Unified Active Retrieval for Retrieval Augmented Generation

Qinyuan Cheng[1,2*] **Xiaonan Li**[1*] **Shimin Li**[1] **Qin Zhu**[1] **Zhangyue Yin**[1]
Yunfan Shao[1,2] **Linyang Li**[2] **Tianxiang Sun**[1] **Hang Yan**[2] **Xipeng Qiu**[1,3,†]

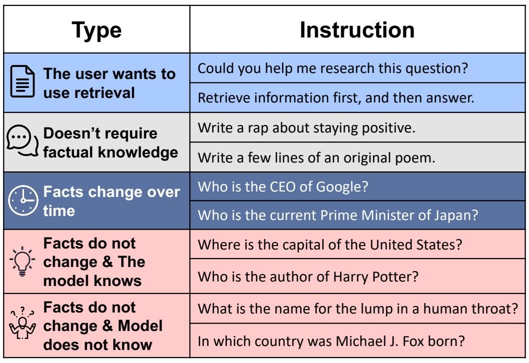
[1]Fudan University
[2]Shanghai AI Laboratory
[3]Shanghai Collaborative Innovation Center of Intelligent Visual Computing
chengqy21@m.fudan.edu.cn {lixn20, xpqiu}@fudan.edu.cn

Abstract

In Retrieval-Augmented Generation (RAG), retrieval is not always helpful and applying it to every instruction is sub-optimal. Therefore, determining whether to retrieve is crucial for RAG, which is usually referred to as Active Retrieval. However, existing active retrieval methods face two challenges: 1. They usually rely on a single criterion, which struggles with handling various types of instructions. 2. They depend on specialized and highly differentiated procedures, and thus combining them makes the RAG system more complicated and leads to higher response latency. To address these challenges, we propose **U**nified **A**ctive **R**etrieval (**UAR**). UAR contains four orthogonal criteria and casts them into plug-and-play classification tasks, which achieves multifaceted retrieval timing judgements with negligible extra inference cost. We further introduce the Unified Active Retrieval Criteria (UAR-Criteria), designed to process diverse active retrieval scenarios through a standardized procedure. Experiments on four representative types of user instructions show that UAR significantly outperforms existing work on the retrieval timing judgement and the performance of downstream tasks, which shows the effectiveness of UAR and its helpfulness to downstream tasks.

1 Introduction

With the rapid development of large language models (LLMs) (Brown et al., 2020; Touvron et al., 2023; Zeng et al., 2023; Yang et al., 2023; Cai et al., 2024; Bai et al., 2023), AI assistants based on LLMs become unbiquitous and show remarkable abilities on various types of instructions, e.g., coding, writing and reasoning (OpenAI, 2022; Taori et al., 2023; Chiang et al., 2023; Sun et al., 2024; OpenAI, 2023; Anthropic, 2023; Anil et al., 2023). However, LLMs often generate fabricated and non-factual information (Lin et al., 2022b; Cheng et al., 2023; Wang et al., 2023a), which is called "hallucination" and makes LLMs' responses not trustworthy in real-world scenarios.



Type	Instruction
The user wants to use retrieval	Could you help me research this question?
	Retrieve information first, and then answer.
Doesn't require factual knowledge	Write a rap about staying positive.
	Write a few lines of an original poem.
Facts change over time	Who is the CEO of Google?
	Who is the current Prime Minister of Japan?
Facts do not change & The model knows	Where is the capital of the United States?
	Who is the author of Harry Potter?
Facts do not change & Model does not know	What is the name for the lump in a human throat?
	In which country was Michael J. Fox born?

Figure 1: Different types of user instructions, which can not be handled by single active retrieval criteria.

Retrieval-Augmented Generation (RAG) is a prevailing approach to address LLM's hallucination (Guu et al., 2020; Gao et al., 2024). Given a user query, it usually first retrieves relevant documents and then uses them to augment the LLM's factual correctness. However, retrieval is not always helpful and applying it to every instruction is sub-optimal. When faced with instructions that do not require external knowledge, RAG can impair the creativity and versatility of LLMs (Asai et al., 2023).

If irrelevant knowledge is retrieved, it will hinder the LLM from utilizing its internal knowledge effectively and make it produce low-quality responses (Shi et al., 2023; Yoran et al., 2023). Meanwhile, compared with only LLM, RAG involves an additional retrieval process and the longer LLM input, resulting in significantly longer response latency. Therefore, applying RAG for all instructions is sub-optimal and unnecessary, and determining the correct timing for retrieval is crucial for LLMs' real-world application, which is often reftered to as *Active Retrieval* (Jiang et al., 2023; Asai et al., 2023).

In general, there are two lines of active retrieval

[*]Equal contribution
[†]Corresponding author.

Findings of the Association for Computational Linguistics: EACL 2024, pages 17153–17166
November 12-16, 2024 ©2024 Association for Computational Linguistics

	UAR (our work)	FLARE (Jiang et al., 2023)	Self-RAG (Asai et al., 2023)	SKR (Wang et al., 2023b)
Intent Awareness?	✓	✗	✗	✗
Knowledge Awareness?	✓	✗	✓	✗
Time Awareness?	✓	✗	✗	✗
Self Awareness?	✓	✓	✗	✓

Table 1: Comparison of UAR to other active retrieval methods. Exciting methods only consider a single active retrieval criterion, while UAR unifies four orthogonal criteria and can handle various types of user instructions.

methods. One is the "knowledge-aware" method, based on the instruction's factual relevance, e.g., Self-RAG (Asai et al., 2023). If the instruction requires factual information, the retrieval will be triggered. Another line of work is the "self-aware" method, based on the LLM's self awareness (Wang et al., 2023b). The retrieval is only triggered when the LLM thinks that it does not know the answer, i.e., when it is uncertain. In this way, the retrieval can supplement knowledge for the LLM when necessary and avoid unnecessary retrieval cost. Although these methods can determine retrieval timing for specialized scenarios, they still face two challenges: 1. Previous work usually relies on a single criterion, which struggles with diverse scenarios. For instance, the self-aware method (Wang et al., 2023b; Liu et al., 2024; Ding et al., 2024) struggles with various instructions such as time-sensitive queries or those with user's explicit retrieval intent. For time-sensitive questions, it is challenging for a static LLM to judge whether it possesses the correct knowledge for a rapidly changing answer. Additionally, these methods often overlook user's intent in real-world scenarios, such as when a user seeks a verifiable answer that requires external information sources, necessitating retrieval. Therefore, correctly determining whether to retrieve requires multifaceted decision-making. 2. Existing methods rely on specialized procedures, complicating the integration within the RAG system and increasing computational load. For example, FLARE (Jiang et al., 2023) uses the confidence of generation and Rowen (Ding et al., 2024) relies on response divergence for the same question. These highly differentiated approaches are difficult to unify, making it very difficult to extend them to new scenarios.

To address these challenges, we propose **U**nified **A**ctive **R**etrieval (**UAR**), a unified and comprehensive framework for judging whether to retrieve for various types of user instructions. UAR consists of various orthogonal criteria of retrieval timing and casts them into unified classification tasks, and thus can judge the LLM's retrieval timing both comprehensively and efficiently. Specifically, UAR consists of four orthogonal criteria for determining the retrieval timing: 1) **Intent-aware**: whether the user desires retrieval / external information; 2) **Knowledge-aware**: whether the question requires fact knowledge; 3) **Time-Sensitive-aware**: whether the question is time-sensitive; 4) **Self-aware**: whether the LLM has the internal knowledge. As shown in Table 1, compared with previous methods of single criterion (Jiang et al., 2023; Wang et al., 2023b; Asai et al., 2023), UAR can comprehensively handle various types of user instructions and call retrieval accurately considering multiple active retrieval criteria. To efficiently achieve judgements of multiple criteria, UAR unifies each criterion's judgement into binary classification tasks using lightweight classifiers. For each criterion c_i, we train a plug-and-play binary classifier on the last layer's hidden states of a fixed LLM, to judge whether the input requires retrieval according to c_i. In this way, UAR does not change LLMs' parameters, avoiding the costly LLM fine-tuning and performance degradation (Yang et al., 2024). Meanwhile, the classifiers and LLM generation share the same input encoding, which makes UAR only need to encode the input once and thus achieves multifaceted retrieval timing judgements with negligible extra inference cost.

To handle various instructions in an unified procedure, we further propose Unified Active Retrieval Criteria (UAR-Criteria), which specifies priorities for multiple retrieval criteria and unifies them into a single multifaceted decision tree. As shown in Figure 2, UAR-Criteria can trigger retrieval for time-sensitive or LLM-unknown instructions, which facilitates necessary external information supplement. Meanwhile, UAR-Criteria cancels retrieval for those non-knowledge-intensive or LLM-known instructions, which avoids the negative effect of unnecessary retrieval. In this way, UAR-Criteria unifies the process to comprehensively decide whether to retrieval for various types of user instructions, which facilitates more effective RAG.

Experiments on four representative types of user instructions show that UAR significantly outperforms existing work on the retrieval timing judgement accuracy and the performance of downstream tasks, which verifies the effectiveness of UAR and its helpfulness to downstream tasks. We summarize our contributions as follows:

- We propose an active retrieval framework named Unified Active Retrieval (UAR) for Retrieval-Augmented Generation (RAG). To the best of our knowledge, UAR is the first work to propose multifaceted criteria for active retrieval and demonstrate its necessity.

- We curate the Active Retrieval benchmark (AR-Bench) for evaluating the accuracy of retrieval timing and conduct comprehensive experiments on AR-Bench and downstream tasks. The results show that UAR significantly outperforms existing work and achieves more efficient RAG.

- We release the code, data, models and relevant resources to facilitate future research[1].

2 Related Work

2.1 Active Retrieval

Compared to applying retrieval for every instruction (passive retrieval), active retrieval has advantages such as not hurting the versatility of the model, reducing the number of retrievals, and preventing interference from low-quality retrieval results. Self-RAG (Asai et al., 2023) construct active retrieval data using GPT-4 and teach the model to not retrieve when encounter non-knowledge-intensive instructions. FLARE (Jiang et al., 2023) proposes forward-looking active retrieval augmented generation based on model's confidence, only retrieving information when the model's uncertainty for the prediction is high. SKR (Wang et al., 2023b), RA-ISF (Liu et al., 2024) and Self-DC (Wang et al., 2024) first determines whether the model knows the questions and then retrieves only when the model does not know. However, current active retrieval methods mostly consider only a single scenario and are unable to adapt to complex situations in real-world applications.

2.2 Time-awareness of LLMs

There are some papers focus on the time awareness of large language models. Chen et al. (2021) construct a time-sensitive QA dataset called TimeQA to evaluate the model's ability to handle temporal questions. Fierro et al. (2024) create a benchmark named MULAN for evaluating the ability of language models to predict mutable facts. They find representations classification can distinct immutable and mutable facts, which means language models have a certain degree of temporal awareness. Zhao et al. (2024) investigate whether language models can align their internal knowledge to a target year. They construct a dataset which contains time-sensitive questions.

2.3 Self-awareness of LLMs

Self-awareness means that large language model can be aware of what they know and what they don't know. Kadavath et al. (2022) find that language models can be well-calibrated when using a multiple-choice template. And they also finetune a value head to predict whether language models know the answer to the given question. Lin et al. (2022a) finetune GPT-3 to express uncertainty in words on math questions. Yin et al. (2023) collect some unanswerable questions to evaluate whether language models can express uncertainty to these unanswerable questions. Zhang et al. (2023) utilize supervised fine-tune to teach large language models to refuse questions which beyond their knowledge scope. Cheng et al. (2024) explore more alignment methods beyond supervised fine-tuning to teach language models know and express what they don't know, like preference optimization. Results of previous work show that we can enhance language models' self-awareness with corresponding dataset.

3 Methodology

UAR is a plug-and-play active retrieval framework. As shown in Figure 2, we fix the parameters of the LLM and train a lightweight classifier for each active retrieval criteria using the model's hidden states, which is far more efficient than fine-tuning the entire model. Besides, UAR determines the need for active retrieval following the UAR-Criteria shown on the right side of Figure 2, invoking retrieval when necessary and avoiding unnecessary across various scenarios, making RAG more effective and efficient. For instructions requiring retrieval, we append the retrieved documents to the

[1] https://github.com/xiami2019/UAR

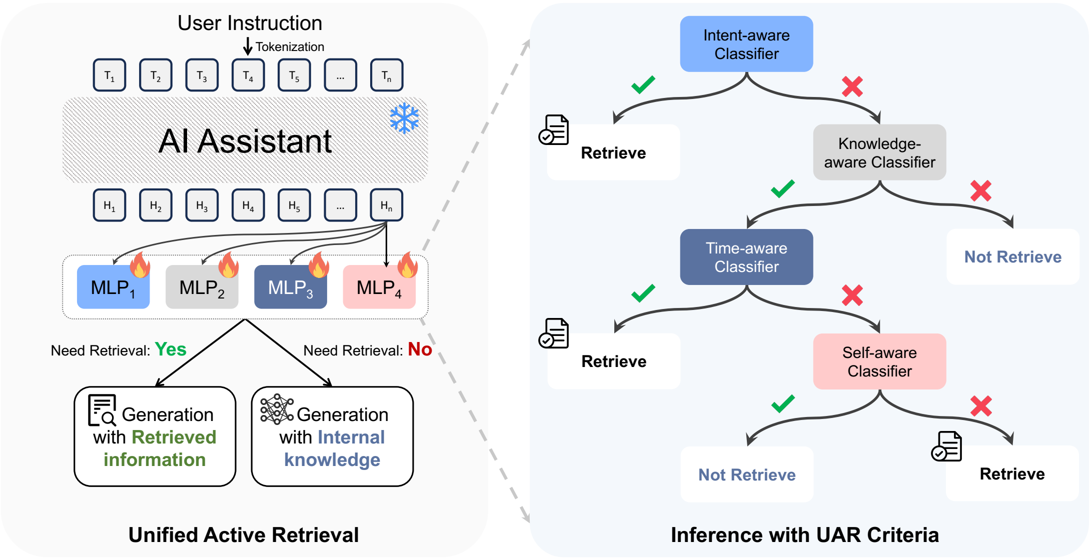

User Instruction
Tokenization
T_1 T_2 T_3 T_4 T_5 ... T_n
AI Assistant
H_1 H_2 H_3 H_4 H_5 ... H_n
MLP_1 MLP_2 MLP_3 MLP_4
Need Retrieval: Yes
Need Retrieval: No
Generation with Retrieved information
Generation with Internal knowledge
Unified Active Retrieval
Intent-aware Classifier
Retrieve
Knowledge-aware Classifier
Time-aware Classifier
Not Retrieve
Retrieve
Self-aware Classifier
Not Retrieve
Retrieve
Inference with UAR Criteria

Figure 2: Overview of the UAR framework. ❄ indicates that we freeze these parameters. 🔥 indicates that we update these parameters. Each MLP is a fully connected layer, with an input dimension equal to the model's hidden state dimension and an output dimension of 2.

original instruction, which means that UAR does not introduce extra LLM inference cost. We introduce the details of our UAR framework in the following sections.

3.1 UAR Classifiers Training

We construct distinct training data tailored to each scenario.

Self-aware In the self-aware scenario, the model must determine if it knows the answer to a given question. Following the methodology in Cheng et al. (2024), we create model-specific IDK (I don't know) datasets. For example, with the Llama2-7B-chat model, we use the TriviaQA (Joshi et al., 2017) dataset, sampling ten responses for each question. If all responses are correct, the question is marked as known; otherwise, it is unknown. 10% of the TriviaQA training set is used for validation, with the rest designated as the training set.

Time-aware In the time-aware scenario, it is critical to determine if a user's question is time-sensitive, meaning the answer changes over time. We utilize questions from TAQA's (Zhao et al., 2024) training and validation sets as time-sensitive questions. In contrast, we sample an equivalent number of questions from the TriviaQA training set to represent non-time-sensitive questions, which typically have static answers.

Knowledge-aware In the knowledge-aware scenario, identifying whether a user's instruction requires factual knowledge is essential. We use non-retrieval instruct-following data from the Self-RAG (Asai et al., 2023) training set, which GPT-4 classifies as non-knowledge-intensive. We select 2,000 entries for our validation set and 22,801 for training. Additionally, we incorporate all entries from our time-aware data's training and validation sets as knowledge-intensive instructions to complete the final knowledge-aware training and validation sets.

Intent-aware In the intent-aware scenario, it's crucial to identify users' intentions to use retrieval-augmented generation. Due to a lack of data with explicit retrieval intentions, we use Self-Instruct (Wang et al., 2023c) to generate 3,000 user intents from ten handwritten intents. We allocate 2,000 for training, 500 for validation, and 500 for testing. We assemble user queries by sampling 52,949 entries from Self-RAG's non-retrieval-required data, and factual knowledge questions from TAQA and TriviaQA for the training set, with an additional 5,000 for validation. We integrate half of these data with user retrieval intents, alternating the position of intents before and after user inputs, to create inputs with retrieval intents. The remaining data are used as inputs without retrieval intents.

For each scenario, we train a single-layer MLP as the classifier, using the hidden states from the last token in the input as the input to the classification head. In this way, UAR can achieve various criteria's judgements with negligible extra computational cost. We include details of classifiers' training in Appendix E.

3.2 UAR Criteria

We further propose UAR-Criteria to unify the judgements of different types of user instructions in to one unified procedure. During the inference stage, UAR sequentially utilizes four classifiers according to different priorities to determine the correct timing for retrieval calls, and we introduce its details as follows.

Initially, UAR checks whether the user intends to use retrieval augmentation. If so, retrieval is triggered. If not, UAR evaluates whether the input is knowledge-intensive. For non-knowledge-intensive tasks, retrieval is not used. For knowledge-intensive tasks, UAR further assesses whether the knowledge is time-sensitive. Retrieval is necessary for time-sensitive questions. For non-time-sensitive, knowledge-intensive tasks, UAR checks whether the model already has the relevant knowledge, activating retrieval only for unfamiliar questions. In this way, UAR can handle various types of instructions. Specifically, UAR-Criteria activates retrieval for instructions that are time-sensitive, unknown to the model, and have explicit retrieval intent, which facilitates necessary external information supplement. Meanwhile, UAR-Criteria cancels retrieval for those non-knowledge-intensive or LLM-known instructions, which avoids the negative effect of unnecessary retrieval. Meanwhile, since UAR achieves the judgement of multifaceted criteria by linear classifiers, the introduced extra computational cost is negligible.

3.3 Generation with Relevant Information

For instructions requiring retrieval augmentation, we append the retrieved external information with a RAG template to the original user input. Since most of the prevailing LLMs are based on the decoder-only architecture (Brown et al., 2020), UAR can avoid the need to recompute the original instruction. The retriever might fetch information irrelevant to the question, our prompt instructs the model to utilize only the information relevant to the question. This approach helps prevent irrelevant information from misleading the model. An example of our RAG prompt is as follows:

```
{question}
Here are some additional reference passages:
{reference passages}
You can refer to the content of relevant
reference passages to answer the questions.
Now give me the answer.
```

For instructions that do not require retrieval, we allow the model to generate outputs in its original format.

4 Experiments

4.1 Benchmarking Retrieval Timing

We curate an Active Retrieval Benchmark (AR-Bench) to evaluate the accuracy of various active retrieval methods in determining the timing of retrieval. The AR-Bench includes four sub-tasks: intent-aware, knowledge-aware, time-aware and self-aware, covering all the active retrieval scenarios mentioned in this paper. Each sub-task is a binary classification task comprising 8,000 samples, with a 1:1 ratio of positive to negative examples, and these samples do not overlap with the training data of UAR. These four sub-tasks separately evaluate one single active retrieval criterion and we control variables to ensure that each task's retrieval decision solely depends on one single criterion. We introduce details of AR-Bench construction in Appendix A.

4.2 Downstream Tasks

We select six datasets to test UAR's performance in real downstream tasks and its adaptability to different active retrieval scenarios. Since the intent-aware judgement focuses on satisfying users' retrieval intent, which is not reflected on the objective downstream performance, the selected datasets cover the remaining three scenarios: knowledge-aware, time-aware, and self-aware. For knowledge-aware scenario, we use DROP (Dua et al., 2019) and (Cobbe et al., 2021). For time-aware scenario, we use TAQA (Zhao et al., 2024) and FreshQA (Vu et al., 2023). For self-aware scenario, we use TriviaQA (Joshi et al., 2017) and WebQuestions (WQ) (Berant et al., 2013). We provide a detailed introduction to these datasets in Appendix F. In these six datasets, we only use the training sets of TriviaQA anf TAQA for UAR's training, and thus the remaining evaluation dataset can reflect the UAR's out-of-distribution (OOD) performance, which can further verify the effectiveness of UAR in complicated real-world scenarios.

4.3 Baselines

We choose three active retrieval methods as our baseline methods: FLARE (Jiang et al., 2023), Self-RAG (Asai et al., 2023), and SKR (Wang et al.,

Scenario	Intent-aware	Knowledge-aware	Time-aware	Self-aware	Overall
7B Models					
FLARE	61.95	56.76	53.69	53.59	56.50
Self-RAG†	64.26	72.82	47.45	55.95	60.12
SKR	58.73	42.94	76.61	70.28	62.14
UAR	**91.88**	**90.38**	**86.69**	**72.32**	**85.32**
13B Models					
FLARE	65.49	53.54	55.20	54.61	57.21
Self-RAG†	67.80	64.85	54.44	52.49	59.89
SKR	59.00	43.18	79.91	68.70	62.70
UAR	**92.49**	**91.04**	**87.94**	**73.84**	**86.33**

Table 2: Comparisons of active retrieval accuracy on our active retrieval benchmark (AR-Bench). †: Self-RAG is fine-tuned from Llama2-base models. Other methods are based on Llama2-chat models.

2023b), covering two main active retrieval criteria. FLARE determines whether external retrieval is needed by assessing the model's uncertainty about the generated responses. SKR first collects model's self-knowledge (knowns and unknowns) data, then trains a BERT-based (Devlin et al., 2019) classifier to determine whether the model knows a certain question. For questions the model does not know, retrieval augmentation is used. Self-RAG gathers a large amount of knowledge-intensive and instruction-following data (no fact knowledge required), then trains the pre-trained model to only use retrieval augmentation for knowledge-intensive tasks. For downstream tasks, we also include generation with never-retrieval and always-retrieval as baseline methods. The original SKR and FLARE are not based on Llama2, so we re-implement these methods on the Llama2 model. The details of our re-implementation are provided in Appendix B.

4.4 Retrievers

For time-sensitive datasets TAQA and FreshQA, we follow the settings in FreshQA Vu et al. (2023) and use Google Search. For other datasets, following the settings in Self-RAG, we use off-the-shelf Contriever-MS MARCO (Izacard et al., 2022) and retrieve up to ten documents for each input. During generation, we use the top five retrieved documents. For other datasets, following the settings in Self-RAG, we adopt off-the-shelf Contriever-MS MARCO (Izacard et al., 2022) and use the top-5 documents.

4.5 Evaluation Metrics

Following previous work (Asai et al., 2023; Mallen et al., 2023; Schick et al., 2023), we check whether gold answers are included in model's generations to evaluate performance on the DROP, TriviaQA, and WQ datasets, instead of strictly requiring exact matching. For GSM8K, we use the prompts for answer extraction in Kojima et al. (2022) to extract model's answers and then use exact matching to calculate the accuracy. For TAQA and FreshQA, since the golden answers are too long to conduct lexical matching, we use ChatGPT to evaluate whether the model's answers are correct. Details of ChatGPT evaluation are included in Appendix C. Furthermore, for downstream tasks, we also report the percentage of samples that invoke retrieval. For AR-Bench, we use accuracy as the metric. Since AR-Bench is a binary classification task with an equal number of positive and negative samples, accuracy and micro F1 score are equivalent.

4.6 Comparisons on AR-Bench

We show the results in Table 2. We observe that UAR outperforms existing active retrieval methods across all AR-Bench scenarios, demonstrating its versatility and effectiveness. Since baseline methods depend on a single criterion, they struggle with various active retrieval scenarios, which demonstrates the limitation of single criterion and the necessity of multifaceted decision for active retrieval. Additionally, we find FLARE struggle with self-aware scenario, which it is targeted at. We think it is because its uncertainty estimation heavily depends on model calibration and this leads to its poor performance on less calibrated models like chat models (He et al., 2023) or those with fewer parameters. Self-RAG uses the knowledge-intensive nature of tasks as the retrieval criterion, performing well in knowledge-aware scenarios but poorly in others. SKR bases retrieval on the model's knowledge of an answer, excelling in self-aware and time-aware scenarios but failing in others. Additionally,

Dataset	Drop	GSM8K	TriviaQA	WQ	TAQA	FreshQA	Overall
				7B Models			
Never-Ret	57.67$_{(0\%)}$	26.91$_{(0\%)}$	62.15$_{(0\%)}$	59.79$_{(0\%)}$	16.43$_{(0\%)}$	35.64$_{(0\%)}$	43.10
Always-Ret	49.57$_{(100\%)}$	23.65$_{(100\%)}$	68.73$_{(100\%)}$	53.99$_{(100\%)}$	34.49$_{(100\%)}$	65.35$_{(100\%)}$	49.23
				Active Retrieval			
Self-RAG†	39.17$_{(5.7\%)}$	16.07$_{(4.9\%)}$	61.68$_{(53.5\%)}$	43.01$_{(61.9\%)}$	11.09$_{(42.1\%)}$	44.88$_{(51.2\%)}$	35.98
SKR	53.00$_{(61.4\%)}$	26.38$_{(35.3\%)}$	65.39$_{(48.9\%)}$	58.96$_{(26.8\%)}$	30.63$_{(79.9\%)}$	48.84$_{(39.3\%)}$	47.17
FLARE	**56.98**$_{(9.6\%)}$	26.76$_{(45.8\%)}$	65.98$_{(58.8\%)}$	55.46$_{(67.9\%)}$	28.08$_{(63.5\%)}$	57.76$_{(57.4\%)}$	48.50
UAR	52.55$_{(49.7\%)}$	**26.91**$_{(0.1\%)}$	**69.02**$_{(50.1\%)}$	**60.53**$_{(25.0\%)}$	**34.46**$_{(99.7\%)}$	**59.74**$_{(78.5\%)}$	**50.49**
				13B Models			
Never-Ret	58.76$_{(0\%)}$	40.64$_{(0\%)}$	63.18$_{(0\%)}$	57.63$_{(0\%)}$	11.14$_{(0\%)}$	34.98 $_{(0\%)}$	44.39
Always-Ret	54.16$_{(100\%)}$	37.68$_{(100\%)}$	71.02$_{(100\%)}$	54.08$_{(100\%)}$	34.20$_{(100\%)}$	62.05$_{(100\%)}$	52.09
				Active Retrieval			
Self-RAG†	44.68$_{(0.1\%)}$	21.00$_{(0.0\%)}$	62.53$_{(30.0\%)}$	42.37$_{(51.9\%)}$	15.42$_{(37.0\%)}$	39.60$_{(39.3\%)}$	37.60
SKR	56.58$_{(50.9\%)}$	39.35$_{(27.6\%)}$	67.21$_{(49.2\%)}$	56.20$_{(31.5\%)}$	31.66$_{(89.2\%)}$	50.17$_{(45.9\%)}$	50.16
FLARE	58.12$_{(17.5\%)}$	38.05$_{(61.2\%)}$	68.00$_{(54.9\%)}$	53.64$_{(69.6\%)}$	25.40$_{(60.9\%)}$	50.17$_{(55.8\%)}$	48.90
UAR	**58.55**$_{(3.7\%)}$	**40.64**$_{(0.0\%)}$	**71.71**$_{(48.5\%)}$	**59.20**$_{(31.2\%)}$	**34.14**$_{(99.6\%)}$	**55.45** $_{(73.3\%)}$	**53.26**

Table 3: Comparisons of downstream tasks performance. Never-Ret means that retrieval augmentation is never used during generation, while Always-Ret means that retrieval augmentation is used in every generation.†: Self-RAG is fine-tuned from Llama2-base models. Other methods are based on Llama2-chat models.

since SKR uses BERT as the classifier, whose internal knowledge has a significant gap with Llama, it underperforms UAR with value heads based on the Llama's representation, in the self-aware scenario.

4.7 Comparisons on Downstream Tasks

For Self-RAG, we use inference scripts provided by the authors. For FLARE, SKR, UAR, and always-retrieval methods, we use the same prompts to generate responses by incorporating the retrieved information. We introduce the details of generation in Appendix D.

The results are shown in Table 3. We see that UAR leads to the best overall performance across different downstream task scenarios, which indicates its effectiveness. The percentage inside the parentheses represents the proportion of retrieval-invoked samples to the total samples. We analyze each scenario as follows.

UAR does not invoke retrieval when factual knowledge is not needed. The DROP and GSM8K dataset do not require fact knowledge, and using retrieval enhancement will interfere with the model. The results of always-retrieval are worse than never-retrieval. UAR only invokes a small amount of retrieval, while SKR and FLARE incorrectly invoke retrieval extensively. And since UAR avoid unnecessary retrieval[2] and thus prevents affecting the original capabilities of the LLM, it achieves the best results among all active retrieval methods on DROP and GSM8K, coming close to the results of never-retrieval. Although Self-RAG does not incorrectly invoke retrieval, its final performance is not very good because it is fine-tuned based on the base model rather than leveraging the capabilities of the chat model.

UAR accurately invokes retrieval for time-sensitive questions. Since the questions in TAQA and FreshQA are time-sensitive and their answers keep changing, each question requires the retrieval of the latest information. It is evident that the always-retrieval method based on Google Search performs significantly better than the never-retrieval method. For TAQA, UAR almost perfectly invokes retrieval. For FreshQA, UAR also invokes retrieval for most of the questions. In contrast, other methods invoke retrieval less frequently and therefore do not use the latest information for responses, resulting in lower accuracy compared to UAR.

UAR accurately assesses the model's knowledge, avoiding poor retrieval impacts. For questions in TriviaQA and WQ whose answers do not change over time, always-retrieval is sub-optimal and the reason is two-fold: 1. For questions which model knows, retrieval increases unnecessary latency. 2.

[2] UAR based on the 7B model incorrectly invokes retrieval 50% of the time on the DROP dataset. We speculate that this may be due to the limited representation capacity of the 7B model's hidden states. In contrast, the 13B model only incorrectly invokes retrieval 3.7% of the time.

Potential incorrect external information will interfere correct internal knowledge. Retrieving information only for knowledge that the model does not know can mitigate this issue. Compared to SKR, UAR can more accurately determine whether the model knows a particular piece of knowledge. Although SKR and UAR use a comparable number of retrieval calls, the accuracy of SKR's answers is lower than that of UAR, indicating that SKR's retrieval calls are less precise than UAR's. We believe this is because SKR uses independent models, whereas our approach uses hidden states of the original model, resulting in better generalization. Moreover, UAR outperforms always-retrieval with fewer retrieval calls, demonstrating the superiority of the Active Retrieval method.

5 Analysis

5.1 Single Classifiers vs UAR

Scenario	Single Classifier	UAR
Intent-aware	98.29	91.88
Knowledge-aware	99.66	90.38
Time-aware	99.41	86.69
Self-aware	72.56	72.32

Table 4: Comparison between single classifiers and UAR based on Llama2-7B-chat.

Different scenarios have varying levels of discrimination difficulty. As shown in Table 4, the single classifier for the self-aware scenario has the lowest accuracy, which implies that determining whether the model is self-aware is a relatively challenging task. We can also observe that the accuracy of each single classifier is higher than UAR in their respective scenarios. The self-aware classifier may become the bottleneck restricting the performance of UAR, which also results in the accuracy of UAR on the AR-Bench being lower than the accuracy of using a single classifier alone.

5.2 Using the Whole LLM as Classifier

Self-aware	Only Value Head	Whole LLM
Llama2-7B-chat	72.56	75.65
Llama2-13B-chat	73.48	76.28

Table 5: Comparison of the performance between training a value head as the classifier and training a entire large language model as the classifier.

To improve the performance bottleneck of the self-aware classifier, we attempt to fine-tune the entire large language model as the classifier. From the results in Table 5, we can observe that on both 7B and 13B models, fine-tuning the entire model only achieves slight higher accuracy compared to just fine-tuning a lightweight value head. Using a whole LLM as the classifier, UAR's inference latency and required parameters will significantly increase. Therefore, we use lightweight value heads as classifiers, ensuring the efficiency of the entire framework with minimal performance loss.

5.3 The Impact of Document Number

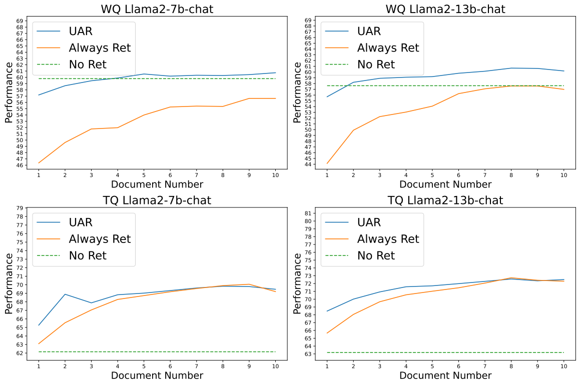


Figure 3: The impact of the number of reference documents on model performance.

We evaluate performance on the TriviaQA (TQ) and WebQuestions (WQ) datasets by varying the number of reference documents from 1 to 10. The results, shown in Figure 3, indicate that on the WQ dataset, the always-retrieval method performs worse than the never-retrieval method, possibly because some documents disrupt the correct knowledge within the model. UAR reduces retrieval frequency, enabling more precise retrieval calls and outperforming the never-retrieval method. On the TQ dataset, always-retrieval outperforms never-retrieval, and performance improves with more documents, suggesting useful information might be in lower-ranked documents. UAR performs best with fewer documents. With more documents, it matches the performance of always-retrieval, although it requires significantly fewer retrieval calls.

6 Conclusion

In this paper, we introduce UAR, a unified active retrieval framework for retrieval-augmented generation. Unlike existing methods that rely on a single criterion, UAR incorporates four orthogonal criteria into plug-and-play classification tasks, enabling comprehensive retrieval timing judgments with minimal inference cost and no loss of model capabilities. We also introduce UAR-Criteria for processing various active retrieval scenarios uniformly. We curate the Active Retrieval Benchmark (AR-Bench) to assess the retrieval timing accuracy of active retrieval methods across different scenarios. Experimental results demonstrate that UAR significantly outperforms existing methods on AR-Bench and downstream tasks, highlighting its effectiveness and benefits to downstream applications.

Limitations

We summarize limitations of our work as follows:

- Our experiments primarily focus on the generation of short texts, such as in knowledge-based question answering, and involve only a single retrieval call. How to implement multiple active retrieval calls within longer text responses remains an area for future investigation.
- Our active retrieval criteria are primarily derived from our experience in practical applications, which may overlook some active retrieval scenarios.
- Our classifier is based on a single-layer MLP network. Whether using a deeper network can further enhance performance remains to be explored.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (No. 62236004). The computations in this research were performed using the CFFF platform of Fudan University.

References

Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul Ronald Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, and et al. 2023. Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805.

Anthropic. 2023. Introducing claude.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *CoRR*, abs/2310.11511.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *CoRR*, abs/2309.16609.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1533–1544. ACL.

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*.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, and et al. 2024. Internlm2 technical report. *CoRR*, abs/2403.17297.

Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Qinyuan Cheng, Tianxiang Sun, Xiangyang Liu, Wenwei Zhang, Zhangyue Yin, Shimin Li, Linyang Li, Zhengfu He, Kai Chen, and Xipeng Qiu. 2024. Can AI assistants know what they don't know? *CoRR*, abs/2401.13275.

Qinyuan Cheng, Tianxiang Sun, Wenwei Zhang, Siyin Wang, Xiangyang Liu, Mozhi Zhang, Junliang He, Mianqiu Huang, Zhangyue Yin, Kai Chen, and Xipeng Qiu. 2023. Evaluating hallucinations in chinese large language models. *CoRR*, abs/2310.03368.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

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. *Preprint*, arXiv:2402.10612.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. 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, Volume 1 (Long and Short Papers)*, pages 2368–2378. Association for Computational Linguistics.

Constanza Fierro, Nicolas Garneau, Emanuele Bugliarello, Yova Kementchedjhieva, and Anders Søgaard. 2024. Mulan: A study of fact mutability in language models. *CoRR*, abs/2404.03036.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *Preprint*, arXiv:2002.08909.

Guande He, Peng Cui, Jianfei Chen, Wenbo Hu, and Jun Zhu. 2023. Investigating uncertainty calibration of aligned language models under the multiple-choice setting. *CoRR*, abs/2310.11732.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022.

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, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7969–7992. Association for Computational Linguistics.

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.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know. *CoRR*, abs/2207.05221.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022a. Teaching models to express their uncertainty in words. *Trans. Mach. Learn. Res.*, 2022.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022b. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3214–3252. Association for Computational Linguistics.

Yanming Liu, Xinyue Peng, Xuhong Zhang, Weihao Liu, Jianwei Yin, Jiannan Cao, and Tianyu Du. 2024. RA-ISF: learning to answer and understand from retrieval augmentation via iterative self-feedback. *CoRR*, abs/2403.06840.

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. 2022. Introducing chatgpt.

OpenAI. 2023. Gpt-4 technical report.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 9248–9274. Association for Computational Linguistics.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 31210–31227. PMLR.

Tianxiang Sun, Xiaotian Zhang, Zhengfu He, Peng Li, Qinyuan Cheng, Xiangyang Liu, Hang Yan, Yunfan Shao, Qiong Tang, Shiduo Zhang, Xingjian Zhao, Ke Chen, Yining Zheng, Zhejian Zhou, Ruixiao Li, Jun Zhan, Yunhua Zhou, Linyang Li, Xiaogui Yang, Lingling Wu, Zhangyue Yin, Xuanjing Huang, Yu-Gang Jiang, and Xipeng Qiu. 2024. Moss: An open conversational large language model. *Machine Intelligence Research*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

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. *CoRR*, abs/2307.09288.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry W. Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc V. Le, and Thang Luong. 2023. Freshllms: Refreshing large language models with search engine augmentation. *CoRR*, abs/2310.03214.

Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Jiayang Cheng, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, Yidong Wang, Linyi Yang, Jindong Wang, Xing Xie, Zheng Zhang, and Yue Zhang. 2023a. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *CoRR*, abs/2310.07521.

Hongru Wang, Boyang Xue, Baohang Zhou, Tianhua Zhang, Cunxiang Wang, Guanhua Chen, Huimin Wang, and Kam-Fai Wong. 2024. Self-dc: When to retrieve and when to generate? self divide-and-conquer for compositional unknown questions. *CoRR*, abs/2402.13514.

Yile Wang, Peng Li, Maosong Sun, and Yang Liu. 2023b. Self-knowledge guided retrieval augmentation for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 10303–10315. Association for Computational Linguistics.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023c. Self-instruct: Aligning language models with self-generated instructions. 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 13484–13508. Association for Computational Linguistics.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, Fan Yang, Fei Deng, Feng Wang, Feng Liu, Guangwei Ai, Guosheng Dong, Haizhou Zhao, Hang Xu, Haoze Sun, Hongda Zhang, Hui Liu, Jiaming Ji, Jian Xie, Juntao Dai, Kun Fang, Lei Su, Liang Song, Lifeng Liu, Liyun Ru, Luyao Ma, Mang Wang, Mickel Liu, MingAn Lin, Nuolan Nie, Peidong Guo, Ruiyang Sun, Tao Zhang, Tianpeng Li, Tianyu Li, Wei Cheng, Weipeng Chen, Xiangrong Zeng, Xiaochuan Wang, Xiaoxi Chen, Xin Men, Xin Yu, Xuehai Pan, Yanjun Shen, Yiding Wang, Yiyu Li, Youxin Jiang, Yuchen Gao, Yupeng Zhang, Zenan Zhou, and Zhiying Wu. 2023. Baichuan 2: Open large-scale language models. *CoRR*, abs/2309.10305.

Zhaorui Yang, Qian Liu, Tianyu Pang, Han Wang, Haozhe Feng, Minfeng Zhu, and Wei Chen. 2024. Self-distillation bridges distribution gap in language model fine-tuning. *CoRR*, abs/2402.13669.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuanjing Huang. 2023. Do large language models know what they don't know? In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 8653–8665. Association for Computational Linguistics.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2023. Making retrieval-augmented language models robust to irrelevant context. *CoRR*, abs/2310.01558.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. GLM-130B: an open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Hanning Zhang, Shizhe Diao, Yong Lin, Yi R. Fung, Qing Lian, Xingyao Wang, Yangyi Chen, Heng Ji, and Tong Zhang. 2023. R-tuning: Teaching large language models to refuse unknown questions. *CoRR*, abs/2311.09677.

Bowen Zhao, Zander Brumbaugh, Yizhong Wang, Hannaneh Hajishirzi, and Noah A. Smith. 2024. Set the clock: Temporal alignment of pretrained language models. *CoRR*, abs/2402.16797.

A Details of AR-Bench Construction

For the self-aware task, we employ the same method as described in Section 3.1 to construct test samples on the TriviaQA validation set. Questions the model does not know are marked as requiring retrieval. The test set comprise 4000 questions the model knows and 4000 questions it does not.

For the time-aware task, we use 4000 time-sensitive questions from the TAQA test set as inputs requiring retrieval, and 4000 questions the model knows from the TriviaQA validation set as inputs not requiring retrieval.

For the knowledge-aware task, we use 4000 samples from the Self-RAG non-retrieval training data as inputs not requiring retrieval, and combine 2000 time-sensitive questions from the TAQA test set with 2000 questions the model does not know from the TriviaQA validation set as inputs requiring retrieval.

For the intent-aware task, we use 4000 questions the model knows from the TriviaQA validation set and 4000 instructions from the Self-RAG non-retrieval training data, half of which are concatenated with user retrieval intents as inputs requiring retrieval, and the other half as inputs not requiring retrieval.

It is important to note that the self-aware data for different models may vary, leading to different AR-Benches for different models. In our experiments, we curate two separate AR-Benches for Llama2-7B-chat and Llama2-13B-chat respectively.

B Details of Baselines Re-implementation

B.1 FLARE

In implementing FLARE, we make two modifications. First, we conduct experiments based on the Llama2-chat series of models, rather than using text-davinci-003. Second, we eliminate the initial retrieval step in FLARE since our setting is active retrieval rather than passive retrieval. We find that FLARE based on Llama2 struggle to achieve satisfactory results, which we suspect may be due to poor calibration of the Llama2-7B-chat and Llama2-13B-chat models. The uncertainty estimation in FLARE heavily relies on model calibration, making it challenging to adapt to poorly calibrated models. Therefore, on the AR-Bench, we conduct a direct search for the best retrieval thresholds for FLARE, ultimately setting them at 0.006 and 0.02 for the Llama2-7B-chat and Llama2-13B-chat models, respectively.

B.2 SKR

In implementing SKR, we first use the 849 original pieces of data provided by the authors of SKR and collect self-knowledge data for the Llama2-7B-chat model according to the scripts in SKR's code

Training Hyper-parameters	
Optimizer	AdamW
Warmup Steps	0
Learning Rate	2e-5
Batch Size	32
Train Epochs	5
LR Scheduler	Linear
Max-seq-length	512

Table 6: Training hyper-parameters of SKR.

repository. We obtain 15 questions that the model does not know and 143 questions that it knows, and find that these data are not sufficient to train an effective BERT classifier. Therefore, we use the data from our training data of the self-aware classifier to train the BERT classifier for SKR. Our training hyper-parameters are shown in Table 6.

C ChatGPT Evaluation

We use gpt-3.5-turbo-instruct as the evaluator. During the evaluation, we input the correct answer and the answer to be evaluated into gpt-3.5, and then let the model compare the correct answer with the answer to be evaluated to determine if the latter is correct. Following Shao et al. (2023), we use the following prompt for evaluation.

```
In the following task, you are given a Question,
 a model Prediction for the Question, and a
Ground-truth Answer to the Question. You should
decide whether the model Prediction implies the
Ground-truth Answer.

Question:
{question}

Prediction:
{predicted answer}

Ground-truth Answer:
{ground-truth answer}
Does the Prediction imply the Ground-truth
Answer? Output Yes or No:
```

D Details of Generation

D.1 Self-RAG

We use the inference script provided by the Self-RAG authors for generation. We determine the need for retrieval by whether the retrieval special token appears in the generated response. For datasets using Contriever-MS MARCO as the retriever, we provide all 10 documents retrieved to Self-RAG for generation.

D.2 Generation without Retrieval

For the DROP dataset, we use the following prompt:

```
Please answer the question based on the given
passage.
Passage: {passage in the dataset}
Question: {question}
Now give me the answer.
```

For the GSM8K dataset, we use the following prompt:

```
Answer the math word question step by step. Your
 answer needs to end with 'The answer is'.
Question: {question}
Let's think step by step and give me the answer.
```

For other datasets, we directly input the question to the model:

```
{question}
```

D.3 Generation with Retrieval

For the DROP dataset, we use the following prompt:

```
Please answer the question based on the given
passage.
Passage: {passage in the dataset}
Question: {question}

Here are some additional reference passages:
{retrieved documents}

You can refer to the content of relevant
reference passages to answer the questions.
Now give me the answer.
```

For the GSM8K dataset, we use the following prompt:

```
Answer the math word question step by step. Your
 answer needs to end with 'The answer is'
Question: {question}

Here are some additional reference passages:
{retrieved documents}

You can refer to the content of relevant
reference passages to answer the questions.
Let's think step by step and give me the answer.
```

For other datasets, we use the following prompt:

```
{question}

Here are some additional reference passages:
{retrieved documents}

You can refer to the content of relevant
reference passages to answer the questions.
Now give me the answer.
```

E Details of UAR Training

When training the UAR classifiers, we set the batch size to 32 and train for a total of 10 epochs, saving after each epoch and selecting the checkpoint that perform best on the validation set. We conduct a grid search on the validation set and ultimately determine the learning rate to be 5e-5. Our classifier is a fully connected layer with an input dimension equal to the hidden state dimension and an output dimension of 2.

F Downstream Task Datasets

For knowledge-aware scenario, we use the validation set of DROP (Dua et al., 2019) and the test set of GSM8K (Cobbe et al., 2021) as the test sets. DROP is a reading comprehension benchmark, which needs the model to answer questions based on given paragraphs. GSM8K is a dataset containing diverse grade school math word problems, primarily used to assess the reasoning ability of models. These two datasets evaluate the model's abstract abilities, e.g., reading comprehension and math reasoning, and thus do not require extra fact knowledge. Therefore, they can measure the ability of active retrieval methods to avoid unnecessary retrieval for scenarios that requires little fact knowledge.

For time-aware scenario, we use the test set of TAQA (Zhao et al., 2024) and questions whose answers will change over time from FreshQA (Vu et al., 2023) (We remove questions with false premises). Since these questions are time-sensitive, the active retrieval system need to retrieve real-time information for every question.

For self-aware scenario, we use the validation set of TriviaQA (Joshi et al., 2017) and the test set of WebQuestions (WQ) (Berant et al., 2013). These test samples are non-time-sensitive questions. The active retrieval system only needs to retrieve questions which the model does not know, and try to achieve high answer accuracy with an appropriate number of retrieval calls.