RaDIO: Real-Time Hallucination Detection with Contextual Index Optimized Query Formulation for Dynamic Retrieval Augmented Generation

Jia Zhu[1], **Hanghui Guo**[1], **Weijie Shi**[2*], **Zhangze Chen**[1], **Pasquale De Meo**[3]

[1]The Zhejiang Key Laboratory of Intelligent Education Technology and Application, Zhejiang Normal University, China
[2]Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, China
[3]Department of Computer Science, University of Messina, Italy
{jiazhu, ghh1125}@zjnu.edu.cn, wshiah@connect.ust.hk, zjnuczz@zjnu.edu.cn, pdemeo@unime.it

Abstract

The Dynamic Retrieval Augmented Generation (RAG) paradigm actively decides when and what to retrieve during the text generation process of Large Language Models (LLMs). However, current dynamic RAG methods fall short in both aspects: identifying the optimal moment to activate the retrieval module and crafting the appropriate query once retrieval is triggered. To overcome these limitations, we introduce an approach, namely, **RaDIO**, **R**eal-Time H**a**llucination **D**etection with Contextual **I**ndex **O**ptimized query formulation for dynamic RAG. The approach is specifically designed to make decisions on when and what to retrieve based on the LLM's real-time information needs during the text generation process. We evaluate RaDIO along with existing methods comprehensively over several knowledge-intensive generation datasets. Experimental results show that RaDIO achieves superior performance on all tasks, demonstrating the effectiveness of our work.

Code — https://github.com/ghh1125/RaDIO

Introduction

Large Language Models (LLMs) have advanced significantly in a range of natural language processing (NLP) tasks and are increasingly being integrated into various AI applications (Chowdhery et al. 2022; Touvron et al. 2023a). Despite their impressive capabilities, LLMs often generate text that appears coherent but is factually incorrect, a problem known as *LLM hallucination* (Maynez et al. 2020; Ji et al. 2023a,b; Su et al. 2024c).

Retrieval-Augmented Generation (RAG) has become a key approach to addressing hallucinations in LLMs. As illustrated in Figure 1, RAG improves LLMs by retrieving relevant information from external databases and adding it to the model's inputs (Jiang et al. 2022). In many NLP challenges, the use of RAG has led to superior effectiveness (Borgeaud et al. 2022; Shi et al. 2023b).

Traditional RAG usually relies on *single-round retrieval*, that is they use the LLM generated text as input to retrieve relevant information from external sources. This method performs well for simpler tasks but struggles with complex, multi-step, and long-form generation tasks.

*Corresponding author.

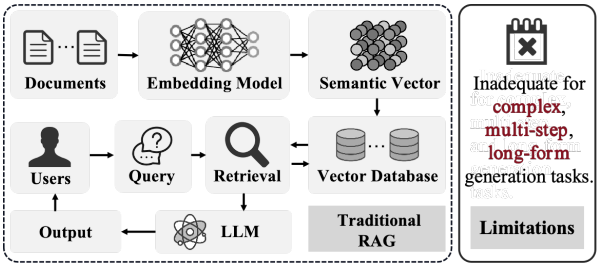

Figure 1: The framework of traditional RAG

A more advanced solution is the *dynamic RAG*, i.e., a RAG that performs multiple retrievals during the LLM's generation process (Borgeaud et al. 2022). Dynamic RAG involves two key steps: identifying the optimal time instant to activate the retrieval module and crafting the appropriate query once retrieval is triggered (Jiang et al. 2022).

Several kinds of dynamic RAG methods are currently available and each of them implements ad-hoc strategies to carry out the two steps above. For instance, IRCoT (Trivedi et al. 2022) employs a *global augmentation method*, and it activates the retrieval module for each generated sentence, and the query is formed using the latest sentence. IC-RALM (Ram et al. 2023) defines a sliding window and triggers the retrieval module after processing a preset number of tokens, using the last n tokens to form the query.

Designing a dynamic Retrieval-Augmented Generation technique raises two crucial questions: When is the optimal time to initiate the retrieval phase? How can we formulate effective queries to external sources that address the immediate information needs of a Large Language Model?

Current dynamic RAG strategies often rely on static rules to determine when to retrieve information, without considering whether retrieval is truly necessary. This approach can lead to unnecessary retrievals, introducing irrelevant or noisy data that compromises the quality of the LLM output.

Furthermore, implementing retrieval augmentation increases the time and computational cost of LLM inference. This added cost is unwarranted if the LLM can produce accurate output independently or if the integrated content does not meet acceptable standards.

Moreover, current RAG strategies for determining what to

retrieve typically focus on the LLM's most recent sentence or last few tokens. However, this narrow scope may fail to capture the model's real-time information needs, which could span the entire context. Consequently, retrieving documents based solely on recent sentences or tokens is likely to yield suboptimal results in LLM generation performance.

To address these limitations, we propose a new framework called **RaDIO** (**R**eal-Time H**a**llucination **D**etection with Contextual **I**ndex **O**ptimized query formulation for dynamic RAG), which is specifically designed to make informed decisions about when and what to retrieve, ensuring that the generated content is accurate and relevant.

A crucial component of RaDIO is our novel hallucination detection method, which determines when to initiate retrieval. Unlike existing approaches that rely on static rules, our method uses a data-driven approach to select the optimal time for retrieval. Our approach considers three key factors: the uncertainty of the Large Language Model about its generated content, the importance of choosing an entity, and the semantic significance of each entity. By analyzing these factors, we can identify instances where the model may be producing inaccurate information or "hallucinating".

To achieve this, our hallucination detection module first randomly selects entities from Wikipedia documents on a given topic and truncates the documents at those entities. From the truncation point, the LLM then generates uncertain content (entities) that might have hallucinations. We compare the entities at the beginning of the generated content with the original entities from the truncated Wikipedia text to detect semantic discrepancies, as hallucinations often involve changes in key entities (e.g., incorrect dates, events, or names). Using this approach, we can construct a large training set of hallucination examples and train a multi-layer perceptron (MLP)-based hallucination detector. This detector takes the contextual embeddings of tokens from the final layer of the LLM's Transformer architecture as input and provides real-time feedback to the RAG system.

Once we have determined the best possible time to trigger the RAG, we need to formulate a query that retrieves the most relevant documents.To achieve this, we propose a novel query formulation approach that leverages multi-headed attention mechanisms. Our method benefits from the information associated with different representational subspaces, allowing it to accurately distinguish between important and less important tokens. Our approach considers the content and relative position of each token within the text, providing a deeper understanding of the contextual relevance and importance of each token. We further refine our query by analyzing the semantic similarity between embedding vectors, selecting the most crucial tokens that convey the essential meaning and context. By integrating these components, our query formulation process generates a high-quality query that effectively captures the nuances of the input text, leading to improved performance in downstream tasks.

We evaluate RaDIO against other well-known dynamic RAG frameworks across four knowledge-intensive generation benchmarks. The results demonstrate that RaDIO outperforms all other methods on these datasets, demonstrating the effectiveness of our method. Additionally, the results of the ablation study indicate that our strategies for *when to retrieve* and *what to retrieve* consistently outperform other strategies, regardless of the retrieval model used.

The main contributions of our work are as follows:

- **Introduction of RaDIO Framework**: We propose a novel dynamic Retrieval-Augmented Generation (RAG) framework named RaDIO, which stands for Real-Time Hallucination Detection with Optimized Query Formulation. Unlike previous approaches, RaDIO dynamically optimizes both the timing and content of retrievals based on the Large Language Model's (LLM) real-time information needs during text generation.
- **Unsupervised Real-Time Hallucination Detection**: We develop an unsupervised training method for real-time hallucination detection, leveraging the internal states of the LLM throughout the text generation process. This approach allows the system to identify and address hallucinations as they occur, ensuring more accurate and reliable outputs while reducing the LLM's content uncertainty during generation.
- **Comprehensive Evaluation on Knowledge-Intensive Datasets**: We conduct an extensive evaluation of existing dynamic RAG methods alongside RaDIO across four knowledge-intensive datasets using LLMs. Our experimental results demonstrate that RaDIO achieves state-of-the-art (SOTA) performance across all tasks, showcasing the effectiveness and superiority of our approach.

Related Works

LLMs have shown impressive effectiveness across various tasks, but their built-in knowledge often falls short for knowledge-intensive applications. To address this, RAG strategies are widely used to enhance LLM performance.

A common method is *single-round retrieval augmentation* (Khandelwal et al. 2019; Borgeaud et al. 2022; Jiang et al. 2022; Shi et al. 2023a), which uses the initial input to query an external corpus and integrates the retrieved knowledge into the model (Mondal et al. 2024). For instance, REPLUG (Shi et al. 2023a) uses LLMs to generate training data for retrieval models, while UniWeb (Li et al. 2023) employs an adaptive search engine to determine when retrieval is needed. While effective for straightforward tasks, single-round retrieval struggles with complex tasks involving long-form text, such as open-domain summarization and chain-of-thought reasoning. In these cases, relying solely on the initial input often lacks sufficient external knowledge (Jiang et al. 2023a; Salemi and Zamani 2024).

To address this, *multi-round retrieval augmentation* strategies have been explored. RETRO (Borgeaud et al. 2022) and ICRALM (Ram et al. 2023) perform retrieval every few tokens, while IRCot (Trivedi et al. 2022) retrieves information for each sentence. Fixed intervals can be inefficient, so FLARE (Jiang et al. 2023a) triggers retrieval when encountering uncertain tokens. Recently, Su *et al.* (Su et al. 2024a) introduced DRAGIN, a dynamic RAG framework that makes real-time retrieval decisions based on LLM needs. However, DRAGIN's reliance on attention entropy and token relevance may be insufficient.

Despite advancements, current retrieval-augmented methods still have limitations. Single-round retrieval may not cover all necessary external knowledge for complex tasks, while multi-round techniques with fixed intervals or simplistic triggers may not fully align with LLMs' dynamic needs. DRAGIN, despite its innovation, relies on potentially inadequate metrics for retrieval triggering.

Proposed Approach

In this section, we introduce RaDIO, our approach to Dynamic Retrieval-Augmented Generation. As shown in Figure 2, RaDIO includes two main components: ***Real-time Hallucination Detection*** and ***Contextual Index Optimized Query Formulation***, described in detail below.

Real-time Hallucination Detection

Most existing dynamic RAG frameworks rely on static, predefined rules to trigger the retrieval module. An example of a static rule to trigger the retrieval module can be found in the FLARE system (Jiang et al. 2023a), which dynamically initiates retrieval when the LLM's next-token generation probability falls below a certain threshold. However, the activation of the retrieval module should depend not only on the generation probability but also consider the token's importance within the global generation context. Meanwhile, hallucinations often involve changes in entities (e.g., incorrect dates or names) (Chang et al. 2024). Therefore, how to effectively judge the semantic differences of entities is one of the urgent problems to be solved in judging hallucinations.

To address these limitations, we propose an enhanced real-time hallucination detection approach for triggering retrieval within dynamic RAG frameworks which takes into account the importance of tokens of entities.

We leverage unsupervised data generation and automatic annotation of LLM-generated content to detect hallucinations. The process of hallucination detection is graphically shown in Figure 3 and, initially, it selects a Wikipedia dataset $\mathcal{W}$ represented as $\mathcal{W} = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n\}$, where each $\mathbf{w}_i$ represents an individual article. Subsequently, we select key entities (eg. name, year, number, and other significant terms) from each article $\mathbf{w}_i$ and truncate each document $\mathbf{w}_i$ according to the locations of the extracted entities (Except for the beginning of each sentence). Truncated documents are then sent as input to 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]."* The LLM then generates a piece of text, say $\mathbf{T}_i$, for each truncated segment.

To detect hallucinations, we perform entity extraction on the generated text $\mathbf{T}_i$ at each breakpoint of the original article. However, entities often exist in multiple forms of expression, such as abbreviations, variations in naming conventions, or differences in word order. Therefore, simple direct matching may not capture all entity-level errors.

To address this, we combine manual evaluations with automated comparisons between the extracted entities and the originally selected entities. For example, we leverage external linguistic resources such as WordNet (Miller 1995)

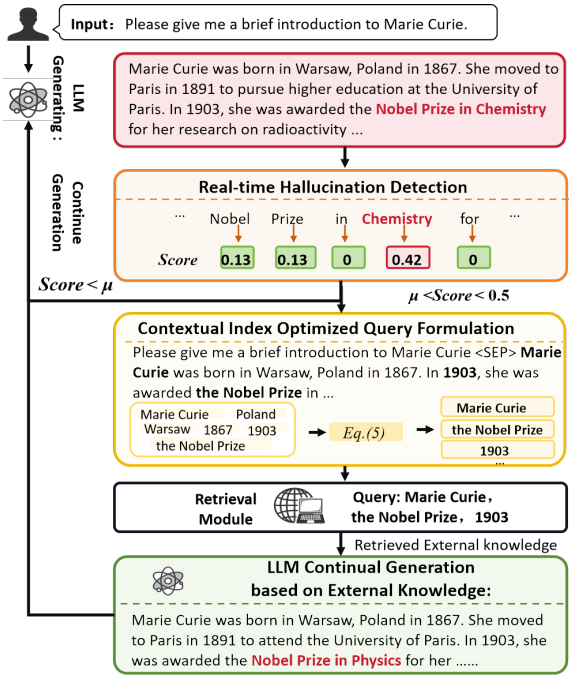

Figure 2: The framework of RaDIO

and utilize Word2Vec (Rong 2014) to calculate the cosine similarity between the vectors of different entity representations. This allows us to assess whether two entities are semantically equivalent. If the entities are semantically similar (e.g., *"NBA"* and *"National Basketball Association"* are the same), no hallucination is detected. Otherwise, we label the LLM-generated text as hallucinatory.

More formally, for each article $\mathbf{w}_i$, we construct a tuple $\mathbf{D}_i$=$\{\mathbf{w}_i, \mathbf{T}_i, \mathbf{e}_i, \mathbf{n}_i, H_i\}$, where $\mathbf{T}_i$ is the LLM-generated article, $\mathbf{e}_i$ is the originally selected entity, $\mathbf{n}_i$ is the entity extracted from the LLM-generated text and H_i is the hallucination label, i.e., 1 for hallucinatory, 0 for non-hallucinatory. This process enables us to automatically annotate a large dataset with hallucination labels, which can be used as a training set to detect hallucinations.

We train an MLP (multi-layer perceptron) using input matrices derived from the processed Wikipedia hallucination dataset $\mathbf{D}_i$, which includes LLM-generated text and associated features. More specifically, the classifier outputs a probability P, indicating the likelihood of hallucination occurrence when generating a specific text segment:

$$P = \text{MLP}\left(\text{Vec}\left(\text{Mx}\left(\mathbf{D}_i\left\{\mathbf{w}_i, \mathbf{T}_i, \mathbf{e}_i, \mathbf{n}_i : \mathbf{H}_i\right\}\right)\right) + b\right), \quad (1)$$

where $\text{Vec}(\text{Mx}(\mathbf{D}_i))$ represents the process of converting the input features $\mathbf{D}_i$ from the hallucination dataset into a matrix and then into vector form, and b is the bias vector.

Meanwhile, we utilize the binary cross-entropy (BCE) loss to train the classifier:

$$L_{\text{BCE}}(y_i, p_i) = y_i \log(p_i) + (1 - y_i) \log(1 - p_i), \quad (2)$$

where y_i is a true label of the i-th example, and p_i indicates the predicted probability of belonging to the positive class.

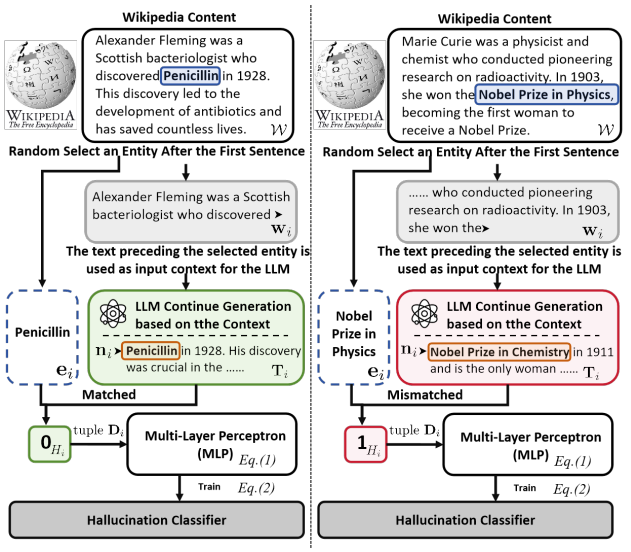

Figure 3: The Real-time Hallucination Detection Framework

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

To further enhance the accuracy and efficiency of hallucination detection, we employ a token-level scoring mechanism. Specifically, for each token generated by the LLM's last layer, we compute its probability score using an MLP that has been previously trained on Wikipedia data. If this score exceeds the average score of all tokens and the probability output is below 0.5, we indicate a potential hallucination and trigger the RaDIO alarm. This approach allows us to achieve dynamic (real-time) hallucination detection for LLM outputs, improving judgment efficiency and accuracy while reducing resource consumption.

Contextual Index Optimized Query Formulation

Once the retrieval augmentation position is identified (i.e., the hallucination monitoring token's location), the following step in the RAG framework is to formulate a query to retrieve necessary information from an external database, enabling the LLM to continue generating text. However, existing dynamic RAG frameworks rely solely on attention weights of the LLM's latest sentence or the last few tokens but such a solution is not entirely satisfactory because it may not be able to accurately represent the relationships between words or indicate the importance of specific tokens.

To mitigate this limitation, RaDIO leverages the inherent framework of Transformer-based LLMs and introduces a multi-head attention mechanism to capture information from multiple representation subspaces.

Our approach has two significant advantages: on the one hand, it can employ contextual features and similarity information of the tokens used to detect the hallucination, and on the other hand, it provides a more comprehensive understanding of which tokens to prioritize as retrieval objects.

The multi-head attention mechanism enables the model to compute attention weights through multiple heads and subsequently combine them to generate more precise weight judgments (Vaswani 2017). 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. This mechanism involves several steps: linear transformations of the input, concatenation of the results, and transformation of the attention weights to produce the final attention representation. Specifically, the multi-head attention result is calculated using the following formula:

$$\text{Att}_i(Q, K, V) = C(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)W^O, \quad (3)$$

where C denotes the concatenation operation and head_i represents the attention result for the i^{th} head, calculated as:

$$\text{head}_i = \text{softmax}\left(QW_i^Q \cdot (KW_i^K)^T \cdot d_k^{-\frac{1}{2}}\right) VW_i^V. \quad (4)$$

Here, Q, K, and V represent the query, key, and value matrices, respectively. W_i^Q, W_i^K, and W_i^V are linear transformation matrices for the i^{th} attention head, and W^O is the output linear transformation matrix (Eq.3).

In addition to attention scores, we also need to consider the actual content and relative position of tokens in the text context to evaluate their importance. To this end, RaDIO calculates the TF-IDF score (TD) of each word in the text to identify words with high information density enabling the system to prioritize content-rich tokens over less informative ones, thereby better capturing key parts of the information.

Meanwhile, we extract the corresponding words from the vocabulary table for these token markers, arrange them in their original order in the text, and normalize their positions using the token position index to obtain a relative position score P_i. The position score P_i is calculated as $P_i = \frac{\text{pos}(i)}{N}$, where $\text{pos}(i)$ is the position of the token in the text, and N is the total number of tokens in the text.

By incorporating TF-IDF score and position score evaluation into our framework, we enhance our ability to capture the significance of each word within the context of the entire text, rather than relying solely on attention scores. This approach complements the attention mechanism by assigning greater weight to rare but contextually important tokens, ensuring that retrieval focuses on the most relevant parts of the generated text (Ramos et al. 2003; Radford et al. 2019). As a result, we can perform more precise importance evaluations for tokens, identifying those that are most pertinent for subsequent retrieval tasks.

Simultaneously, RaDIO also leverages the similarity between word embedding vectors to measure the semantic relevance between all words in the context and query words. This allows us to more accurately identify which words have semantic relevance to the query content. Specifically, we employ a pre-trained word embedding model to convert each word in the text into a vector representation. We then compute the cosine similarity between these vectors and the vector representation of the query words. Cosine similarity

effectively measures the directional similarity between two vectors, which assists in identifying the words most semantically relevant to the query content. Through this process, we can extract the most relevant words from the text, optimize retrieval results, and ensure that the retrieved information better matches the query requirements, thereby reducing noise at the word level.

To assess the significance of each token, we use a multi-faceted evaluation framework integrating attention scores, TF-IDF values, and cosine similarity measures. We normalize these metrics to a common scale for fair comparison and combine the normalized values using linear addition to produce a comprehensive token importance score:

$$\text{Tok}_i = \sigma(\text{Att}_i) + \sigma(\text{TD}_i) + \sigma(\text{Cos}_i) + \text{P}_i, \quad (5)$$

where σ denotes the normalization function and $+$ represents the addition of the four token computation results.

To enhance the effectiveness of the retrieval process, we incorporate token-level contextual information and relevance scores to refine the selection of documents, ensuring that the retrieved information is highly pertinent. Specifically, RaDIO leverages the computed token importance scores to rank tokens in descending order of their significance. This ranking helps identify the most critical tokens likely to be relevant for the retrieval task and obtain external knowledge support. Subsequently, RaDIO employs a state-of-the-art retrieval model (e.g., SGPT) to fetch relevant documents from an external knowledge base, based on the top-ranked tokens. The retrieved documents are seamlessly integrated into the LLM's prompt template:

External Knowledge: [1] K_{i1} **[2]** K_{i2} **[3]** K_{i3} **... Using the external knowledge provided above, please answer the following question: Question: [Qes.] Answer: Insert truncated output [] and additional relevant details here**

This integration effectively bridges the knowledge gaps in the LLM's generation process. Specifically, at the truncation point where the LLM-generated content exhibits hallucinations, we integrate external retrieval knowledge to enable the LLM to continue generating content from this point based on external knowledge.

This synergistic integration of token importance evaluation, ranking-based token selection, and external knowledge retrieval allows RaDIO to bridge the knowledge gaps in the LLM's generation process, ensuring that the generated content is more accurate and complete, with a noticeable improvement in the quality and reliability of the output, while also alleviating the uncertainty (Hallucination) of the Large Language Model about its generated content.

Experiments

Experimental Setups

Datasets. We consider four benchmark datasets for experimental analysis, namely: ***a)*** 2WikiMultihopQA (Ho et al. 2020): This dataset tests the model's ability to perform chain-of-thought (CoT) reasoning and generate answers by integrating information from multiple Wikipedia articles. ***b)*** HotpotQA (Yang et al. 2018): This dataset, similar to 2WikiMultihopQA, focuses on questions requiring information from multiple documents and generates CoT reasoning processes and final answers to evaluate the model's ability to connect different pieces of evidence. ***c)*** StrategyQA (Geva et al. 2021): This dataset assesses commonsense reasoning by posing questions that require implicit strategies, generating CoT processes and final answers to evaluate abstract thinking and commonsense knowledge. ***d)*** IIRC (Abdelsalam et al. 2021): This dataset tests reading comprehension by requiring the integration of information from multiple documents, with questions often needing answers that span several passages and demand advanced synthesis.

Evaluation Metrics. We employed a comprehensive set of metrics to evaluate the performance of RaDIO, and other RAG methods across different datasets and retrieval methods. These metrics can be broadly categorized into two groups: *effectiveness metrics* and *efficiency metrics*.

Effectiveness metrics measure the accuracy and quality of generated answers. We used Exact Match (EM), F1 Score, Precision (Pre.), and Recall (Rec.). EM gauges the proportion of answers that exactly match the reference, offering a strict accuracy assessment. The F1 Score, the harmonic mean of precision and recall, provides a balanced evaluation of answer quality. Precision and recall were assessed to evaluate response accuracy.

Efficiency metrics evaluate computational resources and time complexity. We measured the Retrieve Count (Rc) for the number of documents retrieved, indicating the model's efficiency in gathering relevant information. Generate Count (Gc) reflects the number of times answers are produced, assessing response generation capacity. Hallucinated Count (Hc) tracks occurrences of content not grounded in the input or retrieved documents. Finally, Token Count (Tc) and Sentence Count (Sc) evaluate verbosity and response length.

Implementation. We choose six text generation baselines for comparison (Su et al. 2024b) such as DRAGIN, wo-RAG, SR-RAG, FL-RAG, FS-RAG, and FLARE. The wo-RAG means the LLM does not apply any RAG. SR-RAG, according to the initial query, meaningful passages are retrieved from an external corpus and integrated into the input of the LLM. FL-RAG, the retrieval module is triggered every n tokens, FS-RAG, the retrieval module is triggered every sentence (Trivedi et al. 2022), FLARE, the retrieval model is triggered every time an uncertain token is encountered (Jiang et al. 2023b), and finally DRAGIN (Su et al. 2024a), which has been already presented in the related work section. Note that wo-RAG and SR-RAG are single-round RAG, while the remaining methods implement a multi-round retrieval augmentation strategy.

In our experiments, DRAGIN serves as the Base, as referenced in other tables throughout this work. We choose LLaMA2-7B CHAT (Touvron et al. 2023b), a fine-tuned LLM specifically designed for dialogue-based applications, as the underlying LLM. We also employed three retrieval strategies, namely: BM25 (Lv and Zhai 2011), SBERT (Wang and Kuo 2020) and SGPT (Muennighoff 2022).

Experiments are conducted on two NVIDIA H800 GPUs, taking approximately 18 hours. As for the hyperparameter settings, we set the maximum generated sequence length to 256 tokens, with a retrieval top-k value of 3; we selected the

Matrix	2WikiMultihopQA		HotpotQA		IIRC		StrategyQA	
	Base	**RaDIO**	Base	**RaDIO**	Base	**RaDIO**	Base	**RaDIO**
EM (BM25)	0.214	0.228	0.219	**0.246**	0.156	**0.196**	0.639	**0.654**
EM (SGPT)	0.209	0.214	0.202	0.211	0.125	0.173	0.604	0.610
EM (SBERT)	0.231	**0.254**	0.165	0.181	0.142	0.148	0.645	0.651
F1 (BM25)	0.282	0.303	0.314	**0.351**	0.188	**0.239**	0.639	**0.654**
F1 (SGPT)	0.278	0.282	0.301	0.306	0.153	0.216	0.604	0.610
F1 (SBERT)	0.294	**0.317**	0.244	0.262	0.172	0.175	0.645	0.651
Pre. (BM25)	0.288	0.301	0.331	**0.350**	0.195	**0.251**	0.639	**0.654**
Pre. (SGPT)	0.284	0.288	0.319	0.322	0.159	0.224	0.604	0.610
Pre. (SBERT)	0.298	**0.322**	0.256	0.273	0.176	0.179	0.645	0.651
Rec. (BM25)	0.285	0.295	0.316	**0.345**	0.196	**0.239**	0.639	**0.654**
Rec. (SGPT)	0.281	0.286	0.305	0.317	0.156	0.220	0.604	0.610
Rec. (SBERT)	0.299	**0.326**	0.255	0.278	0.180	0.183	0.645	0.651

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

Dataset	Metric	RaDIO	Base	wo-RAG	SR-RAG	FL-RAG	FS-RAG	FLARE
2WikiMultihopQA	EM	**0.254**	0.231	0.146	0.169	0.112	0.189	0.143
	F1	**0.317**	0.294	0.223	0.255	0.192	0.265	0.213
HotpotQA	EM	**0.246**	0.219	0.184	0.164	0.146	0.214	0.149
	F1	**0.351**	0.314	0.275	0.150	0.211	0.304	0.221
StrategyQA	Pre.	**0.654**	0.645	**0.659**	0.645	0.634	0.629	0.627
IIRC	EM	**0.196**	0.156	0.139	0.187	0.172	0.178	0.136
	F1	**0.239**	0.188	0.173	0.226	0.203	0.216	0.164

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

top 25 passages to ensure the quality of the generated responses. We ran our procedure on datasets containing 1000 data points each to obtain robust and reliable results.

Experimental Results

We discuss our experimental results, which aim to comprehensively evaluate the performance of RaDIO against various competitors across four datasets: 2WikiMultihopQA, HotpotQA, StrategyQA, and IIRC. For the case study, please refer to the code link provided in the Abstract.

Overall Results. We first focused on the impact of the retrieval method. As mentioned above, the methods we included in our experimental evaluation were a BM25, SGPT, and SBERT; we measured the EM, F1, Pre. and Rec. obtained by RaDIO with DRAGIN, which is currently the state of the art in dynamic RAG. The results obtained are reported in Table 1. The main lessons learned from our experiments are *a)* RaDIO, regardless of the retrieval module we consider, always performs better than DRAGIN on all the datasets examined and for all the evaluation measures. In some cases, the percentage improvement of RaDIO against DRAGIN exceeds 25%. *b)* The chosen retrieval module affects the performance of both RaDIO and DRAGIN: generally, the BM25 method achieves the best results, but on the 2WikiMultihopQA dataset it is outperformed by SBERT. SGPT is also less effective than both BM25 and SBERT.

In the 2WikiMultihopQA dataset, RaDIO in conjunction with BM25 achieves an EM score of 0.228 while DRAGIN achieves a score equal to 0.214 in the same experimental configuration. F1 score rises from 0.282 to 0.303, Pre. improves from 0.288 to 0.301, and Rec. enhances to 0.295. As already pointed out, the SBERT retrieval model achieves the most convincing results on the 2WikiMultihopQA dataset, and, more precisely, we report a further improvement in the EM score (which increases to 0.254), while F1 reaches 0.317, Pre. rises to 0.322, and Rec. rises to 0.326.

The results on the HotpotQA dataset are also noteworthy. With BM25 retrieval, RaDIO's EM score jumps from 0.219 to 0.246, F1 score increases from 0.314 to 0.351, Precision improves from 0.331 to 0.350, and Recall grows to 0.345. Using SBERT retrieval, RaDIO's EM score rises to 0.181, F1 score reaches 0.262, Precision climbs to 0.273, and Recall enhances to 0.278. Similar improvements also occur on IIRC and StrategyQA datasets.

Comparison with other RAG methods. We compared RaDIO with six competitors using EM and F1 scores for the 2WikiMultihopQA, HotPotQA, and IIRC datasets, and precision for the StrategyQA dataset. The results, summarized in Table 2, highlight several key points: *a)* RaDIO always performs best, regardless of the dataset we consider, and often achieves a significant improvement over DRAGIN. *b)* Methods based on static rules (i.e. SR-RAG and FL-RAG) show worse performance than methods that do not use RAG (e.g. wo-RAG on the HotpotQA dataset has a higher EM and F1 score than SR-RAG or FL-RAG). Such a result highlights the importance of choosing the right time to perform the re-

Matrix	2WikiMultihopQA		HotpotQA		IIRC		StrategyQA	
	Base	**RaDIO**	Base	**RaDIO**	Base	**RaDIO**	Base	**RaDIO**
Rc (BM25)	1.018	**1.015**	2.832	**1.041**	3.696	**1.209**	4.422	**1.131**
Rc (SGPT)	3.602	**1.016**	3.002	**1.304**	3.002	**1.242**	4.305	**1.122**
Rc (SBERT)	1.804	**1.035**	0.974	**1.078**	1.534	**1.014**	2.390	**1.268**
Gc (BM25)	2.038	**2.031**	6.372	**2.082**	7.431	**2.417**	8.849	**2.262**
Gc (SGPT)	7.208	**2.033**	6.007	**2.068**	6.009	**2.563**	8.613	**2.244**
Gc (SBERT)	3.982	**2.087**	2.725	**2.542**	3.206	**2.040**	5.601	**3.315**
Hc (BM25)	1.018	**1.015**	2.832	**1.041**	3.696	**1.209**	4.422	**1.131**
Hc (SGPT)	3.602	**1.016**	3.002	**1.304**	3.002	**1.242**	4.305	**1.122**
Hc (SBERT)	1.804	**1.035**	0.974	**1.078**	1.534	**1.014**	2.390	**1.268**
Tc (BM25)	523.763	**521.979**	694.887	**268.811**	959.080	**621.381**	894.252	**228.675**
Tc (SGPT)	1852.543	**522.494**	775.129	**267.144**	775.328	**668.590**	870.086	**226.875**
Tc (SBERT)	537.053	**386.106**	216.537	**200.988**	497.527	**471.739**	325.277	**123.398**
Sc (BM25)	36.530	**36.474**	34.372	**13.141**	47.099	**28.235**	59.893	**15.580**
Sc (SGPT)	123.930	**36.275**	40.494	**13.087**	36.826	**29.106**	60.191	**14.923**
Sc (SBERT)	30.908	**22.932**	10.629	**10.334**	25.094	**26.603**	23.592	**9.181**

Table 3: Compare the efficiency of various retrieval and generation methods in base and RaDIO across four different datasets.

trieval: an imprecise choice not only wastes computational resources but also does not lead to any quality improvement. *c)* FLARE often shows less brilliant results than all the other methods; we recall that FLARE calculates the probability of generating a token and, if this probability is below a certain threshold, it starts the retrieval. This rule seems to be ineffective in many datasets; conversely, the decision model implemented by RaDIO is richer and more sophisticated, taking into account several factors to detect important tokens.

Hence, RaDIO can effectively decide when to start retrieval operations and what to retrieve. *d)* Generally, single-round RAG perform worse than multi-round RAG methods. However, it should be noted that the performance of a RAG depends crucially on datasets and the evaluation metrics: for example, in the StrategyQA dataset, the differences in precision between the different models are negligible.

On the 2WikiMultihopQA dataset, RaDIO achieved state-of-the-art results with an EM of 0.254 and an F1 of 0.317, outperforming DRAGIN and wo-RAG (0.146). RaDIO also led on the HotpotQA and IIRC datasets, with an F1 of 0.351, surpassing DRAGIN and wo-RAG (0.275), demonstrating its effectiveness in handling complex queries.

However, on the StrategyQA dataset, wo-RAG (0.659) outperformed RaDIO and other RAG methods due to its simpler approach, which is more effective for binary classification. Additionally, Llama2-7B model's limited capacity might also restrict RaDIO's advanced features, making wo-RAG's straightforward method more efficient in this case.

Efficiency Comparison Experiment. Finally, We compared the efficiency of the Base method and RaDIO, and the results show that RaDIO consistently outperforms the Base method across multiple datasets and evaluation metrics.

On the 2WikiMultihopQA dataset, RaDIO achieved an Rc of 1.015 with BM25 retrieval, slightly lower than the Base method's Rc of 1.018. With SGPT retrieval, RaDIO's Rc decreased from 3.602 to 1.016, and with SBERT retrieval, it improved from 1.804 to 1.035. In the HotpotQA dataset, RaDIO's Rc was 1.041 with BM25 retrieval, significantly outperforming the Base method's Rc of 2.832. With SGPT retrieval, RaDIO's Rc decreased from 3.002 to 1.304, and with SBERT retrieval, it improved from 0.974 to 1.078.

In the IIRC dataset, RaDIO's Gc was 2.417 with BM25 retrieval, a notable improvement over the Base method's Gc of 7.431. With SGPT retrieval, RaDIO's Gc decreased from 6.009 to 2.563, and with SBERT retrieval, it decreased from 3.206 to 2.040. In the StrategyQA dataset, RaDIO's Tc was 228.675 with BM25 retrieval, significantly lower than the Base method's Tc of 894.252. With SGPT retrieval, RaDIO's Tc decreased from 870.086 to 226.875, and with SBERT retrieval, it decreased from 325.277 to 123.398.

Conclusions and Future Works

In this work, we introduce RaDIO, an innovative dynamic Retrieval-Augmented Generation (RAG) framework designed to address the real-time information needs of LLMs during text generation. RaDIO integrates two key components, namely, timely retrieval activation and precise query formulation, to achieve state-of-the-art performance across various knowledge-intensive benchmarks, significantly outperforming existing dynamic RAG methods.

Our experimental results highlight RaDIO's effectiveness in overcoming the limitations of existing dynamic RAG approaches, which often rely on static or pre-defined retrieval strategies. The reasons for RaDIO's success lie in its ability to make optimal decisions about when and what to retrieve, and these decisions fit well with the real-time information needs of the LLM.

Looking ahead, we envisage that future work can build upon our research by exploring new directions. One potential area is to extend RaDIO to multi-task learning scenarios, where a single model is trained on multiple tasks simultaneously. This could lead to even more robust and versatile models that can handle different NLP tasks. Additionally, integrating RaDIO with other NLP tasks, such as question answering, text summarization, and machine translation, could further demonstrate its effectiveness in various applications.

Acknowledgments

This work was supported by the National Key R&D Program of China under Grant No. 2022YFC3303600, the National Natural Science Foundation of China under Grant Nos. 62337001 and 62077015, the Zhejiang Provincial Philosophy and Social Sciences Planning Project under Grant No. 24NDJC191YB, and the Zhejiang Provincial Natural Science Foundation of China under Grant No. LY23F020010.

References

Abdelsalam, M.; Faramarzi, M.; Sodhani, S.; and Chandar, S. 2021. Iirc: Incremental implicitly-refined classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11038–11047.

Borgeaud, S.; Mensch, A.; Hoffmann, J.; Cai, T.; Rutherford, E.; Millican, K.; Driessche, G. V. D.; Lespiau, J.-B.; Damoc, B.; Clark, A.; et al. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*, 2206–2240. PMLR.

Chang, T. A.; Tomanek, K.; Hoffmann, J.; Thain, N.; Van Liemt, E.; Meier-Hellstern, K.; and Dixon, L. 2024. Detecting Hallucination and Coverage Errors in Retrieval Augmented Generation for Controversial Topics. *arXiv preprint arXiv:2403.08904*.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; and Gehrmann, S. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint*. ArXiv:2204.02311.

Geva, M.; Khashabi, D.; Segal, E.; Khot, T.; Roth, D.; and Berant, J. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9: 346–361.

Ho, X.; Nguyen, A.-K. D.; Sugawara, S.; and Aizawa, A. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023a. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12): 1–38.

Ji, Z.; Yu, T.; Xu, Y.; Lee, N.; Ishii, E.; and Fung, P. 2023b. Towards mitigating LLM hallucination via self reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 1827–1843.

Jiang, Z.; Gao, L.; Araki, J.; Ding, H.; Wang, Z.; Callan, J.; and Neubig, G. 2022. Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer. *arXiv preprint*. ArXiv:2212.02027.

Jiang, Z.; Xu, F. F.; Gao, L.; Sun, Z.; Liu, Q.; Dwivedi-Yu, J.; Yang, Y.; Callan, J.; and Neubig, G. 2023a. Active retrieval augmented generation. *arXiv preprint*. ArXiv:2305.06983.

Jiang, Z.; Xu, F. F.; Gao, L.; Sun, Z.; Liu, Q.; Dwivedi-Yu, J.; Yang, Y.; Callan, J.; and Neubig, G. 2023b. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*.

Khandelwal, U.; Levy, O.; Jurafsky, D.; Zettlemoyer, L.; and Lewis, M. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*.

Li, J.; Tang, T.; Zhao, W. X.; Wang, J.; Nie, J.-Y.; and Wen, J.-R. 2023. The web can be your oyster for improving large language models. *arXiv preprint arXiv:2305.10998*.

Lv, Y.; and Zhai, C. 2011. Adaptive term frequency normalization for BM25. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, 1985–1988.

Maynez, J.; Narayan, S.; Bohnet, B.; and McDonald, R. 2020. On faithfulness and factuality in abstractive summarization. *arXiv preprint*. ArXiv:2005.00661.

Miller, G. A. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11): 39–41.

Mondal, D.; Modi, S.; Panda, S.; Singh, R.; and Rao, G. S. 2024. Kam-cot: Knowledge augmented multimodal chain-of-thoughts reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 18798–18806.

Muennighoff, N. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Ram, O.; Levine, Y.; Dalmedigos, I.; Muhlgay, D.; Shashua, A.; Leyton-Brown, K.; and Shoham, Y. 2023. In-context retrieval-augmented language models. *arXiv preprint*. ArXiv:2302.00083.

Ramos, J.; et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, 29–48. Citeseer.

Rong, X. 2014. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*.

Salemi, A.; and Zamani, H. 2024. Towards a search engine for machines: Unified ranking for multiple retrieval-augmented large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 741–751.

Shi, W.; Min, S.; Yasunaga, M.; Seo, M.; James, R.; Lewis, M.; Zettlemoyer, L.; and tau Yih, W. 2023a. Replug: Retrieval-augmented black-box language models. *arXiv preprint*. ArXiv:2301.12652.

Shi, W.; Min, S.; Yasunaga, M.; Seo, M.; James, R.; Lewis, M.; Zettlemoyer, L.; and Yih, W.-t. 2023b. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Su, W.; Tang, Y.; Ai, Q.; Wu, Z.; and Liu, Y. 2024a. DRAGIN: Dynamic Retrieval Augmented Generation based on the Information Needs of Large Language Models.

Su, W.; Tang, Y.; Ai, Q.; Wu, Z.; and Liu, Y. 2024b. Dragin: Dynamic retrieval augmented generation based on the real-time information needs of large language models. *arXiv preprint arXiv:2403.10081*.

Su, W.; Wang, C.; Ai, Q.; Hu, Y.; Wu, Z.; Zhou, Y.; and Liu, Y. 2024c. Unsupervised real-time hallucination detection based on the internal states of large language models. *arXiv preprint*. ArXiv:2403.06448.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; and Azhar, F. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint*. ArXiv:2302.13971.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2022. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint*. ArXiv:2212.10509.

Vaswani, A. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Wang, B.; and Kuo, C.-C. J. 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.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.