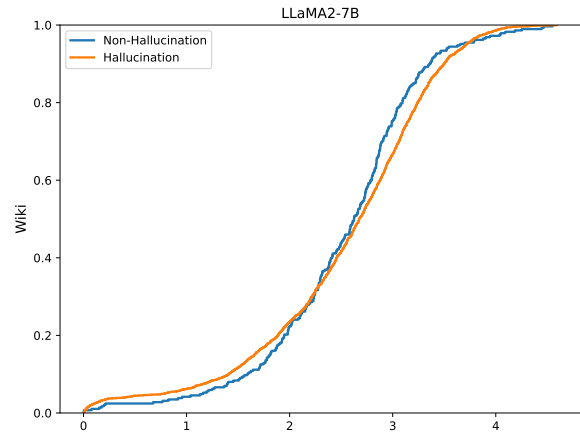


Figure 8: The model outputs the difference in softmax probability distribution when generating correct answers and incorrect answers.

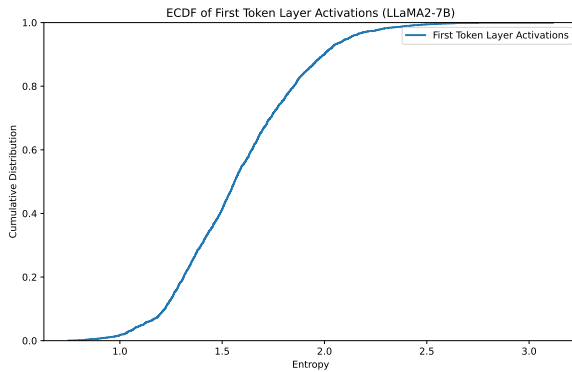


Figure 6: The uncertainty distribution of the model when processing the input and generating the first token.

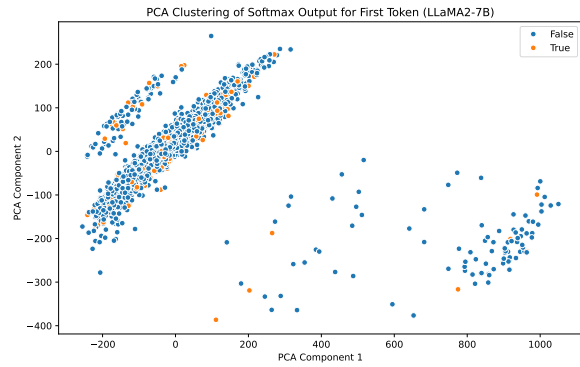


Figure 9: The distribution of correct and incorrect answers in the PCA dimensionality reduction space when the model generates the first token.

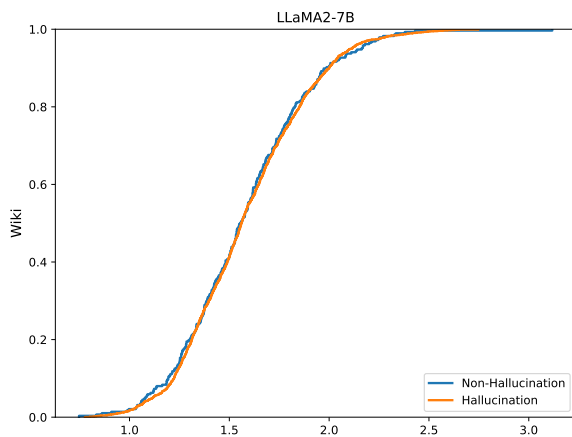


Figure 7: The difference in the distribution of input token attribution scores when the model generates correct answers and incorrect answers.

C.6 Effectiveness Analysis of Hallucination Detection Methods

As mentioned earlier, to validate whether our method can effectively determine whether a large language model (LLM) is capable of answering a question, we assess the model’s attention to particular words within the input question. Specifically, we use the Integrated Gradients (IG) method to generate feature attribution, which allows us to quantify the attention distribution of the LLM when processing different questions. This approach enables us to determine whether the model is able to recognize and focus on the key features of a question during the answering process, to assess the level of confidence LLMs have in facing a new problem.

As shown in Figure 4. By comparing instances of hallucination and non-hallucination generation, we observe significant differences in the model’s attention distribution between the two cases. In

non-hallucination examples, the model accurately identifies and focuses on the key information in the question. In contrast, in hallucination examples, the model’s attention is more dispersed, failing to concentrate on the core words, which leads to incorrect or irrelevant responses. These experimental results indicate that the model’s ability to effectively focus on key features directly impacts the accuracy of its answers, thus validating that our method can effectively assess whether an LLM is capable of answering a question before the generation to reduce the probability of hallucination generation (The confidence level of LLMs in answering questions).

We can observe from Figures 5, 6, 7, 8, and 9 that the model exhibits noticeable differences when generating hallucinations versus non-hallucinations. The horizontal axis of the first three Figures is the entropy value, the vertical axis is the cumulative distribution function, and the horizontal axis of the fourth Figure is the Softmax probability value. These Figures provide insight into the behavior of the LLaMA2-7B model in generating text, analyzed through various methods such as ECDF, PCA, integrated gradients, and more.

As shown in Figure 5, the two curves for non-hallucination and hallucination are clearly separated in specific entropy ranges, indicating that the model’s behavior differs significantly between these two scenarios.

Figure 6 illustrates the cumulative distribution of entropy values from the first layer of activation values when the model generates the first token. By comparing the distributions of activation values for correct and incorrect answers, we can observe the differences in the model’s uncertainty during the generation of the initial word, further revealing how the internal state varies between correct and incorrect answers. A higher cumulative distribution in the higher entropy regions suggests that the model experiences greater uncertainty when generating the first token.

Figure 7 shows the cumulative distribution of entropy values for input token attribution scores generated using the integrated gradients (IG) method. This figure highlights the disparity in the model’s attention to input tokens when generating correct and incorrect answers. We can observe a subtle difference in IG scores between scenes that produce hallucinations and those that do not.

Figure 8 displays the cumulative distribution of the Softmax output probabilities. Higher Softmax

probability values reflect greater model confidence in a prediction. We can observe that in the high softmax range, the range of non-hallucinations is greater than that of hallucinations, indicating that the model is more confident in generating non-hallucination scenes. However, in the low softmax range, the model appears less confident and prone to hallucinations.

Lastly, Figure 9 presents a two-dimensional scatter plot of the Softmax output after PCA dimensionality reduction for the first token. The distribution of correct and incorrect predictions in the PCA space shows a clear separation between the two, indicating that the Softmax output is significantly different when the model generates correct versus incorrect answers.

These findings validate that our Early-Detection method can effectively assess whether a large language model (LLM) is capable of answering a question before it starts generating text.

We also experimented with the Real-time Detection module to prove its feasibility in hallucination detection. We conduct experiments on LLaMA2-7B to judge the state-of-the-art methods for sentence-level hallucination detection, achieving an AUC of 0.7913, compared to 0.6583 for GPT4-HDM (Achiam et al., 2023) and 0.7876 for MIND (Su et al., 2024b).

D Hallucination Prevention vs. Detection and Regeneration

Listing 1: Hallucination Prevention (DioR)

```
function answerWithHallucinationControl(question
):
    // Initial detection phase
    confidence = EarlyDetection(question)

    // Based on confidence, decide whether to
    retrieve information upfront
    context = confidence ? "" :
        retrieveRelevantInformation(question)
    answer = ""

    // Token-by-token generation
    while not isEndOfSequence(answer) and len(
        answer) < max_tokens:
        // Generate next token
        next_token = generateNextToken(question,
            answer, context)

    // Real-time detection for hallucination
    in the token
    if RealTimeDetection(next_token):
        // Hallucination detected, retrieve
        relevant documents
        new_context = Retrieve()
```

```

// Continue generation with newly
retrieved context
return continueGeneration(question,
answer, new_context)

// Token is valid, add it to the answer
answer += next_token

return answer

```

Listing 2: Detection and Regeneration

```

function answerWithHallucinationControl(question
):
answer = generateLLMResponse(question)
isHallucination = detectHallucination(
question, answer)

if isHallucination:
// Discard the hallucinated answer
retrievedContext =
retrieveRelevantInformation(question)
return generateNewResponse(question,
retrievedContext)
else:
return answer

```

E Traditional RAG vs. Dynamic RAG

Traditional RAG: Traditional RAG uses a single-round retrieval method to perform a one-time document retrieval operation before generating answers. This method is based on the user’s original query, retrieves relevant documents, and simultaneously inputs all retrieved information into a large language model, forming a static contextual window. Due to the fact that the retrieval process is only performed once before the start of generation, the model cannot obtain additional knowledge based on the new information requirements that arise during the generation process, which may result in information gaps or inaccuracies in the answers.

Dynamic RAG: Dynamic RAG fundamentally changes the way information is obtained and integrated by implementing multiple rounds of retrieval processes. It consists of two key steps: 1) Determining the appropriate retrieval timing during the generation process of the LLM; 2) Formulating queries upon retrieval activation.

Listing 3: Traditional RAG Pseudocode

```

def traditional_rag(user_query, knowledge_base):
# 1. Single-round retrieval based on user
query
relevant_documents = retrieve_documents(
user_query, knowledge_base)

# 2. Input retrieved documents and user query
to LLM
context = prepare_context(relevant_documents)

```

```

# 3. Generate response
response = generate_response(user_query,
context)

return response

```

Listing 4: Dynamic RAG Pseudocode

```

def dynamic_rag(user_query, knowledge_base):
# Initialize response state
response_so_far = ""
llm_state = initialize_llm()

# Iteratively generate response
while not is_generation_complete(llm_state):
# 1. Determine if retrieval is needed at
current generation step
if should_retrieve_now(llm_state,
response_so_far, user_query):
# 2. Dynamically construct retrieval
query
retrieval_query = formulate_query(
llm_state, response_so_far,
user_query)

# 3. Retrieve documents
relevant_documents =
retrieve_documents(
retrieval_query, knowledge_base)

# 4. Update LLM state with new
context
llm_state = update_llm_with_context(
llm_state, relevant_documents)

# 5. Continue generating the next part of
the response
next_token, llm_state =
generate_next_token(llm_state)
response_so_far += next_token

return response_so_far

```

F The prompt words input to LLM after removing the “Post-retrieval” component in the ablation experiment

After removing the “Post-retrieval” component in the ablation experiment, the RAG prompt template is as follows:

External Knowledge: [1] K_{i1} [2] K_{i2} ...
Using the external knowledge provided above, please answer the following question:
Question: [Ques.]
Answer: Insert truncated output [] and additional relevant details here

Table 8: Comparison between the Base method and DioR methods in terms of Hc, Sc, and Gc for a case study. The table highlights specific examples, their predictions and outcomes, and the metrics associated with each method. A checkmark (✓) denotes correct outcomes, while a cross (×) indicates incorrect outcomes.

Example	Base			Num.			DioR			Num.		
	Prediction	Outcome	Hc	Sc	Gc	Prediction	Outcome	Hc	Sc	Gc		
1	The text states that Howard worked at the train station, which is a place of business. Businesses are typically open during the summer. Thus, Santa Claus works during summer. So the answer is yes.	×	2	16	5	Santa Claus works during winter. Winter is from December to February. Summer is from June to August. Thus, Santa Claus does not work during summer. So the answer is no.	✓	1	15	2		
2	An uninsured person would be more likely than an insured person to decline a CT scan. Reasoning: 1. The diagnostic accuracy of coronary CT angiography (CCTA) is similar to that of invasive coronary angiography via cardiac catheterization (ICA). (1) 2. Both CCTA and ICA yield similar prognostic information, such as stenosis severity and extent of coronary artery disease.	×	4	46	8	The cost of a CT scan is \$500. If a person has insurance, the person's insurance company pays for the CT scan. If a person does not have insurance, the person pays for the CT scan. Thus, if a person does not have insurance, the person is more likely than an insured person to decline a CT scan. So the answer is yes.	✓	1	16	2		
3	1. The text states that the Christmas tree was a 40-foot Colorado Blue Spruce donated by the Loving Shepherd Lutheran Church and School. 2. Deciduous trees are trees that lose their leaves in the fall. 3. Colorado Blue Spruce trees are evergreen, which means they keep their needles year-round. Thus, the Christmas tree is not dissimilar to deciduous trees. So the answer is no.	×	6	59	13	Christmas trees are evergreen. Evergreen trees are not deciduous. Thus, Christmas trees are dissimilar to deciduous trees. So the answer is yes.	✓	1	11	2		
4	No. Reasoning: The passage states that the official language of Guernsey is English, but it also mentions that French was the official language until 1948. This implies that the language used in Guernsey is not rooted in English. Therefore, the answer is no.	×	2	15	5	Saint Vincent and the Grenadines is an independent country. The official language of an independent country is the language of the country. Thus, the language used in Saint Vincent and the Grenadines is rooted in English. So the answer is yes.	✓	1	10	2		
5	Genghis Khan lived in the 13th century. Julius Caesar lived in the 1st century. Today is in the 21st century. The distance between the 13th century and the 21st century is 890 years. There are 890 years between the number of people alive today and Genghis Khan. Thus, more people today are related to Genghis Khan than Julius Caesar. So the answer is yes.	✓	5	33	11	The world population is about 7 billion. Genghis Khan lived about 800 years ago. Julius Caesar lived about 2000 years ago. So, more people today are related to Genghis Khan than Julius Caesar. Thus, the answer is yes.	✓	1	12	2		
6	No. Explanation: 1. Hades is a character from Greek mythology. 2. The Disney Channel is a television network. 3. Musical movies are movies with music. 4. Hades does not appear in any Disney Channel musical movies. Therefore, the answer is no.	×	3	27	6	Hades appears in the Disney Channel musical movie <i>Hercules</i> . Thus, the answer is yes.	✓	2	18	3		
7	Someone theoretically could use an armadillo as a shield. Reasoning: 1. Armadillos have a hard carapace that protects their body. 2. The carapace is flexible, allowing the armadillo to roll into a ball for defense. 3. The armadillo's small size makes it easier to hold and use as a shield. 4. The armadillo's curved claws could be used to grip onto an attacker's shield.	✓	7	70	14	An armadillo is a mammal. Mammals can use their bodies as shields. Thus, someone theoretically could use an armadillo as a shield. So the answer is yes.	✓	2	25	3		
8	Mediterranean Sea creatures are not known to kill humans. Steve Irwin was killed by a stingray. Thus, no Mediterranean Sea creature killed Steve Irwin. So the answer is no.	×	2	11	3	Steve Irwin died when a stingray barb pierced his chest. The stingray lives in the Mediterranean Sea. Thus, the Mediterranean Sea creature that killed Steve Irwin was a stingray. So the answer is yes.	✓	1	13	3		
9	The NYPD police officer age requirement is 21 years old. A Generation Y member is someone born between 1980 and 1994. The youngest Generation Y member is 26 years old, which is older than the NYPD police officer age requirement. Thus, a Generation Y member does not satisfy the NYPD police officer age requirement. So the answer is no.	×	3	24	7	The NYPD age requirement is 21. Generation Y is 1980 and 1994. 1994 is the year Generation Y ends. Thus, a Generation Y member could satisfy the NYPD age requirement. So the answer is yes.	✓	2	14	3		
10	Someone with back pain could enjoy picking strawberries. The reason is that picking strawberries is a sedentary activity. Sedentary activities do not strain the back. Thus, picking strawberries would not cause back pain.	×	3	23	6	Strawberry picking typically involves bending, twisting, and lifting weights, which may cause discomfort for patients with back pain.	✓	2	12	3		