Adaptive RAG: A Literature Overview

Dominic P. Fischer
University of Zurich
dominicphilipp.fischer@uzh.ch

Abstract

Retrieval-Augmented Generation (RAG) enhances the factual accuracy of language models by incorporating external knowledge. However, traditional RAG systems apply fixed retrieval strategies, leading to unnecessary computational overhead for simple queries and insufficient support for complex or long-tail queries, i.e. containing niche topics. Adaptive RAG approaches address this by dynamically adjusting retrieval and generation based on the characteristics of the query, answer, or context. This paper provides a comprehensive overview of the current landscape of adaptive RAG systems, categorising them into heuristic, learning-based, and uncertainty-driven approaches. I analyse key contributions across these categories, comparing strategies, performance and efficiency. My findings underscore the potential of adaptive frameworks to improve both answer quality and resource usage, while highlighting open challenges and future directions.

1 Introduction

Adaptive RAG refers to any type of RAG-setup which is not fixed and static, but which adapts its retrieval and generation strategy based on the circumstances; hence the name[1]. These approaches can be divided into two main branches: A RAG-system can either be adapted to specific users or scenarios, i.e. favouring certain sources or answering in a particular style, or it can be adapted based on query or answer characteristics, i.e. when to retrieve, what to retrieve and how to integrate it, in order to optimise answer quality and resource consumption. The motivation for this approach can be summarised as follows: current systems "either handle simple queries with unnecessary computational overhead or fail to adequately address complex multi-step queries" (Jeong et al., 2024), thus not reaching a good balance between answer accuracy and computational overhead.

Current seminal research—and therefore, this paper—focuses primarily on dynamically adjusting retrieval and generation strategies according to the complexity or nature of the query or answer, rather than directly adapting to individual users. Adaptive approaches detailed below, such as thresholds to decide when to retrieve, could be tuned to specific users or domains—a domain expert might want a different level of depth and accuracy than a novice asking a casual question, and in e.g. the medical domain, having grounded facts is paramount; retrieved documents could be re-ranked as a function of what is important given the situation, and so on.

This paper aims to give an overview of the most impactful works on adaptive RAG in its primary sense, and to analyse and compare them from different angles.

2 Background

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) is a framework that combines two components: a *retriever*, which fetches relevant documents from a large corpus (e.g. Wikipedia), and a *generator*, which uses those documents to produce an answer. The retriever is typically a Dense Passage Retriever (DPR) (Karpukhin et al., 2020), which is trained to retrieve semantically relevant documents. DPR uses a dual-encoder architecture: one encoder processes the input query, and another processes candidate passages. Each input is mapped to a fixed-size vector in a shared embedding space and retrieval is then performed via similarity of the query vector and the document vectors. The generator, typically a Language Model (LM), then conditions on both the query and the retrieved documents to generate a response. This setup allows the model to access external knowl-

[1] Note that Adaptive Learning in RAG, albeit sometimes used synonymously, will be used to denote the subset of learning-based adaptive approaches in this paper.

edge at inference time, improving performance on tasks that require factual grounding.

In the original RAG setup, retrieval is always performed in the same way for every query—regardless of how simple or complex it is. The model does not decide when to retrieve, what to retrieve, or how many times to retrieve; all queries go through the same fixed retrieval-and-generation pipeline. While this improves factuality compared to models that rely only on internal parameters, it lacks flexibility and may also introduce misleading knowledge snippets. There is no adaptation based on query difficulty, model confidence, or user context—limitations that more recent adaptive RAG systems aim to overcome.

3 Survey and Taxonomy of Adaptive RAG approaches

3.1 Heuristics-based Adaptive Systems

The most notable contribution of this kind of system is Mallen et al. (2022), termed ***Adaptive Retrieval***. They found that "LMs struggle with less popular factual knowledge, and that retrieval augmentation helps significantly in these cases". However, retrieval can mislead LMs when the knowledge in question is popular. On the other hand, scaling up LMs improves accurate rendering of popular knowledge, but does not have the same effect of memorising factual knowledge in the so-called long-tail. This is what led them to devise a simple yet effective methodology where they retrieve only when information is suspected not to be baked into the parameters of a LM (non-parametric knowledge). Their approach and findings are described below.

They viewed factual knowledge as triplets of subject, relationship, object, e.g. *Louisiana, capital of, Baton Rouge*. They formulated the corresponding questions as follows: *What is the capital of Louisiana?*. If any substring in the answer matches with *Baton Rouge*, the answer is counted as correct. Hypothesising that accurate memorisation of factual knowledge in LMs correlates with the presence of said knowledge on the web and given their knowledge triplets, that led them to the following two measures of memorisation difficulty:

- Subject entity popularity: They use the monthly page views of an entity's Wikipedia page as a proxy for an entity's popularity.
- Relationship type: Common relationship types were found to be more easily memorised than less discussed ones—they demonstrated a dependence of performance on the relationship type, even given the same entities.

They used their own dataset PopQA, where they sampled random knowledge triplets with 16 different relationship types from Wikidata, and verbalised them using manual templates as described in the example above. They also use large parts of the dataset EntityQuestions, filtering out data where the subject entity could not be uniquely identified within Wikidata.

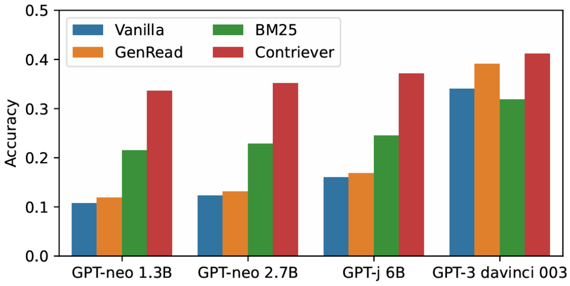

Figure 1: PopQA accuracy of vanilla and retrieval-augmented LMs (BM25, Contriever). Sourced from Mallen et al. (2022), Figure 7.

They then evaluated multiple models in vanilla (purely parametric) and retrieval-augmented configurations across both datasets. Their results show that (i), retrieval augmentation generally improves performance, especially that of smaller models (cf. Figure 1), and that (ii), retrieval augmentation (dashed lines) is highly beneficial for less popular entities, and parametric memory (solid lines) competitive for more popular ones (cf. Figure 2).

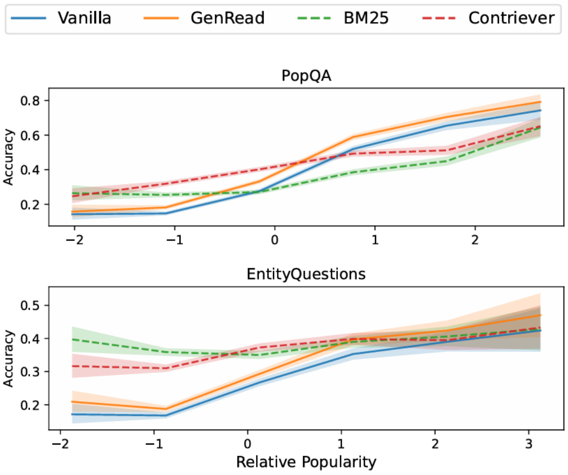

Figure 2: GPT-3 davinci-003 accuracy vs. relative popularity. Sourced from (Mallen et al., 2022), Figure 8.

They further show that (iii), for 10% of questions, retrieval leads to wrong answers where the LM would otherwise answer correctly, illustrating the need for adaptive retrieval.

Based on these findings, the only used retrieval for "questions whose popularity is lower than a threshold". The threshold was chosen to maximise accuracy when training on a development set, and is different for each relationship type. They found that "Adaptive Retrieval robustly outperforms approaches that always retrieve, especially for larger LMs" (cf. Figure 3), while also reducing latency and API costs.

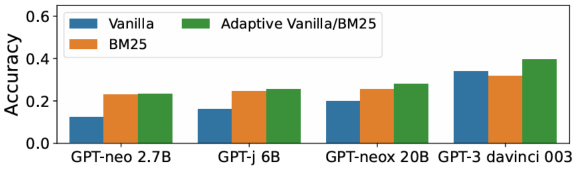

Figure 3: POPQA performance of different GPT models and different retrieval methods. Source from (Mallen et al., 2022), Figure 9.

In summary, (Mallen et al., 2022) provide not only a strong empirical motivation for adaptive retrieval in RAG systems by quantifying when retrieval is actually beneficial, but also a straightforward yet effective implementation.

A comparable approach is found in Zhang et al. (2024). They create their own dataset RetrievalQA, covering not only long-tail, but also new world, i.e. recent knowledge, ensuring that the knowledge necessary to answer the questions is absent from LLMs. Their approach is heuristic in the sense that they ask LLMs to decide themselves whether they need retrieval or not.

In order to enhance models' time awareness, and given that time-sensitive knowledge often comes with time specifications (e.g. "last week", "recent"), they included "Today is current_date()" in the model instruction. They combined that with having some [Yes] and [No] demonstrations in a few-shot prompting style, showing the model when it should or should not invoke retrieval. Across five different models, their approach showed significant improvements over the vanilla prompting counterparts, with an average gain of 14.9% for retrieval (realising that it should retrieve) and 6.7% for answer accuracy. Interestingly, for the GPT-models (3.5 and 4), they noted the biggest improvements of their approach (+37%/+15% resp. +16%/+9% retrieval/answer accuracy), perhaps indicating a certain overconfidence of the GPT-family models.

3.2 Learning-based Adaptive Systems

Jeong et al. (2024) provided an impactful and straighforward approach to adaptive RAG, even if not quite as known as Asai et al. (2023)'s ***Self-RAG***. Both approaches are discussed below.

Jeong et al. (2024) provide a strategy for answering simple questions efficiently and complex questions with the necessary retrieval step(s). They propose a novel adaptive framework, ***Adaptive-RAG***, that decides on one of three strategies—no retrieval, single retrieval and multi retrieval—based on input query complexity. To that end, they trained a ternary classifier, for which they needed training data. However, such data did not exist—they constructed the dataset automatically as follows:

- For each query, try the three different strategies and assign the label of the simplest strategy that answers the question correctly.
- If all three strategies fail, the label is assigned based on the type of question or dataset.

		FLAN-T5-XL (3B)					GPT-3.5 (Turbo)				
Types	**Methods**	EM	F1	Acc	Step	Time	EM	F1	Acc	Step	Time
Simple	**No Retrieval**	14.87	21.12	15.97	0.00	0.11	35.77	48.56	44.27	0.00	0.71
	Single-step Approach	34.83	44.31	38.87	1.00	1.00	34.73	46.99	45.27	1.00	1.00
Adaptive	**Adaptive Retrieval**	23.87	32.24	26.73	0.50	0.56	35.90	48.20	45.30	0.50	0.86
	Self-RAG*	9.90	20.79	31.57	0.72	0.43	10.87	22.98	34.13	0.74	1.50
	Adaptive-RAG (Ours)	**37.17**	**46.94**	**42.10**	**2.17**	**3.60**	**37.97**	**50.91**	**48.97**	**1.03**	**1.46**
Complex	**Multi-step Approach**	39.00	48.85	43.70	4.69	8.81	38.13	50.87	49.70	2.81	3.33
Oracle	**Adaptive-RAG w/ Oracle**	45.00	56.28	49.90	1.28	2.11	47.70	62.80	58.57	0.50	1.03

Figure 4: Averaged results on six benchmark datasets for open-domain QA, including single- and multi-hop. Self-RAG is trained with LLaMA2 (7B and 11B); results from FLAN-T5-XL (3B) are compared with Self-RAG LLaMA2 7B, the GPT-3.5 Turbo (approx. 20B parameters) results with Self-RAG LLaMA2 13B. Their own results (Adaptive-RAG) are emphasised in **bold**. Adapted from Jeong et al. (2024), Figure 1.

They argued in favour of their decision to differentiate between three instead of just two different strategies (retrieval/no retrieval) due to the simple binary decision not being fine-grained enough for the many varying complexities questions might exhibit. And indeed, amongst the adaptive strategies, they reported "remarkable effectiveness over the competitors" (cf. Figure 4). This holds true especially for smaller models, while for GPT-3.5-turbo, the improvements are not quite as striking.

The performance numbers of Self-RAG, especially the EM scores, appear rather low in all tasks, compared to the scores Asai et al. (2023) themselves report, with Self-RAGs underperforming even some single-step or no-retrieval approaches. A question mark has to be set behind the implementation comparability, as Asai et al. (2023) report better scores on TriviaQA than Jeong et al. (2024) report on its more challenging twin TriviaQA-unfiltered, which includes distractor documents. However, cross-checking with other results such as the ones reported by Yao et al. (2024) (cf. Figures 12, 13) confirm low EM scores and low or highly data-dependent performance of Self-RAG.

Asai et al. (2023)'s Self-RAG approach works as follows: "We train an arbitrary LM in an end-to-end manner to learn to reflect on its own generation process given a task input by generating both task output and intermittent special tokens (i.e., reflection tokens)". There are *retrieval* and different types of *critique* tokens. If the model thinks that retrieval would be useful, a retrieval token is output, and retrieval is triggered. Subsequently, Self-RAG processes different retrieved passages concurrently, evaluates their relevance, and generates corresponding task outputs. Finally, critique tokens are output, upon which it critiques its own output and chooses the best one in terms of quality and factuality. It may revise its answer and retrieve again, until it considers its answer satisfactory (cf. Figure 5).

The training can be summarised as follows. They trained the generator model M on a "curated corpus with interleaving passages retrieved by a retriever R and reflection tokens predicted by a critic model C." Starting point is a dataset of input–output pairs (from various sources, with varying factuality and difficulty levels). Manually annotating these with reflection tokens is too costly, so they first used GPT-4 to generate such annotations. Manual evaluation confirmed that GPT-4's outputs align well with human judgment. To avoid long-term reliance on proprietary GPT-4—due to API costs and reproducibility concerns—they train an in-house critic model C to imitate GPT-4's annotations. This model is then used, along with the retriever R, to produce training targets that include: retrieval markers, retrieved passages and reflection tokens. Finally, M is trained on this enriched data using the standard next-token prediction objective. In doing so, it learns to generate both the target output and the reflection tokens, enabling it to self-reflect at inference time. Note that during training, the

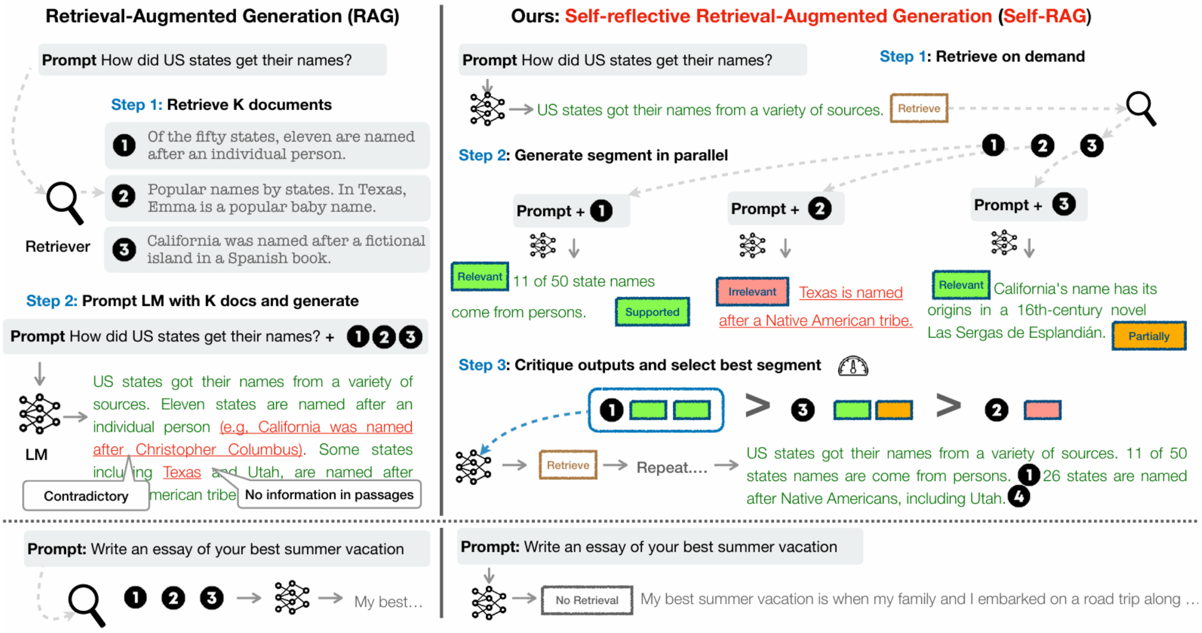
Retrieval-Augmented Generation (RAG)
Prompt How did US states get their names?
Step 1: Retrieve K documents
1 Of the fifty states, eleven are named after an individual person.
2 Popular names by states. In Texas, Emma is a popular baby name.
3 California was named after a fictional island in a Spanish book.
Retriever
Step 2: Prompt LM with K docs and generate
Prompt How did US states get their names? + 1 2 3
LM
US states got their names from a variety of sources. Eleven states are named after an individual person (e.g, California was named after Christopher Columbus). Some states including Texas and Utah, are named after Native American tribe.
Contradictory
No information in passages
Ours: Self-reflective Retrieval-Augmented Generation (Self-RAG)
Prompt How did US states get their names?
Step 1: Retrieve on demand
US states got their names from a variety of sources.
Retrieve
Step 2: Generate segment in parallel
Prompt + 1
Prompt + 2
Prompt + 3
Relevant
11 of 50 state names come from persons.
Supported
Irrelevant
Texas is named after a Native American tribe.
Relevant
California's name has its origins in a 16th-century novel Las Sergas de Esplandián.
Partially
Step 3: Critique outputs and select best segment
1 > 3 > 2
Retrieve
Repeat....
US states got their names from a variety of sources. 11 of 50 states names are come from persons. 1 26 states are named after Native Americans, including Utah. 4
Prompt: Write an essay of your best summer vacation
1 2 3
My best...
Prompt: Write an essay of your best summer vacation
No Retrieval
My best summer vacation is when my family and I embarked on a road trip along …

Figure 5: Overview of the Self-RAG pipeline. Sourced from Asai et al. (2023), Figure 1.

	Short-form		Closed-set		Long-form generations (with citations)					
	PopQA	TQA	Pub	ARC	Bio			ASQA		
LM	(acc)	(acc)	(acc)	(acc)	(FS)	(em)	(rg)	(mau)	(pre)	(rec)
LMs with proprietary data										
ChatGPT	29.3	**74.3**	70.1	**75.3**	71.8	35.3	36.2	68.8	–	–
Ret-ChatGPT	50.8	65.7	54.7	**75.3**	–	**40.7**	**39.9**	**79.7**	65.1	**76.6**
Perplexity.ai	–	–	–	–	71.2	–	–	–	–	–
Baselines with retrieval										
Llama2-FT$_{7B}$	48.7	57.3	64.3	65.8	78.2	31.0	35.8	51.2	5.0	7.5
SAIL*$_{7B}$	–	–	69.2	48.4	–	–	–	–	–	–
Llama2$_{13B}$	45.7	47.0	30.2	26.0	77.5	16.3	20.5	24.7	2.3	3.6
Alpaca$_{13B}$	46.1	66.9	51.1	57.6	77.7	**34.8**	36.7	56.6	2.0	3.8
Our SELF-RAG $_{7B}$	54.9	66.4	72.4	67.3	**81.2**	30.0	35.7	**74.3**	66.9	67.8
Our SELF-RAG $_{13B}$	**55.8**	**69.3**	**74.5**	**73.1**	80.2	31.7	**37.0**	71.6	**70.3**	**71.3**

Figure 6: Experiment results across six tasks. Bold indicates best performance among non-proprietary models, gray and bold indicates a proprietary model outperforming all other models. Adapted from Asai et al. (2023), Table 2.

retrieved text chunks are visible to the model but excluded from loss calculation—they serve as external context to inform generation and critique, but are not targets the model is expected to reproduce.

Self-RAG exhibits the best scores in all but one category amongst non-proprietary models, vanilla and retrieval augmented alike, outperforming many of them by a large margin (cf. Figure 6). The strongest proprietary model, ChatGPT, beats it in slightly more than half of categories, yet Self-RAG is competitive. It is to be noted that no adaptive RAG methods are included in the evaluation.

They do, however, illustrate the effectiveness of their approach, as Llama2-FT7B, the baseline language model trained on the same instruction–output pairs as Self-RAG, but without retrieval or self-reflection tokens, performs worse than Self-RAG, despite being retrieval-enhanced at test-time.

3.3 Uncertainty-/Consistency-based Adaptive Systems

This section examines four related approaches that use the model's confidence in its answer to determine the retrieval strategy. One advantage of such an approach is that no additional model or classifier training is needed. Jiang et al. (2023) looked at uncertainty on a token-level, Su et al. (2024) extended that by incorporating attention, Yao et al. (2024) went in a different direction by looking at *internal* uncertainty, measured as consistency of internal states across different generations. This idea of uncertainty as consistency is also implemented by Ding et al. (2024). They look at paraphases of the same query, different languages and models in order to determine the consistency/uncertainty and thus the retrieval strategy.

Jiang et al. (2023) proposed a method called **F**orward-**L**ooking **A**ctive **RE**trieval augmented generation (***FLARE***), where the model actively decides when and what to retrieve at any point during generation, similarly to how a human would write an article. The core idea is to predict the next sentence and, if the sentence contains low-confidence tokens, use these to formulate a query to retrieve relevant documents and regenerate the sentence (cf. Figure 7). They propose two different strategies of using sentences with low-confidence tokens to generate retrieval queries:

1. Implicit query by masking: *Joe Biden attended _, where he earned _.*

2. Explicit query by LM-based question generation: *What university did Joe Biden attend? What degree did Joe Biden earn?*

Jiang et al. (2023) report "superior or competitive performance on all tasks", with the tasks being four long-form knowledge-intensive generation tasks. The tasks were: multihop QA, which requires reasoning across multiple Wikipedia articles; commonsense reasoning (StrategyQA), involving commonsense yes/no questions such as 'Would a pear sink in water?'; long-form QA (ASQA) with ambiguous questions, whereby they used two settings, one as-is and one with a hint regarding the ambiguities; and open-domain summarisation (WikiAsp), focused on generating summaries of Wikipedia entities.

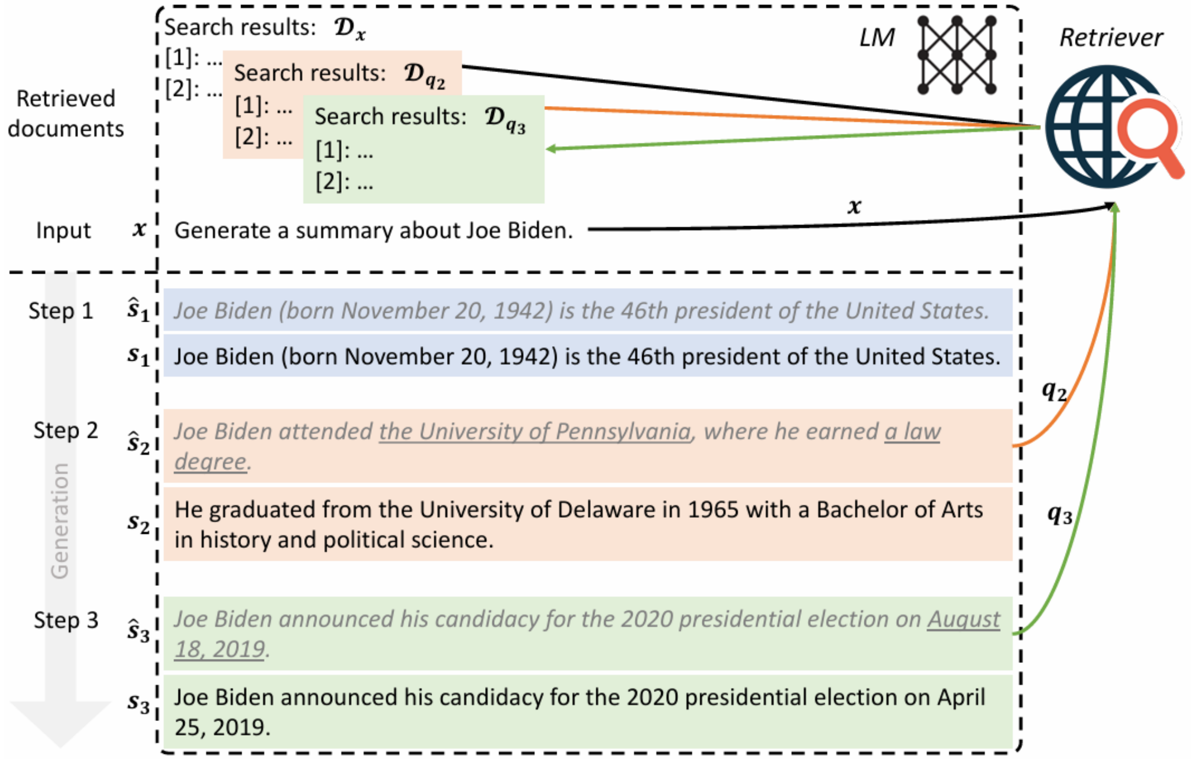
Search results: $\mathcal{D}_x$
[1]: ...
[2]: ...
Search results: $\mathcal{D}_{q_2}$
[1]: ...
[2]: ...
Search results: $\mathcal{D}_{q_3}$
[1]: ...
[2]: ...
LM
Retriever
Retrieved documents
Input
x
Generate a summary about Joe Biden.
x
Step 1
$\hat{s}_1$
Joe Biden (born November 20, 1942) is the 46th president of the United States.
s_1
Joe Biden (born November 20, 1942) is the 46th president of the United States.
q_2
Step 2
$\hat{s}_2$
Joe Biden attended the University of Pennsylvania, where he earned a law degree.
s_2
He graduated from the University of Delaware in 1965 with a Bachelor of Arts in history and political science.
q_3
Generation
Step 3
$\hat{s}_3$
Joe Biden announced his candidacy for the 2020 presidential election on August 18, 2019.
s_3
Joe Biden announced his candidacy for the 2020 presidential election on April 25, 2019.

Figure 7: An Illustration of the FLARE approach. Given the user input x and the initial set of retrieved documents D_x, FLARE repeatedly predicts the next sentence (shown in gray italics). It then checks if the sentence contains any low-confidence words (underlined). If it does (as shown in steps 2 and 3), the system retrieves additional relevant documents and uses them to revise and regenerate the sentence. Sourced from Jiang et al. (2023), Figure 1.

FLARE outperformed all tested baselines—both single-time and multi-time retrieval methods—across all tasks. In multihop QA, the method sees the largest gains (cf. Figure 8). In the ablation studies, they showed that using the next sentence for retrieval works better than using the previous one, that selective retrieval triggered only for low-confidence tokens outperforms always retrieving, and that both masking and explicit retrieval query formulation methods are effective, but masking slightly improves precision.

Su et al. (2024)'s ***DRAGIN*** is heavily inspired by FLARE and proposed a refined approach using the transformer-inherent self-attention. They aim to not only evaluate the uncertainty of each token, but also its semantics and importance in the sentence context. Their two key components are: RIND (Real-time Information Needs Detection) and QFS (Query Formulation based on Self-Attention).

Given a predicted sentence, RIND quantifies the uncertainty of each token by calculating the entropy of the token's probability distribution across the

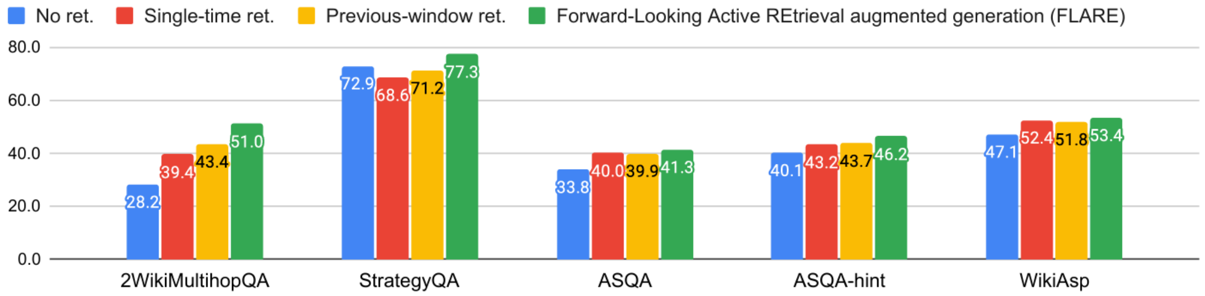

Figure 8: Comparison of FLARE with baseline methods across all tasks and datasets, using the primary evaluation metric for each: exact match (EM) for 2WikiMultihopQA, StrategyQA and ASQA, and UniEval for WikiAsp. Sourced from Jiang et al. (2023), Figure 4.

vocabulary (like FLARE): high confidence means low entropy (one token has most probability mass), and vice-versa. Additionally, it multiplies it with the maximum self-attention weight and a binary stopword indicator to compute the final uncertainty score (cf. Figure 9). Stopwords will get thus get an uncertainty indicator of 0, and uncertain *and* semantically important words get high scores.

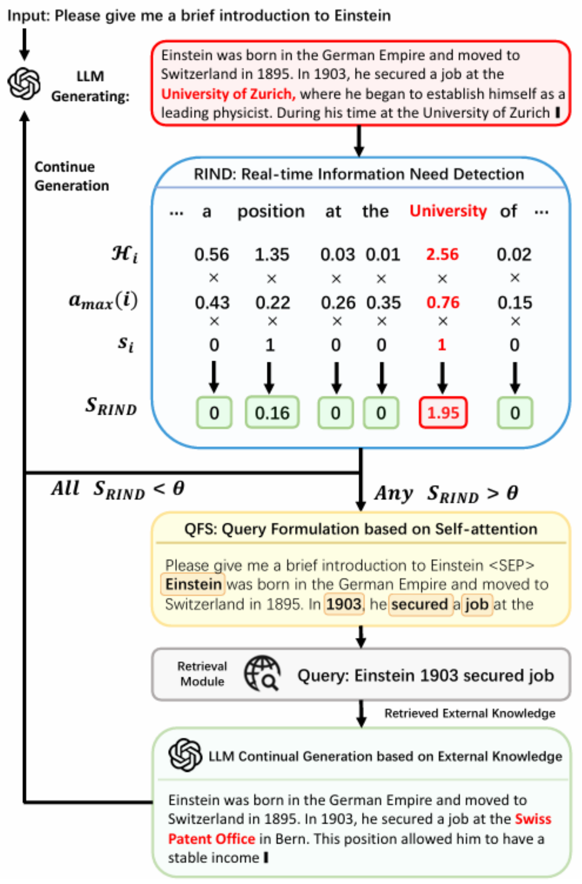
Input: Please give me a brief introduction to Einstein
LLM Generating:
Einstein was born in the German Empire and moved to Switzerland in 1895. In 1903, he secured a job at the University of Zurich, where he began to establish himself as a leading physicist. During his time at the University of Zurich
Continue Generation
RIND: Real-time Information Need Detection
... a position at the University of ...
$\mathcal{H}_i$ 0.56 1.35 0.03 0.01 2.56 0.02
$a_{max}(i)$ 0.43 0.22 0.26 0.35 0.76 0.15
s_i 0 1 0 0 1 0
S_{RIND} 0 0.16 0 0 1.95 0
All $S_{RIND} < \theta$
Any $S_{RIND} > \theta$
QFS: Query Formulation based on Self-attention
Please give me a brief introduction to Einstein <SEP> Einstein was born in the German Empire and moved to Switzerland in 1895. In 1903, he secured a job at the
Retrieval Module
Query: Einstein 1903 secured job
Retrieved External Knowledge
LLM Continual Generation based on External Knowledge
Einstein was born in the German Empire and moved to Switzerland in 1895. In 1903, he secured a job at the Swiss Patent Office in Bern. This position allowed him to have a stable income

Figure 9: The DRAGIN framework. Sourced from Su et al. (2024), Figure 1.

Once the position i that needs retrieval is determined, QFS evaluates all the preceding attention weights with regards to token i and selects the top n tokens to construct the query (cf. Figure 9).

For evaluation, two of the four datasets overlap with FLARE, namely 2WikiMultihopQA and StrategyQA. Additionally, the use HotpotQA, which is similar to 2WikiMultihopQA, and IIRC, which involves answering questions based on a core passage and selectively retrieving additional passages for clarification. As DRAGIN, just like FLARE, can be applied to any LLM, they use different LLMs and different settings for their experiments: **wo-** stands for **without**, **SR-** stands for **single-round**, **FL-** for **fixed length**, **FS-** for **fixed sentence**, whereby fixed length means retrieval after every n tokens and fixed sentence after every sentence.

DRAGIN consistently outperforms FLARE in every single setting, and is only narrowly (< 0.02) beaten in three out of twelve (three models times four benchmarks) settings. Figure 10 illustrates these results by showing the results with LLaMa2-13b-chat, one of the three tested models.

Yao et al. (2024)'s ***SEAKR***, instead of relying on these surface-level probabilities, used an internal measure: consistency of internal states across generations, which they called **self-aware uncertainty**. More precisely, *internal state* refers to the hidden representation of the $< EOS >$-token, as it compresses the information of both input and output. Additionally, SEAKR is augmented with adaptive integration strategies of retrieved knowledge, termed **self-aware re-ranking** and **self-aware reasoning.** Self-aware re-ranking consists in the following: for each of the n retrieved snippets, k rationales are generated, and the snippet for which the self-aware uncertainty, the variation across generation of rationales, was the lowest is chosen. Self-aware reasoning makes up the final step: once the retrieval and rationale generation has stopped, there are two ways to generate the final answer:

1. k answer samples are generated based on all rationales and the uncertainty is measured.

		2WikiMultihopQA		HotpotQA		StrategyQA	IIRC	
LLM	**RAG Method**	**EM**	**F1**	**EM**	**F1**	**Accuracy**	**EM**	**F1**
Llama2-13b-chat	**wo-RAG**	0.187	0.2721	0.223	0.3097	0.650	0.168	0.2039
	SR-RAG	0.245	0.3364	0.263	0.3706	0.654	**0.196**	**0.2303**
	FL-RAG	0.217	0.3054	0.177	0.2682	0.648	0.155	0.1875
	FS-RAG	0.270	0.3610	0.267	0.3715	0.655	0.171	0.2061
	FLARE	0.224	0.3076	0.180	0.2756	0.655	0.138	0.1667
	DRAGIN (Ours)	**0.304**	**0.3931**	**0.314**	**0.4238**	**0.689**	0.185	0.2221

Figure 10: The results of DRAGIN and other baselines applied to LLaMa2-13b-chat on fours benchmarks, with the best results in bold. Adapted from Su et al. (2024), Table 2.

2. k answer samples are generated based on all retrieved knowledge snippets, using CoT-reasoning, and the uncertainty is measured.

The approach with the more consistent answers is chosen and a fresh, final answer gets generated and output.

The process, illustrated in Figure 11, is as follows: at each iterative step, k rationales are generated based on the context and evaluated in terms of uncertainty. If there is high uncertainty, n knowledge snippets are retrieved and the most useful snippet chosen using self-aware re-ranking as described above. It is stored separately and based on it, the next rationale is generated and added to the context, whereby the next iteration begins.

We break out of the loop once a maximum step-limit is reached or an answer (recognisable by a specific given format) is output. Then, self-aware reasoning is applied to generate the best final answer.

They measure performance on complex QA (2WikiMultiHopQA, HotpotQA, and IIRC) as well as simple QA datasets (NaturalQuestions, TriviaQA, and SQuAD). They evaluate against some of the strongest adaptive-RAG models like Self-RAG, FLARE and DRAGIN, and also strong baselines of different a different kind, such as CoT and retrieval-augmented CoT (IRCOT), which retrieves for every reasoning step by default.

SEAKR outperforms the best baselines by 6%, 5.5% and 0.6% on the three complex datasets, indicating that their strategy is well-suited for solving complex questions. Its adaptive components prove beneficial, as SEAKR follows a CoT-style reasoning, yet both plain CoT and always-retrieve-CoT underperform it by a large margin (cf. Figure 12).

Models	2Wiki		HPQA		IIRC	
	EM	F1	EM	F1	EM	F1
CoT	14.6	22.3	18.4	27.5	13.9	17.3
IR-CoT	18.9	26.5	21.4	30.4	17.8	21.6
Self-RAG	4.6	19.6	6.8	17.5	0.9	5.7
FLARE	14.3	21.3	14.9	22.1	13.6	16.4
DRAGIN	22.4	30.0	23.7	34.2	19.1	22.9
SEAKR	**30.2**	**36.0**	**27.9**	**39.7**	**19.5**	**23.5**

Figure 12: SEAKR's results on complex datasets in percent, with the best results in **bold**. Sourced from Yao et al. (2024), Table 1.

Note that while Self-RAG's results are less satisfactory, they state that this is "mainly caused by the distribution of its fine-tuning data, which is generated by GPT-4 [...] with demonstrations from NaturalQuestions, a simple QA dataset". This shows Self-RAGs dependence on training data and lack of generaliseability as opposed to tuning-free adaptive RAG methods such as SEAKR.

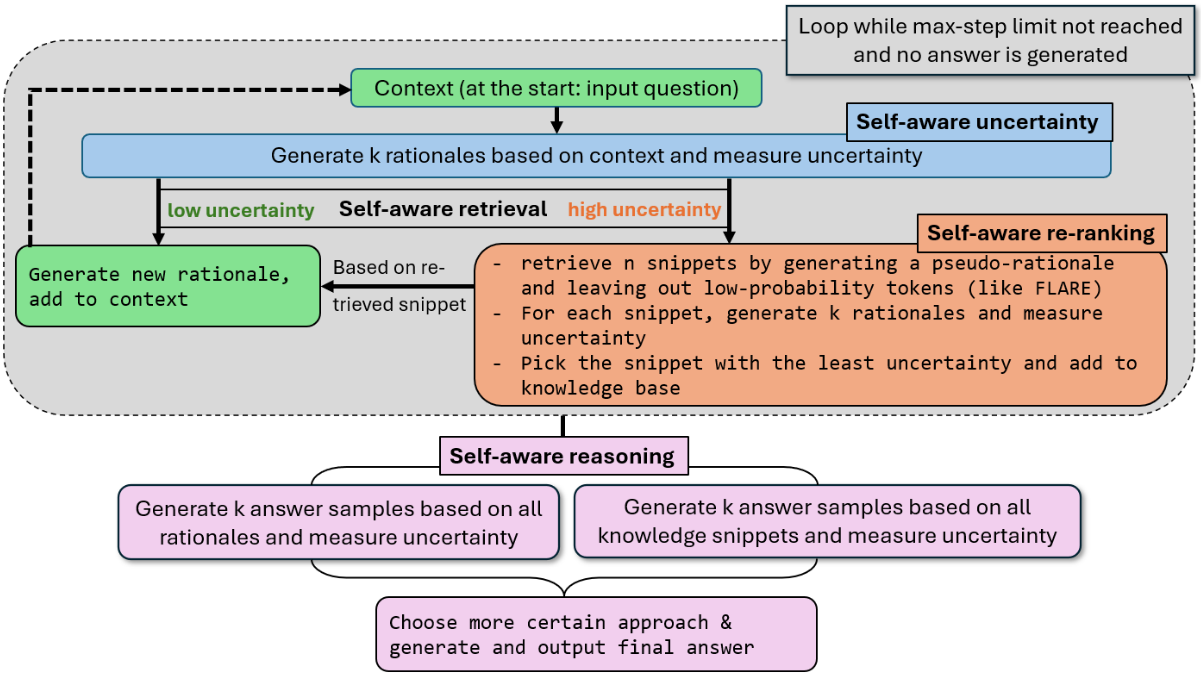
Loop while max-step limit not reached and no answer is generated
Context (at the start: input question)
Self-aware uncertainty
Generate k rationales based on context and measure uncertainty
low uncertainty
Self-aware retrieval
high uncertainty
Self-aware re-ranking
Generate new rationale, add to context
Based on re-trieved snippet
- retrieve n snippets by generating a pseudo-rationale and leaving out low-probability tokens (like FLARE)
- For each snippet, generate k rationales and measure uncertainty
- Pick the snippet with the least uncertainty and add to knowledge base
Self-aware reasoning
Generate k answer samples based on all rationales and measure uncertainty
Generate k answer samples based on all knowledge snippets and measure uncertainty
Choose more certain approach & generate and output final answer

Figure 11: An Illustration of how SEAKR works. Created by Dominic P. Fischer based on Yao et al. (2024).

On the simple QA datasets, SEAKR achieves the best performance on TriviaQA and SQuAD. On NaturalQuestions, SEAKR is comparable to FLARE and better than DRAGIN, yet lagging behind Self-RAG, which, as stated above, was fine-tuned on NaturalQuestions-style data. They note that for simple QA, SEAKR advantages are less clear-cut than for complex ones (cf. Figure 13).

Model	NQ		TriviaQA		SQuAD	
	EM	F1	EM	F1	EM	F1
CoT	13.4	18.7	42.6	48.6	8.7	13.6
Self-RAG	**32.3**	**40.2**	21.2	37.9	5.1	18.3
FLARE	25.3	35.9	51.5	60.3	19.4	28.3
DRAGIN	23.2	33.2	54.0	62.3	18.7	28.7
SEAKR	25.6	35.5	**54.4**	**63.1**	**27.1**	**36.5**

Figure 13: SEAKR's results on simple datasets in percent, with the best results in **bold**. Sourced from Yao et al. (2024), Table 2.

The ablation studies confirm that each of their self-aware components improves performance. Below are detailed the averages of EM and F1 across three datasets. Self-aware uncertainty estimation, replaced by a prompt-based approach in the ablation study (*"do I [as a model] have sufficient knowledge to solve the question?"*), improves performance by 2.28% , followed by re-ranking with 1.6%, and reasoning and retrieval at around 1%.

Finally, Ding et al. (2024)'s approach ***Rowen*** (**R**etrieve **o**nly **when** needed) uses different kinds of consistency as measure of uncertainty. First, they employ CoT-reasoning to generate an initial response. They then measure consistency for the same query, (i), across different paraphrases, (ii), across different languages, and (iii), across different models. The results are integrated into a final consistency score, depending on which retrieval may or may not be invoked. If so, the final answer is generated by prompting the model with the initial query, the CoT, the initial answer and the retrieved documents. If not, the initial answer is the final answer. This approach outperforms strong adaptive retrieval baselines (cf. Figure 14).

4 Comparison and Timeline

Mallen et al. (2022)'s Adaptive Retrieval was one of the earliest and probably *the* earliest highly impactful paper on the topic of adaptive RAG. Besides introducing their own dataset **PopQA** based on knowledge triplets (cf. Section 3.1), their core contribution lies in empirically motivating adaptive RAG approaches and demonstrating their usefulness with their own very simple approach, which is based on the query entities' popularity/frequency, as well as their relationship. In a subsequent study, Zhang et al. (2024) confirm that simple adaptive approaches, in their case prompt-based, yield positive results, while proposing **RetrievalQA**, a dataset of new-world and long-tail knowledge.

Models	**TruthfulQA**			**StrategyQA**
	GPT-Judge ↑	**BLEU ↑**	**Rouge-L ↑**	**Accuracy ↑**
Vanilla LLMs				
ChatGPT (`gpt-3.5-turbo`)	47.92	10.17	**31.31**	61.40
Adaptive Retrieval Methods				
FLARE (Jiang et al., 2023)	45.04	11.59	26.83	61.19
Adaptive-Retrieval (Mallen et al., 2023)	45.55	8.87	26.75	62.50
Self-RAG (Asai et al., 2023)	40.36	4.36	21.28	58.40
Adaptive-RAG (Jeong et al., 2024)	46.02	10.29	26.24	68.50
LUQ (Zhang et al., 2024)	55.08	5.79	21.44	71.00
Our Framework				
`Rowen-CL`	57.39	7.60	24.16	74.00
`Rowen-CM`	56.29	6.85	22.36	72.40
`Rowen-Hybrid`	**59.34**	**15.27**	31.15	**75.60**

Figure 14: Results on TruthfulQA and StrategyQA datasets. Rowen-Hybrid achieves some of the best results following careful hyperparameter tuning (such as retrieval threshold and weight of the different consistency measures). Adapted from Ding et al. (2024), Table 1.

In the meantime, a major breakthrough had occurred with Jiang et al. (2023)'s FLARE. Their novel approach is a full-fledged framework of forward-looking, active RAG, leveraging the token (im-)probabilities to find tokens in need of RAG.

Subsequently, a number of papers build on FLARE, notably Su et al. (2024)'s DRAGIN and Yao et al. (2024)'s SEAKR. DRAGIN compares itself to standard RAG baselines as well as FLARE, noting improvements in every single setting. SEAKR, having now strong adaptive baselines at its disposal, compares itself to FLARE, DRAGIN, and Asai et al. (2023)'s Self-RAG, which has since been devised. SEAKR confirms that DRAGIN outperforms FLARE and notes its own improvements over both, while Self-RAG is shown to perform well exclusively on data similar to its fine-tuning data.

This underlines the high potential of the end-to-end approach of fine-tuning models to learn to reflect on their own outputs—as advertised in the original paper (Asai et al., 2023) and reflected by its popularity—yet also illustrates the dependence on fine-tuning or implementation, as highlighted in Section 3.2. A weakness of Jeong et al. (2024)'s Adaptive RAG—an approach which uses a ternary learned classifier to classify queries as without retrieval, or with single or multi retrieval—is the lack of comparison to other strong adaptive RAG systems, especially ones with a different approach, such as FLARE, DRAGIN, SEAKR or Rowen. It is to be noted that due to the timeline, some of these comparisons may not have been possible.

Finally, Ding et al. (2024)'s Rowen has a wide range of adaptive RAG systems to compare against. They inspire themselves by consistency-based retrieval invocation, yet their consistency detection module spans from different paraphrases to different languages to different models. In their results, many of the above mentioned adaptive RAG approaches work well and are in the same range (FLARE, Adaptive-Retrieval, Self-RAG, Adaptive-RAG), while Rowen manages to surpass them considerably. DRAGIN and especially SEAKR, having been shown to surpass other approaches, are not mentioned in the evaluation. Ding et al. (2024) with Rowen and Yao et al. (2024) with SEAKR both acknowledge in their limitations section that the number of different generations poses a problem. While they address the computational side resp. the inference speed, they do not consider the resource consumption.

5 Challenges and Open Questions

Open questions, then, are how to balance factual accuracy and resource use. This paper has illustrated that uncertainty-driven adaptive RAG, checking consistency across different variants or perturbations in different steps of the retrieval and generation process, is a very strong approach in the (adaptive) RAG landscape. Yet evidently, such amounts of generation come at a cost. What weight should one assign to these different aspects of an approach? It seems that there is no clear answer, and the tradeoff largely depends on the scenario an approach is applied to.

Another point of discussion are metrics and comparability. If we followed only the numbers, approaches like DRAGIN (Su et al., 2024), SEAKR (Yao et al., 2024) or Rowen (Ding et al., 2024) should enjoy much more popularity than they do, as they outperform e.g. the famous FLARE (Jiang et al., 2023) or Self-RAG (Asai et al., 2023). The crux is that to see whether the numbers actually mean what they claim to imply, we have to have a very close look at the papers, even their code. Furthermore, might the prestige of certain researchers or lack thereof skew how a paper is perceived?

What is duly needed are standardised, consistent and telling benchmarks and evaluation metrics, as well as transparent implementations.

6 Conclusion

In conclusion, this paper shows that adaptive RAG is highly beneficial, with adaptive approaches generally outperforming both non-RAG and traditional always-retrieving RAG systems. Consistency- resp. perturbation-driven approaches exhibit consistently strong performance, with no fine-tuning required. However, they might consume more resources, undermining another strength of adaptive RAG, which is its potential to undercut traditional RAG's resource consumption.

Implementation, benchmark dataset and evaluation differences can make comparisons between approaches difficult, as well as technical-philosophical questions of tradeoffs between time plus resource consumption and accuracy of generated output. There may not be a universally optimal adaptive RAG system—instead, we see a landscape of viable options, each to be chosen adaptively according to situational requirements.

References

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

Hanxing Ding, Liang Pang, Zihao Wei, Huawei Shen, and Xueqi Cheng. 2024. Retrieve only when it needs: Adaptive retrieval augmentation for hallucination mitigation in large language models. *arXiv preprint arXiv:2402.10612*.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403*.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781.

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

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. Dragin: Dynamic retrieval augmented generation based on the information needs of large language models. *arXiv preprint arXiv:2403.10081*.

Zijun Yao, Weijian Qi, Liangming Pan, Shulin Cao, Linmei Hu, Weichuan Liu, Lei Hou, and Juanzi Li. 2024. Seakr: Self-aware knowledge retrieval for adaptive retrieval augmented generation. *arXiv preprint arXiv:2406.19215*.

Zihan Zhang, Meng Fang, and Ling Chen. 2024. Retrievalqa: Assessing adaptive retrieval-augmented generation for short-form open-domain question answering. *arXiv preprint arXiv:2402.16457*.