Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity

Anonymous ACL submission

Abstract

Retrieval-Augmented Large Language Models (LLMs), which incorporate the non-parametric knowledge from external knowledge bases into LLMs, have emerged as a promising approach to enhancing response accuracy in several tasks, such as Question-Answering (QA). However, even though there are various approaches dealing with queries of different complexities, they either handle simple queries with unnecessary computational overhead or fail to adequately address complex multi-step queries; yet, not all user requests fall into only one of the simple or complex categories. In this work, we propose a novel adaptive QA framework, that can dynamically select the most suitable strategy for (retrieval-augmented) LLMs from the simplest to the most sophisticated ones based on the query complexity. Also, this selection process is operationalized with a classifier, which is a smaller LM trained to predict the complexity level of incoming queries with automatically collected labels, obtained from actual predicted outcomes of models and inherent inductive biases in datasets. This approach offers a balanced strategy, seamlessly adapting between the iterative and single-step retrieval-augmented LLMs, as well as the no-retrieval methods, in response to a range of query complexities. We validate our model on a set of open-domain QA datasets, covering multiple query complexities, and show that ours enhances the overall efficiency and accuracy of QA systems, compared to relevant baselines including the adaptive retrieval approaches.

1 Introduction

Recent Large Language Models (LLMs) (Brown et al., 2020; OpenAI, 2023; Touvron et al., 2023; Anil et al., 2023) have shown overwhelming performances across diverse tasks, including question-answering (QA) (Yang et al., 2018; Kwiatkowski et al., 2019). However, they still generate factually incorrect answers since their knowledge solely relies on their parametric memory (Kasai et al., 2022; Mallen et al., 2023). Meanwhile, memorizing all the (ever-changing) world knowledge may not be possible. To address this problem, retrieval-augmented LLMs (Borgeaud et al., 2022; Izacard et al., 2023; Shi et al., 2023), which incorporate non-parametric knowledge into LLMs with additional retrieval modules, have gained increasing attention. Specifically, these models access a knowledge base, serving as an extensive repository of information across various subjects and disciplines, to retrieve relevant information to the given input, and then incorporate this retrieved knowledge into LLMs, which enables the models to stay accurate and current with the world knowledge.

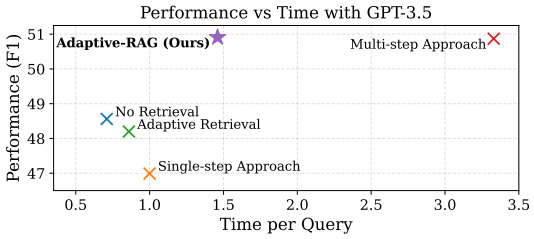

Figure 1: QA performance (F1) and efficiency (Time/Query) for different retrieval-augmented generation approaches. We use the GPT-3.5-Turbo-Instruct as the base LLM.

A particularly salient application of retrieval-augmented LLMs is in handling QA tasks, whose goal is to provide correct answers in response to user queries, especially those of high complexity. Early work on retrieval-augmented LLMs focuses primarily on single-hop queries (Lazaridou et al., 2022; Ram et al., 2023), where answers are typically found within a single document; therefore, this approach involves retrieving a relevant document based on the query and subsequently integrating this information into QA models to formulate a response. However, unlike this single-hop QA, some queries require connecting and aggregating multiple documents, which are, furthermore, often not answerable through a single-step process of retrieval-and-response. An example query is 'When did the people who captured Malakoff

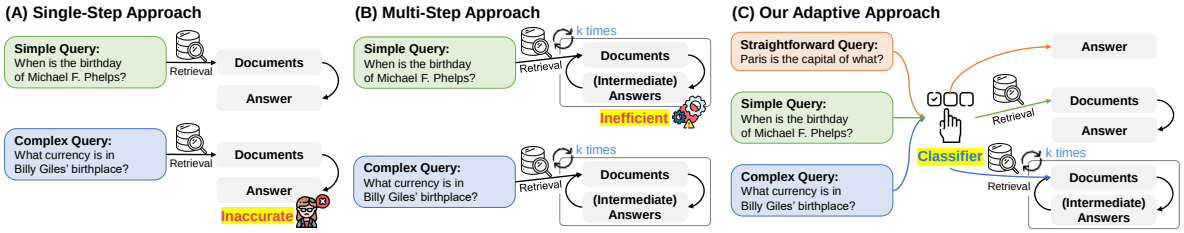

Figure 2: A conceptual comparison of different retrieval-augmented LLM approaches to question answering. (A) In response to a query, this single-step approach retrieves relevant documents and then generates an answer. However, it may not be sufficient for complex queries that require multi-step reasoning. (B) This multi-step approach iteratively retrieves documents and generates intermediate answers, which is powerful yet largely inefficient for the simple query since it requires multiple accesses to both LLMs and retrievers. (C) Our adaptive approach can select the most suitable strategy for retrieval-augmented LLMs, ranging from iterative, to single, to even no retrieval approaches, based on the complexity of given queries determined by our classifier.

come to the region where Philipsburg is located?', which requires four reasoning steps to solve. Therefore, to effectively handle such complex queries, recent studies have concentrated largely on multi-step and multi-reasoning QA, which requires iterative accesses to both LLMs and retrievers multiple times (Press et al., 2023; Trivedi et al., 2023), at the cost of heavy computational overheads.

Yet, we should rethink that: In a real-world scenario, are all the requests from users complex? Instead, users might often ask simple and straightforward questions, while only occasionally asking complex ones. Specifically, a query such as 'Paris is the capital of what?' is likely to be asked more frequently, compared to the aforementioned multi-step query, and this simpler query might also be easily answerable by the LLMs themselves, without accessing external knowledge. In other words, a multi-step QA approach could result in unnecessary computational overheads for simple queries, even though it is vital for complex queries (see Figure 2 (A)). On the other hand, handling complex queries with single-step-retrieval or even non-retrieval strategies would be largely insufficient (Figure 2 (B)). This suggests a need for an adaptive QA system, which can dynamically adjust the operational strategies of retrieval-augmented LLMs based on the query complexity. While some recent approaches are capable of doing this based on the frequency of entities in queries (Mallen et al., 2023) or the generated outputs from models for multi-step QA (Trivedi et al., 2023), they are suboptimal: former methods are overly simplistic, failing to consider multi-hop queries; meanwhile, later ones are excessively complex, terminating answer solving steps after several rounds of module accesses.

In this work, considering the diverse complexity levels of real-world queries, we argue that previous one-size-fits-all approaches might be inadequate to cover all of these variations. Instead, we propose to select the most suitable strategy from a range of (retrieval-augmented) LLMs, each of which is tailored to the specific complexity of the input query. Notably, a critical step in this process is pre-defining the query complexity, which is instrumental in determining the most fitting model to it. In this work, we operationalize this process with a novel classifier, which is a smaller model trained to predict the complexity level of incoming queries (see Figure 2 (c)). Moreover, we automatically collect its training datasets without human labeling, by leveraging the predicted outcomes (i.e., which models accurately respond to which queries) as well as by capitalizing on the inherent biases in existing datasets (i.e., samples in the datasets are designed for either single-step or multi-step QA scenarios). This proposed approach can offer a robust middle ground among the iterative LLM augmentation methods for complex queries, the single-step methods for simpler queries, and even no-retrieval-augmented methods for the most straightforward queries (answerable by LLMs themselves), thereby significantly enhancing the overall efficiency and accuracy of QA systems, as shown in Figure 1. We refer to our framework as Adaptive Retrieval-Augmented Generation (Adaptive-RAG).

We validate our Adaptive-RAG using benchmark open-domain QA datasets, covering a wide range of query complexity from single-hop (Rajpurkar et al., 2016; Joshi et al., 2017; Kwiatkowski et al., 2019) to multi-hop (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022b) queries. The experimental results show that our model significantly improves the overall accuracy and efficiency of QA, compared to the prior adaptive retrieval strategies, on multiple LLMs, such as GPT-3.5 (Brown et al., 2020) and FLAN-T5 series (Chung et al., 2022).

Our contributions and findings are threefold:

- We pointed out the realistic scenario of queries of varying complexities, and discovered that existing retrieval-augmented generation approaches tend to be overly simple or complex for them.

- We propose to adapt retrieval-augmented LLMs to the query complexity assessed by the classifier, which can enable the utilization of the most suitable approach tailored to each query.
- We show that our Adaptive-RAG is highly effective and efficient, balancing between the complexity and the simplicity for diverse queries.

2 Related Work

Open-domain QA Open-domain QA is the task of accurately answering a query by sourcing for the query-relevant documents, and then interpreting them to provide answers (Chen et al., 2017; Zhu et al., 2021), which, thus, generally involves two modules: a retriever (Karpukhin et al., 2020; Xiong et al., 2021) and a reader (Yang et al., 2019; Izacard and Grave, 2021). Along with the recent emergence of LLMs with superior reasoning capabilities thanks to their billion-sized parameters (Wei et al., 2022a), the synergy between the LLMs and retrievers has led to significant advancements (Lazaridou et al., 2022; Ram et al., 2023). Specifically, this integration has been shown to enhance Open-domain QA by mitigating the hallucination problem from LLMs through strengthened reasoning abilities of the reader, as well as utilizing the retrieved, external documents (Cho et al., 2023). Despite these advancements for single-hop retrieval-augmented LLMs, however, the complexity of some queries necessitates a more complex strategy.

Multi-hop QA Multi-hop QA is an extension of conventional Open-domain QA, which additionally requires the system to comprehensively gather and contextualize information from multiple documents (often iteratively), to answer more complex queries (Trivedi et al., 2022a; Yang et al., 2018). In the realm of multi-hop QA, the approach to iteratively access both LLMs and the retrieval module is generally employed. Specifically, Khattab et al. (2022), Press et al. (2023), Pereira et al. (2023) and Khot et al. (2023) proposed to first decompose the multi-hop queries into simpler single-hop queries, repeatedly access the LLMs and retriever to solve these sub-queries, and merge their solutions to formulate a complete answer. In contrast to this decomposition-based approach, other recent studies, such as Yao et al. (2023) and Trivedi et al. (2023), explored the interleaving of Chain-of-Thought reasoning (Wei et al., 2022b) — a method where a logical sequence of thoughts is generated — with document retrieval, repeatedly applying this process until the reasoning chain generates the answer. In addition, Jiang et al. (2023) introduced an approach for repeatedly retrieving new documents if the tokens within generated sentences have low confidence. However, the aforementioned methods overlooked the fact that, in real-world scenarios, queries are of a wide variety of complexities. Therefore, it would be largely inefficient to iteratively access LLMs and retrievers for every query, which might be simple enough to be solved with a single retrieval step or even only with an LLM itself.

Adaptive Retrieval To handle queries of varying complexities, the adaptive retrieval strategy aims to dynamically decide whether to retrieve documents or not, based on each query's complexity. In this vein, Mallen et al. (2023) proposed to decide the query's complexity level based on the frequency of its entities and suggested using the retrieval modules only when the frequency falls below a certain threshold. However, this approach, focusing solely on the binary decision to retrieve or not, may not be sufficient for solving more complex queries that require multiple reasoning steps. Concurrent to our work, Asai et al. (2023) suggested training a sophisticated model to dynamically retrieve, critique, and generate the text. However, we argue that all the aforementioned adaptive retrieval methods that rely on a single model might be suboptimal in handling a variety of queries of a range of different complexities since they tend to be either overly simple or complex for all the input queries, which gives rise to the new approach that can select the most suitable strategy of retrieval-augmented LLMs tailored to the query complexity.

3 Method

In this section, we describe our approach to adapting retrieval-augmented LLMs, by pre-determining the query complexity and then selecting the most fitting strategies for retrieval-augmented LLMs.

3.1 Preliminaries

We begin with preliminaries, formally introducing different strategies of retrieval-augmented LLMs.

Non Retrieval for QA Let us first define an LLM as a model $\texttt{LLM}$, which takes a sequence of tokens $\boldsymbol{x} = [x_1, x_2, ..., x_n]$ as an input and then generates a sequence of tokens $\boldsymbol{y} = [y_1, y_2, ..., y_n]$ as an output, which is formalized as follows: $\boldsymbol{y} = \texttt{LLM}(\boldsymbol{x})$. Then, in our problem setup for QA, $\boldsymbol{x}$ and $\boldsymbol{y}$ become the input query ($\boldsymbol{q}$) from the user and the

generated answer ($\bar{\boldsymbol{a}}$) from the LLM, respectively: $\boldsymbol{q} = \boldsymbol{x}$ and $\bar{\boldsymbol{a}} = \boldsymbol{y}$. Also, subsequently, the most naïve LLM-powered QA model can be represented as follows: $\bar{\boldsymbol{a}} = \texttt{LLM}(\boldsymbol{q})$. Ideally, $\bar{\boldsymbol{a}}$ should match the actual correct answer $\boldsymbol{a}$. This non-retrieval-based QA method is highly efficient and could be a somewhat promising approach to handling easy queries, as the size of LLMs becomes extremely large with its effect on storing a large amount of knowledge. However, this approach is largely problematic on queries that require precise or concurrent knowledge of specific people, events, or any subjects beyond the LLMs' internal knowledge.

Single-step Approach for QA To address the aforementioned scenarios where LLM may struggle with queries that are not answerable by LLM itself, we can utilize the external knowledge $\boldsymbol{d}$, which includes useful information for queries, retrieved from the external knowledge source $\mathcal{D}$ that could be an encyclopedia (e.g., Wikipedia) consisting of millions of documents. Specifically, to obtain such $\boldsymbol{d}$ from $\mathcal{D}$, a specific retrieval model is necessary, which returns documents based on their relevance with the given query. This process can be formulated as follows: $\boldsymbol{d} = \texttt{Retriever}(\boldsymbol{q}; D)$, where Retriever is the retrieval model, with $\boldsymbol{d} \in \mathcal{D}$. Here, we can use any off-the-shelf retriever (Robertson et al., 1994; Karpukhin et al., 2020).

After the retrieval step is done, we now have a pair of query $\boldsymbol{q}$ and its relevant documents $\boldsymbol{d}$. Then, in order to augment LLMs with this retrieved external knowledge, we can incorporate it into the input of LLMs, represented as follows: $\bar{\boldsymbol{a}} = \texttt{LLM}(\boldsymbol{q}, \boldsymbol{d})$. This process allows LLMs to gain access to external information contained in $\boldsymbol{d}$, which can provide the supplementary context that the internal knowledge of LLM lacks, which can subsequently improve the accurateness and concurrency of LLMs for QA.

Multi-step Approach for QA Even though the aforementioned single-step approach offers significant improvements over non-retrieval for $\boldsymbol{q}$ that requires external knowledge, it encounters notable limitations, particularly when dealing with complex queries that necessitate synthesizing information from multiple source documents and reasoning over them. This is where a multi-step approach and reasoning for QA become essential.

In this multi-step approach, LLM interacts with Retriever in several rounds, progressively refining its understanding of $\boldsymbol{q}$, until it formulates the final answer from findings accumulated across those multiple steps. Specifically, the process begins with the initial query $\boldsymbol{q}$, and at every retrieval step i, new documents $\boldsymbol{d}_i$ are retrieved from $\mathcal{D}$ and then incorporated into the input of LLMs, as follows: $\bar{\boldsymbol{a}}_i = \texttt{LLM}(\boldsymbol{q}, \boldsymbol{d}_i, \boldsymbol{c}_i)$, where the additional context $\boldsymbol{c}_i$ can be composed of previous documents and outcomes $(\boldsymbol{d}_1, \boldsymbol{d}_2, ..., \boldsymbol{d}_{i-1}, \bar{\boldsymbol{a}}_1, \bar{\boldsymbol{a}}_2, ..., \bar{\boldsymbol{a}}_{i-1})$, and $\boldsymbol{d}_i = \texttt{Retriever}(\boldsymbol{q}, \boldsymbol{c}_i; D)$[1]. We would like to note that this iterative, multi-step process enables LLM to construct a more comprehensive and extensive foundation to solve queries effectively, specifically adept at complex multi-hop queries where answers depend on interconnected pieces of information. However, it is important to recognize that this multi-step approach can be resource-intensive due to the repeated accesses to Retriever and LLM, which entail substantial computational costs.

3.2 Adaptive-RAG: Adaptive Retrieval-Augmented Generation

We now introduce our adaptive retrieval-augmented LLMs, which are built upon three different strategies described in the previous section, and which are designed to select the most suitable strategy according to the complexity of queries.

Adapting Retrieval-Augmented LLMs Note that in real-world scenarios, not all $\boldsymbol{q}$ from users have the same level of complexity, necessitating tailored strategies for handling each query. In other words, employing the most basic, non-retrieval-based approach $\texttt{LLM}(\boldsymbol{q})$ to respond to the complex query $\boldsymbol{q}$ would be also ineffective (Figure 2, A); conversely, using a more elaborate, multi-step approach $\texttt{LLM}(\boldsymbol{q}, \boldsymbol{d}, \boldsymbol{c})$ for simple $\boldsymbol{q}$ would be inefficient (Figure 2, B). Therefore, our adaptive framework is designed to dynamically adjust the query-handling strategy of retrieval-augmented LLMs, which is achieved by pre-determining the complexity of each query prior to attempting a solution. Notably, this framework can offer a robust middle ground with a range of solutions, from the simplest approach for the most straightforward queries, to the one-step approach for moderate queries, and up to the most comprehensive and rigorous approach for complex queries. In addition, since the operations of LLM and Retriever remain consistent re-

[1]It is worth noting that implementations of the LLM and retriever vary across different multi-step retrieval-augmented LLM approaches (Trivedi et al., 2023; Press et al., 2023; Yao et al., 2023); therefore, the context $\boldsymbol{c}_i$ may incorporate none, some, or all of previous documents and answers.

gardless of inputs to them, our method can seemingly go back and forth across queries of different complexities, without changing the internal model architectures or parameters during adaption.

Query Complexity Assessment To operationalize our adaptive retrieval-augmented LLM framework, we should determine the query complexity, and to achieve this, we propose to model a complexity classifier, whose goal is to return the appropriate complexity level of the given query. Specifically, given the query $\boldsymbol{q}$, our classifier can be formulated as follows: $o = \texttt{Classifier}(\boldsymbol{q})$, where `Classifier` is a smaller Language Model that is trained to classify one of three different complexity levels and o is its corresponding class label. In our classifier design, there are three class labels: 'A', 'B', and 'C', where 'A' indicates that $\boldsymbol{q}$ is straightforward and answerable by $\texttt{LLM}(\boldsymbol{q})$ itself, 'B' indicates that $\boldsymbol{q}$ has the moderate complexity where at least a single-step approach $\texttt{LLM}(\boldsymbol{q}, \boldsymbol{d})$ is needed, and 'C' indicates that $\boldsymbol{q}$ is complex, requiring the most extensive solution $\texttt{LLM}(\boldsymbol{q}, \boldsymbol{d}, \boldsymbol{c})$[2].

Training Strategy The remaining step is to train the smaller Language Model for `Classifier`, to accurately predict its complexity o in response to the given query $\boldsymbol{q}$. Yet, there is no annotated dataset available for query-complexity pairs. Hence, we propose to automatically construct the training dataset with two particular strategies.

To be specific, we first aim at labeling the query complexity based on the results from three different retrieval-augmented LLM strategies, in order to determine the label by its needs. For example, if the simplest non-retrieval-based approach correctly generates the answer, the label for its corresponding query is assigned 'A'. Also, to break the tie between different models in providing the label to the query, we provide a higher priority to a simpler model. In other words, if both single-step and multi-step approaches produce the same correct answer while the non-retrieval-based approach fails, we assign label 'B' to its corresponding query.

However, this labeling strategy has a limitation in that not all the queries are assigned the labels, since all three retrieval-augmented approaches may fail to generate the correct answer. On the other hand, the benchmark datasets may already have meaningful inductive biases about the most appropriate retrieval-augmented LLM strategies for their queries, considering the ways they are created (e.g., QA datasets that require sequential reasoning usually necessitate the multi-step approach; while queries of those with the labeled single documents can be ideally answerable with the single-step approach). Therefore, for those queries that remain unlabeled after the first labeling step, we assign 'B' to queries in single-hop datasets and 'C' to queries in multi-hop datasets. Finally, we train `Classifier` with these automatically-collected query-complexity pairs[3], by using a cross-entropy loss. Then, at inference, we can determine the complexity of the query, which is one of {'A', 'B', 'C'}, by forwarding it to `Classifier`: $o = \texttt{Classifier}(\boldsymbol{q})$.

4 Experimental Setups

In this section, we explain datasets, models, metrics, and implementation details. We further provide the additional details in Appendix A.

4.1 Datasets

In order to simulate a realistic scenario, where different queries have varying complexities, in the unified experimental setting, we use both the single-hop and multi-hop QA datasets simultaneously.

Single-hop QA For simpler queries, we use three benchmark single-hop QA datasets, which consist of queries and their associated documents containing answers, namely **1) SQuAD v1.1** (Rajpurkar et al., 2016), **2) Natural Questions** (Kwiatkowski et al., 2019), and **3) TriviaQA** (Joshi et al., 2017).

Multi-hop QA To consider more complex query scenarios, we use three benchmark multi-hop QA datasets, which require sequential reasoning over multiple documents, namely **1) MuSiQue** (Trivedi et al., 2022a), **2) HotpotQA** (Yang et al., 2018), and **3) 2WikiMultiHopQA** (Ho et al., 2020).

4.2 Models

We compare our Adaptive-RAG against relevant models, including three retrieval-augmented LLM strategies (in Section 3.1) and the adaptive retrieval approaches (Mallen et al., 2023; Asai et al., 2023), which can be grouped into one of three categories: Simple, Adaptive, and Complex. Specifically, Simple approaches include the **1) No Retrieval** and **2) Single-step Approach**-based meth-

[2]We consider three levels of query complexity, and leave the exploration of more fine-grained complexities as future work.

[3]As we automatically assign classifier labels, there might be errors in labeling and might be more advanced strategies to automatically assign labels, which we leave as future work.

Table 1: Averaged results on a collection of benchmark datasets for open-domain question answering including the single-hop and multi-hop queries, with different LLMs. Self-RAG* is trained with the different base LLM, namely LLaMA2 (Touvron et al., 2023); therefore, we compare the results of FLAN-T5-XL (3B) with the results from Self-RAG with LLaMA2 (7B) and the results of others with the results from Self-RAG with LLaMA2 (13B). We emphasize our results in bold, for easy comparisons.

		FLAN-T5-XL (3B)				FLAN-T5-XXL (11B)				GPT-3.5 (Turbo)			
Types	**Methods**	EM	F1	Step	Time	EM	F1	Step	Time	EM	F1	Step	Time
Simple	**No Retrieval**	14.87	21.12	0.00	0.11	17.83	25.14	0.00	0.08	35.77	48.56	0.00	0.71
	Single-step Approach	34.83	44.31	1.00	1.00	37.87	47.63	1.00	1.00	34.73	46.99	1.00	1.00
Adaptive	**Adaptive Retrieval**	23.87	32.24	0.50	0.56	26.93	35.67	0.50	0.54	35.90	48.20	0.50	0.86
	Self-RAG*	9.90	20.79	0.72	0.43	10.87	22.98	0.74	0.23	10.87	22.98	0.74	1.50
	Adaptive-RAG (Ours)	**37.17**	**46.94**	**2.17**	**3.60**	**38.90**	**48.62**	**1.35**	**2.00**	**37.97**	**50.91**	**1.03**	**1.46**
Complex	**Multi-step Approach**	39.00	48.85	4.69	8.81	40.13	50.09	2.13	3.80	38.13	50.87	2.81	3.33

ods. Adaptive approaches include the **3) Adaptive Retrieval** (Mallen et al., 2023), **4) Self-RAG** (Asai et al., 2023), and our **5) Adaptive-RAG**, which can adaptively perform retrieval based on the question complexity. For the **6) Multi-step Approach**, we use the most sophisticated state-of-the-art method (Trivedi et al., 2023) iteratively accessing both the retriever and LLM with Chain-of-Thought reasoning (Wei et al., 2022b), for every query. Note that models across different categories are not directly comparable. Yet, in the ideal setting, Adaptive approaches should be more effective than those in the Simple category while simultaneously being more efficient than the Complex one.

4.3 Evaluation Metrics

When it comes to evaluating adaptive models, it is essential to simultaneously consider both the task performance and efficiency along with their trade-offs. Thus, we report the results with four metrics, where two of them measure the effectiveness and the other two measure the efficiency. In particular, for effectiveness, we use F1 and EM following the standard evaluation protocol (Mallen et al., 2023; Baek et al., 2023), where F1 measures the number of overlapping words between the predicted answer and the ground truth and EM measures whether they are the same. For efficiency, we measure the number of retrieval-and-generate steps and the average time for answering each query relative to the one-step approach.

4.4 Implementation Details

For a fair comparison and following Mallen et al. (2023) and Trivedi et al. (2023), we use the same retriever, a term-based sparse retrieval model known as BM25 (Robertson et al., 1994), across all different models. For the external document corpus, we use different sources depending on the dataset type: the Wikipedia corpus preprocessed by Karpukhin et al. (2020) for single-hop datasets, and the preprocessed corpus by Trivedi et al. (2023) for multi-hop datasets. Regarding the LLMs that are used to generate answers, we use the FLAN-T5 series models (Chung et al., 2022) of XL with 3B parameters and XXL with 11B parameters, and the GPT-3.5 model (gpt-3.5-turbo-instruct). For the retrieval-augmented LLM design, we follow the implementation details from Trivedi et al. (2023), which include input prompts, instructions, and the number of test samples for evaluation (e.g., 500 samples per dataset). In our Adaptive-RAG, for the query-complexity classifier, we use and train the T5-Large model (Raffel et al., 2020). Specifically, the classifier is trained using the epoch that shows the best performance until 100 training iterations from the validation set, with the learning rate of 3e-5 and the AdamW (Loshchilov and Hutter, 2019) as an optimizer. Regarding its training data, we sample and annotate 400 queries from 6 datasets based on its inductive bias (single-hop for one-step approach and multi-hop for multi-step). In addition, we use predicted outcomes of three different strategies over 400 queries sampled from each dataset. Note that those queries used for classifier training do not overlap with the testing queries for QA.

5 Experimental Results and Analyses

In this section, we show the overall experimental results and offer in-depth analyses of our method.

Main Results First of all, Table 1 shows our main results averaged over all considered datasets, which corroborate our hypothesis that simple retrieval-augmented strategies are less effective than the complex strategy, while the complex one is significantly more expensive than the simple ones. In addition, we report the more granular results with FLAN-T5-XL on each of single-hop and multi-hop datasets in Table 2 (and more with different LLMs in Table 6 and Table 7 of Appendix), which are consistent with our hypothesis observed in Table 1.

Table 2: Results on each of a collection of datasets with FLAN-T5-XL (3B) as the LLM. We emphasize our results in bold.

Data	Types	Methods	SQuAD				Natural Questions				TriviaQA			
			EM	F1	Step	Time	EM	F1	Step	Time	EM	F1	Step	Time
Single-step	Simple	**No Retrieval**	3.60	10.50	0.00	0.11	14.20	19.00	0.00	0.13	25.00	31.80	0.00	0.13
		Single-step Approach	27.80	39.30	1.00	1.00	37.80	47.30	1.00	1.00	53.60	62.40	1.00	1.00
	Adaptive	**Adaptive Retrieval**	13.40	23.10	0.50	0.55	28.20	36.00	0.50	0.56	38.40	46.90	0.50	0.56
		Self-RAG*	2.20	11.20	0.63	0.50	31.40	39.00	0.63	0.17	12.80	29.30	0.68	0.45
		Adaptive-RAG (Ours)	**26.80**	**38.30**	**1.37**	**2.02**	**37.80**	**47.30**	**1.00**	**1.00**	**52.20**	**60.70**	**1.23**	**1.54**
	Complex	**Multi-step Approach**	24.40	35.60	4.52	9.03	38.60	47.80	5.04	10.18	53.80	62.40	5.28	9.22
Data	Types	Methods	MuSiQue				HotpotQA				2WikiMultiHopQA			
			EM	F1	Step	Time	EM	F1	Step	Time	EM	F1	Step	Time
Multi-step	Simple	**No Retrieval**	2.40	10.70	0.00	0.11	16.60	22.71	0.00	0.11	27.40	32.04	0.00	0.10
		Single-step Approach	13.80	22.80	1.00	1.00	34.40	46.15	1.00	1.00	41.60	47.90	1.00	1.00
	Adaptive	**Adaptive Retrieval**	6.40	15.80	0.50	0.55	23.60	32.22	0.50	0.55	33.20	39.44	0.50	0.55
		Self-RAG*	1.60	8.10	0.73	0.51	6.80	17.53	0.73	0.45	4.60	19.59	0.93	0.49
		Adaptive-RAG (Ours)	**23.60**	**31.80**	**3.22**	**6.61**	**42.00**	**53.82**	**3.55**	**5.99**	**40.60**	**49.75**	**2.63**	**4.68**
	Complex	**Multi-step Approach**	23.00	31.90	3.60	7.58	44.60	56.54	5.53	9.38	49.60	58.85	4.17	7.37

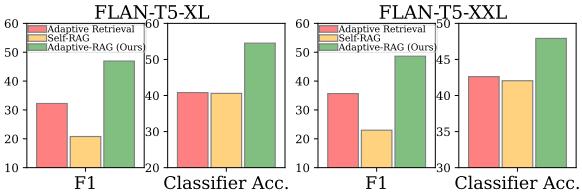

Figure 3: Performance on QA and query-complexity assessment of different adaptive approaches for retrieval-augmented LLMs with FLAN-T5 XL and XXL. For labeling the complexity of queries, we use the silver data annotated from the prediction outcomes of models (described in Section 3.2).

Table 3: Results on QA and complexity classification with varying the data annotation strategies for training the classifier.

Training Strategies	QA		Classifier (Accuracy)			
	F1	Step	All	No	One	Multi
Adaptive-RAG (Ours)	**46.94**	**1084**	**54.52**	**30.52**	**66.28**	**65.45**
w/o Binary	43.43	640	60.30	62.19	65.70	39.55
w/o Silver	48.79	1464	40.00	0.00	53.98	75.91

However, in a real-world scenario, not all users ask queries with the same level of complexity, which emphasizes the importance of the need for adaptive strategies. Note that among the adaptive strategies, our Adaptive-RAG shows remarkable effectiveness over the competitors (Table 1). This indicates that merely focusing on the decision of whether to retrieve or not is suboptimal. Also, as shown in Table 2, such simple adaptive strategies are particularly inadequate for handling complex queries in multi-hop datasets, which require aggregated information and reasoning over multiple documents. Meanwhile, our approach can consider a more fine-grained query handling strategy by further incorporating an iterative module for complex queries. Furthermore, in a realistic setting, we should take into account not only effectiveness but also efficiency. As shown in Table 1, compared to the complex multi-step strategy, our proposed adaptive strategy is significantly more efficient across all model sizes. This is meaningful in this era of LLMs, where the cost of accessing them is a critical factor for practical applications and scalability.

Classifier Performance To understand how the proposed classifier works, we analyze its performance across different complexity options. As Figure 3 shows, the classification accuracy of our Adaptive-RAG is better than that of the other adaptive retrieval baselines, which leads to overall QA performance improvements. This indicates that ours more accurately classifies the options with various granularities, including not retrieving documents, retrieving with a single access, and retrieving with multiple accesses.

Analyses on Training Data for Classifier We have shown that the classifier plays an important role in adaptive retrieval. Here, we further analyze the different strategies for training the classifier by ablating our full training strategy, which includes two approaches: generating silver data from predicted outcomes of models and utilizing inductive bias in datasets (see Section 3.2). As Table 3 shows, compared to the training strategy relying solely on the data derived from inductive bias, ours is significantly more efficient. This efficiency is partly because ours also takes into account the case that does not consider any documents at all, as also implied by the classification accuracy; meanwhile, queries in the existing datasets do not capture the information on whether the retrieval is not required. On the other hand, in the case of only using the silver data annotated from the correct predictions, while its overall classification accuracy is high, the overall QA performance implies that relying on the silver data may not be optimal. This may be

Table 4: Case study with GPT-3.5 (Turbo), where we represent the factual error in red and the accurate information in blue.

Dataset	Question	Adaptive Retrieval	Adaptive-RAG (Ours)
NQ (Single-hop)	Which famous corporate logo changed to a flat colour/color sans serif font in its first major change since 1999?	**Query Type:** B (Single-step Approach) **Reasoning:** The famous corporate logo that changed to a flat color sans-serif font in its first major change since 1999 is Microsoft. Microsoft changed its logo in 2012, replacing the italicized, multi-colored font with a simple, flat, grey font. So the answer is: Microsoft.	**Query Type:** A (Non Retrieval) **Reasoning:** The famous corporate logo that changed to a flat color sans serif font in its first major change since 1999 is Google. Google changed its logo in 2015. So the answer is: Google.
MuSiQue (Multi-hop)	Who is the child of the Italian navigator who explored the eastern coast of the continent César Gaytan was born in for the English?	**Query Type:** A (Non Retrieval) **Reasoning:** César Gaytan was born in the continent of North America. The Italian navigator who explored the eastern coast of North America for the English was Giovanni Caboto, also known as John Cabot. So the answer is: Giovanni Caboto/John Cabot.	**Query Type:** C (Multi-step Approach) **Reasoning:** Césarr Gaytan was born in Guadalajara, Jalisco, Mexico. The Italian navigator who explored the eastern coast of the continent for the English is John Cabot. John Cabot's son is Sebastian Cabot. So the answer is: Sebastian Cabot.

Table 5: Results with varying model sizes for classifiers.

	QA		Classifier (Accuracy)			
Sizes	**F1**	**Step**	**All**	**No**	**One**	**Multi**
Small (60M)	45.83	964	53.48	26.65	70.62	53.18
Base (223M)	45.97	983	53.41	26.42	69.46	56.82
Large (770M)	**46.94**	**1084**	**54.52**	30.52	66.28	65.45

because this silver data does not cover complexity labels on incorrectly predicted queries, which leads to lower generalization effect on queries relevant to them. Meanwhile, by also incorporating complexity labels from dataset bias (single-hop vs multi-hop), the classifier becomes more accurate in predicting multi-hop queries, leading to the better performance. It is worth noting that our automatic labeling strategies are two particular instantiations for training the classifier, and that there could be other instantiations, which we leave as future work.

Analyses on Classifier Size To investigate the sensitivity of our classifier according to its varying sizes, we conducted further experiments. As shown in Table 5, we observe no significant performance differences among classifiers of various sizes, even with reduced complexity and fewer parameters in smaller classifiers. This indicates that our proposed classifier can contribute to resource-efficient settings in real-use cases with smaller sizes without compensating performance.

Case Study We conduct a case study to qualitatively compare our Adaptive-RAG against Adaptive Retrieval. Table 4 shows the classified complexity and the query handling patterns for both simple and complex questions. First, for the simple single-hop question, our Adaptive-RAG identifies that it is answerable by only the LLM's parametric knowledge about 'Google'. In contrast, Adaptive Retrieval fetches additional documents, leading to longer processing times and occasionally producing incorrect responses due to the inclusion of partially irrelevant information about 'Microsoft'. Meanwhile, when faced with a complex question, our Adaptive-RAG seeks out relevant information, including details like 'a son of John Cabot', which may not have been stored in LLMs, while Adaptive Retrieval fails to request such information from external sources, resulting in inaccurate answers.

6 Conclusion

In this work, we proposed the Adaptive Retrieval-Augmented Generation framework, referred to as Adaptive-RAG, to handle queries of various complexities. Specifically, Adaptive-RAG is designed to dynamically adjust its query handling strategies in the unified retrieval-augmented LLM based on the complexity of queries that they encounter, which spans across a spectrum of the non-retrieval-based approach for the most straightforward queries, to the single-step approach for the queries of moderate complexity, and finally to the multi-step approach for the complex queries. The core step of our Adaptive-RAG lies in determining the complexity of the given query, which is instrumental in selecting the most suitable strategy for its answer. To operationalize this process, we trained a smaller Language Model with query-complexity pairs, which are automatically annotated from the predicted outcomes and the inductive biases in datasets. We validated our Adaptive-RAG on a collection of open-domain QA datasets, covering the multiple query complexities including both the single- and multi-hop questions. The results demonstrate that our Adaptive-RAG enhances the overall accuracy and efficiency of QA systems, allocating more resources to handle complex queries while efficiently handling simpler queries, compared to the existing one-size-fits-all approaches that tend to be either minimalist or maximalist over varying query complexities.

Limitations

While our Adaptive-RAG shows clear advantages in effectiveness and efficiency by determining the query complexity and then leveraging the most suitable approach for tackling it, it is important to recognize that there still exists potential avenues for improving the classifier from the perspectives of its training datasets and architecture. Specifically, as there are no available datasets for training the query-complexity classifier, we automatically create new data based on the model prediction outcomes and the inductive dataset biases. However, our labeling process is one specific instantiation of labeling the query complexity, and it may have the potential to label queries incorrectly despite its effectiveness. Therefore, future work would create new datasets that are annotated with a diverse range of query complexities, in addition to the labels of question-answer pairs. Also, as shown in Figure 3, there is still room to improve the effectiveness of the classifier. In other words, our classifier design based on the smaller LM is the initial, simplest instantiation for classifying the query complexity, and based upon it, future work may improve the classifier architecture and its performance, which will positively contribute to the QA performance.

Ethics Statement

The experimental results on Adaptive-RAG validate its applicability in realistic scenarios, where a wide range of diverse user queries exists. Nonetheless, given the potential diversity of real-world user inputs, it is crucial to also consider scenarios where these inputs might be offensive or harmful. We should be aware that such inputs could lead to the retrieval of offensive documents and the generation of inappropriate responses by the retrieval-augmented LLMs. To address this challenge, developing methods to detect and manage offensive or inappropriate content in both user inputs and retrieved documents within the retrieval-augmented framework is essential. We believe that this is a critical area for future work.

References

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernández Ábrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan A. Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vladimir Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, and et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Self-reflective retrieval augmented generation. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Jinheon Baek, Soyeong Jeong, Minki Kang, Jong Park, and Sung Ju Hwang. 2023. Knowledge-augmented language model verification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 1720–1736. Association for Computational Linguistics.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 -*

August 4, Volume 1: Long Papers, pages 1870–1879. Association for Computational Linguistics.

Sukmin Cho, Jeongyeon Seo, Soyeong Jeong, and Jong C. Park. 2023. Improving zero-shot reader by reducing distractions from irrelevant documents in open-domain question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 3145–3157. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

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, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 6609–6625. International Committee on Computational Linguistics.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 874–880. Association for Computational Linguistics.

Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24:251:1–251:43.

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 *EMNLP 2023*.

Mandar Joshi, Eunsol Choi, Daniel S. 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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1601–1611. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, November 16-20, 2020*. Association for Computational Linguistics.

Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir R. Radev, Noah A. Smith, Yejin Choi, and Kentaro Inui. 2022. Realtime QA: what's the answer right now? *arXiv preprint arXiv:2207.13332*.

Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *arXiv preprint arXiv.2212.14024*, abs/2212.14024.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv preprint arXiv:2203.05115*.

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient one-pass end-to-end entity linking for questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6433–6441. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 9802–9822. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pages 8024–8035.

Jayr Alencar Pereira, Robson do Nascimento Fidalgo, Roberto de Alencar Lotufo, and Rodrigo Frassetto Nogueira. 2023. Visconde: Multi-document QA with GPT-3 and neural reranking. In *Advances in Information Retrieval - 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2-6, 2023, Proceedings, Part II*, volume 13981 of *Lecture Notes in Computer Science*, pages 534–543. Springer.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

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*.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994*, volume 500-225 of *NIST Special Publication*, pages 109–126. National Institute of Standards and Technology (NIST).

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023. REPLUG: retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022a. 9835 musique: Multi-hop questions via single-hop question composition. *Trans. Assoc. Comput. Linguistics*, 10:539–554.

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

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), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 10014–10037. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin

Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos*, pages 38–45. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with bertserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 72–77. Association for Computational Linguistics.

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. 2021. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*.

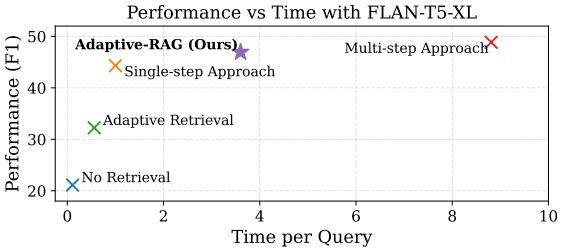

Figure 4: QA performance (F1) and efficiency (Time/Query) for different retrieval-augmented generation approaches. We use the FLAN-T5-XL (3B) as the base LLM.

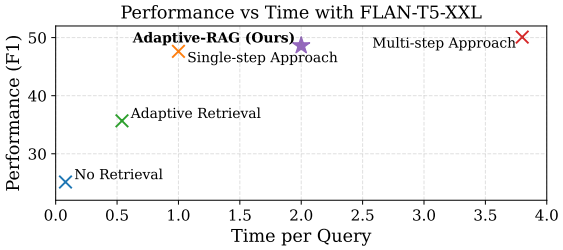

Figure 5: QA performance (F1) and efficiency (Time/Query) for different retrieval-augmented generation approaches. We use the FLAN-T5-XXL (11B) as the base LLM.

A Additional Experimental Setups

A.1 Datasets

We use publicly open datasets for both single-hop and multi-hop QA datasets, referring to Karpukhin et al. (2020) and Trivedi et al. (2023), respectively. We describe the characteristics of each dataset:

1) SQuAD v1.1 (Rajpurkar et al., 2016) is created through a process where annotators write questions based on the documents they read.

2) Natural Questions (Kwiatkowski et al., 2019) is constructed by real user queries on Google Search.

3) TriviaQA (Joshi et al., 2017) comprises trivia questions sourced from various quiz websites.

4) MuSiQue (Trivedi et al., 2022a) is collected by compositing multiple single-hop queries, to form queries spanning 2-4 hops.

5) HotpotQA (Yang et al., 2018) is constructed by having annotators create questions that link multiple Wikipedia articles.

6) 2WikiMultiHopQA (Ho et al., 2020) is derived from Wikipedia and its associated knowledge graph path, needing 2-hops.

A.2 Models

We describe the details of models as follows:

1) No Retrieval. This approach uses only the LLM itself, to generate the answer to the given query.

2) Single-step Approach. This approach first retrieves the relevant knowledge with the given query from the external knowledge sources and then augments the LLM with this retrieved knowledge to generate the answer, which iterates only once.

3) Adaptive Retrieval. This baseline (Mallen et al., 2023) adaptively augments the LLM with the retrieval module, only when the entities appearing in queries are less popular. To extract entities, we use the available entity-linking method (Li et al., 2020), namely BLINK, for questions.

4) Self-RAG. This baseline (Asai et al., 2023) trains the LLM to adaptively perform retrieval and generation, where the retrieval is conducted once it predicts the special retrieval token above a certain threshold, and the answer generation is followed.

5) Adaptive-RAG. This is our model that adaptively selects the retrieval-augmented generation strategy, smoothly oscillating between the non-retrieval, single-step approach, and multi-step approaches[4] without architectural changes, based on the query complexity assessed by the classifier.

6) Multi-step Approach. This approach (Trivedi et al., 2023) is the multi-step retrieval-augmented LLM, which iteratively accesses both the retriever and LLM with interleaved Chain-of-Thought reasoning (Wei et al., 2022b) repeatedly until it derives the solution or reaches the maximum step number.

A.3 Implementation Details

For computing resources, we use A100 GPUs with 80GB memory. In addition, due to the significant costs associated with evaluating retrieval-augmented generation models, we perform experiments with a single run. Finally, we implemented models using PyTorch (Paszke et al., 2019) and Transformers library (Wolf et al., 2020).

B Additional Experimental Results

Performance vs Time We further provide a comparison of different retrieval-augmented generation approaches with FLAN-T5-XL and FLAN-T5-XXL models in Figure 4 and Figure 5, respectively, in the context of performance and efficiency trade-offs. Similar to the observation made from the GPT-3.5 model in Figure 1, our proposed Adaptive-RAG is significantly more effective as well as efficient.

Performance per Dataset In addition to detailing the performance of each dataset with the FLAN-T5-XL model, as shown in Table 2, we also present

[4]For the multi-step approach, we use the state-of-the-art question answering strategy from IRCoT (Trivedi et al., 2023).

Table 6: Results on each of a collection of datasets with FLAN-T5-XXL (11B) as the LLM. We emphasize our results in bold.

			SQuAD				Natural Questions				TriviaQA			
Data	**Types**	**Methods**	EM	F1	Step	Time	EM	F1	Step	Time	EM	F1	Step	Time
Single-step	**Simple**	**No Retrieval**	7.00	14.40	0.00	0.08	18.80	25.50	0.00	0.08	32.80	39.20	0.00	0.08
		Single-step Approach	28.80	40.80	1.00	1.00	41.40	51.20	1.00	1.00	56.00	64.70	1.00	1.00
	Adaptive	**Adaptive Retrieval**	15.60	25.60	0.50	0.54	31.00	39.70	0.50	0.54	44.80	52.20	0.50	0.54
		Self-RAG*	1.60	11.90	0.59	0.31	39.20	47.10	0.75	0.09	14.60	33.70	0.76	0.22
		Adaptive-RAG (Ours)	**27.80**	**39.80**	**1.17**	**1.50**	**41.20**	**51.00**	**1.00**	**1.00**	**52.00**	**60.30**	**1.03**	**1.33**
	Complex	**Multi-step Approach**	24.60	36.90	2.13	3.83	39.60	49.60	2.16	3.94	52.60	61.10	2.17	4.03
			MuSiQue				**HotpotQA**				**2WikiMultiHopQA**			
Data	**Types**	**Methods**	EM	F1	Step	Time	EM	F1	Step	Time	EM	F1	Step	Time
Multi-step	**Simple**	**No Retrieval**	4.20	13.40	0.00	0.08	17.40	25.44	0.00	0.09	26.80	32.93	0.00	0.08
		Single-step Approach	16.80	25.70	1.00	1.00	37.60	49.27	1.00	1.00	46.60	54.13	1.00	1.00
	Adaptive	**Adaptive Retrieval**	8.40	17.80	0.50	0.54	26.60	36.01	0.50	0.54	35.20	42.68	0.50	0.54
		Self-RAG*	1.20	8.20	0.68	0.27	5.60	17.86	0.76	0.26	3.00	19.14	0.90	0.25
		Adaptive-RAG (Ours)	**20.60**	**28.50**	**1.89**	**3.12**	**44.20**	**54.78**	**1.58**	**2.53**	**47.60**	**57.36**	**1.46**	**2.55**
	Complex	**Multi-step Approach**	19.40	27.50	2.09	3.66	47.00	57.81	2.08	3.73	57.60	67.65	2.17	3.63

Table 7: Results on each of a collection of datasets with GPT-3.5 (Turbo) as the LLM. We emphasize our results in bold.

			SQuAD				Natural Questions				TriviaQA			
Data	**Types**	**Methods**	EM	F1	Step	Time	EM	F1	Step	Time	EM	F1	Step	Time
Single-step	**Simple**	**No Retrieval**	16.00	29.20	0.00	0.62	39.80	55.70	0.00	0.56	64.00	75.60	0.00	0.68
		Single-step Approach	18.00	33.80	1.00	1.00	32.40	46.80	1.00	1.00	55.20	66.50	1.00	1.00
	Adaptive	**Adaptive Retrieval**	15.40	30.00	0.50	0.81	36.40	51.20	0.50	0.78	62.00	71.90	0.50	0.84
		Self-RAG*	1.60	11.90	0.59	1.91	39.20	47.10	0.75	0.52	14.60	33.70	0.76	1.59
		Adaptive-RAG (Ours)	**19.80**	**34.40**	**0.87**	**1.21**	**36.80**	**52.00**	**0.68**	**0.86**	**62.40**	**73.80**	**0.22**	**0.79**
	Complex	**Multi-step Approach**	17.40	31.50	2.50	3.24	35.60	49.70	2.58	3.79	54.80	67.10	2.30	2.65
			MuSiQue				**HotpotQA**				**2WikiMultiHopQA**			
Data	**Types**	**Methods**	EM	F1	Step	Time	EM	F1	Step	Time	EM	F1	Step	Time
Multi-step	**Simple**	**No Retrieval**	20.40	31.30	0.00	0.81	37.40	51.04	0.00	0.74	37.00	48.50	0.00	0.90
		Single-step Approach	16.40	26.70	1.00	1.00	39.60	50.44	1.00	1.00	46.80	57.69	1.00	1.00
	Adaptive	**Adaptive Retrieval**	18.80	30.30	0.50	0.90	38.60	50.70	0.50	0.87	44.20	55.11	0.50	0.95
		Self-RAG*	1.20	8.20	0.68	1.66	5.60	17.86	0.76	1.67	3.00	19.14	0.90	1.81
		Adaptive-RAG (Ours)	**21.80**	**32.60**	**1.90**	**2.29**	**40.40**	**52.56**	**0.93**	**1.48**	**46.60**	**60.09**	**1.59**	**2.23**
	Complex	**Multi-step Approach**	23.00	32.50	3.41	3.61	45.80	58.36	2.73	3.18	52.20	66.08	3.36	3.35

the results for each dataset with the FLAN-T5-XXL and GPT-3.5 models in Table 2 and Table 7, respectively. The experimental results show that our Adaptive-RAG consistently balances between efficiency and accuracy. It is worth noting that while the GPT-3.5 model performs effectively in addressing straightforward queries even without document retrieval, it benefits significantly from our Adaptive-RAG in terms of effectiveness when solving complex multi-hop queries.