Vrije Universiteit Amsterdam

Universiteit van Amsterdam

Master Thesis

Evaluation of INFO: A GPT-3.5 and RAG Based Query-Answer System

---

Author: Yixin Hu (2762621)

1st supervisor:	Adam Belloum
daily supervisor:	Maxime Moulin, Thomas Hulard (no particular order) (Mcdermott)
2nd reader:	Chrysa Papagianni

A thesis submitted in fulfillment of the requirements for the joint UvA-VU Master of Science degree in Computer Science

July 9, 2024

For people who always support and love me

Abstract

This study focuses on developing and implementing an evaluation benchmark designed to assess the performance of INFO, an intelligent agent to answer engineering-related questions based on Retrieval-Augmented Generation (RAG) systems. By moving beyond traditional metrics and adopting innovative methods, we created a domain-specific dataset for benchmarking and developed 4 metrics to comprehensively evaluate INFO. The evaluation results provide deeper insights into the system's effectiveness and performance in real-world scenarios. The result of INFO reveals high accuracy and relevance in answer generation. Through evaluation, we found areas for improvement, like in the retrieval phase and document parsing methods. The benchmark has high scalability since it is adaptable to other systems, and can guide future enhancements, making it a significant tool for ongoing development and optimization.

Contents

List of Figures

LIST OF FIGURES

List of Tables

LIST OF TABLES

1

Introduction

Large Language Models (LLMs) have revolutionized various applications across fields from academic achievements to industry scenarios. Even while these models have demonstrated impressive overall ability, they still face several difficulties, such as factual hallucinations (1) and the lack of domain-specific expertise (2).

Retrieval Augmented Generation (RAG) (3) empowers LLM models by incorporating external knowledge to enhance the relevance of the results and the availability in industrial scenarios. Especially for industries that need domain-specific expertise and to keep data confidential, embedding relevant data sources is crucial to let LLMs generate more accurate and reliable responses. For example, consider an engineer who needs to inquire about a company's internal documents. A conventional LLM trained solely on open-source data would struggle to provide an accurate response due to the lack of specific internal knowledge. However, integrating RAG can equip the LLM with essential context, thereby enabling it to generate more precise and reliable responses while ensuring data security.

To address this, we have developed an intelligent question-answer system named INFO, utilizing GPT-3.5-Turbo enhanced with RAG to integrate external engineering knowledge specifically. This system allows engineers to use natural language to efficiently query and retrieve necessary information, bypassing the traditional method of manually searching through extensive datasets or relying on inefficient search techniques. Consequently, this innovation not only conserves valuable labor time but also significantly improves the engineers' experience.

Despite the assistance provided by RAG, Large Language Models (LLMs) continue to face challenges related to unreliable generation. Bian et al. (4) note that LLMs may generate responses that deviate from external information due to the influence of inaccurate contextual data. As a result, LLMs struggle to consistently deliver accurate and reliable responses. A comprehensive study that examines the impact of these factors on RAG, as well as strategies to enhance model resilience against such setbacks, remains lacking. Therefore, there is a pressing need to develop an evaluation framework for RAG that effectively assesses both the utilization of external information and the robustness against its inherent limitations.

Despite the extensive development of benchmarks for evaluating Large Language Models (LLMs), current efforts largely focus on constructing benchmarks suited for generic LLM applications. For example, the comprehensive evaluation framework developed by Liang et al. (5) constructs benchmarks across multiple languages, various scenarios, and diverse tasks. While such approaches broaden the scope of LLM evaluation, they frequently do not meet the specific requirements of domain-specific tasks, particularly within industrial settings.

Industrial applications of RAG systems often necessitate integration with specialized databases containing proprietary or niche information that is typically absent from general-purpose databases like Wikipedia. This requirement for specificity demands benchmarks that are customized not only to the query format but also to the unique content specific to the domain. Such tailored benchmarks ensure that evaluations truly reflect the system's effectiveness in practical industrial scenarios, thus addressing the nuanced needs of these applications.

1.1 Problem and Research Question

Consequently, there is a pronounced need for an evaluation framework that accommodates both the general capabilities of LLMs and the specific challenges presented by domain-specific applications. Developing such a framework would permit a more precise assessment of RAG-enhanced systems, furnishing valuable insights into their operational effectiveness and informing necessary refinements. This becomes particularly vital for systems where the accuracy and relevance to the domain are critical, highlighting the necessity for benchmarks that match the sophistication and specificity of the technologies they aim to evaluate.

From this perspective, the research question is what type of metrics and how can the evaluation tool quantify the performance of infoWiz?

INFO has substantially enhanced the efficiency of engineers by enabling rapid access to accurate and relevant information, thereby revolutionizing the traditional method whereby engineers relied solely on manual searches guided by memory. However, no system is infallible, and INFO has not yet achieved 100% accuracy. This limitation raises two critical questions:

Performance Assessment:

- How can we accurately scale the performance of INFO?

System Enhancement:

- What strategies can be employed to refine INFO further?

For the system to provide reliable service, it is crucial to establish a quantitative performance score prior to deployment in real-world scenarios. Moreover, employing a multidimensional evaluation framework will not only pinpoint current limitations but also guide targeted enhancements, ensuring that improvements are both purposeful and effective.

1.2 Contribution

This research makes several pivotal contributions to the field of retrieval-augmented generation for Large Language Models (LLMs), specifically within domain-specific applications:

- Establish a product-specific benchmark for INFO: We created a tailored dataset for the INFO system which specifically contains domain-related questions and golden answers.

- Bring insights for future improvements based on quantitative assessment: This method extends beyond traditional evaluations by multidimensional qualitative analyses, thus providing a thorough understanding of the system's performance across multiple aspects.

- Develop a reusable evaluation pipeline for similar products: Our framework is designed for adaptability across various domain-specific RAG systems, offering a scalable solution for evaluating performance metrics in diverse settings.

This study is organized as follows: Section 2 introduces the background for the study, including LLM, RAG and other related techniques. Section 3 introduces and compares related works of our study. Section 4 introduces the methodology we have used and the experiment. Then, Section 5 displays the result and visualizes it to analyze it. Section 6 will discuss the findings in the result and discuss some remaining problems for future work. Finally, we draw a brief conclusion in Section 7 by summarizing the main contributions.

2

Background

2.1 Large Language Models (LLMs)

LLMs are large-scale, pre-trained, language models based on artificial intelligence technology. With language modeling, machines can understand and communicate in human language. This research has gained wide attention and can be regarded as 4 main phrases as in Figure 2.1. (6) We set the time mostly based on the publication date of the most representative research at each step, therefore the period for each stage may not be entirely correct.

- *Statistical language models (SLMs).* Originating in the 1990s, SLMs are primarily built on the principles of statistics. The dominating idea is based on Markov chain models, e.g., predicting subsequent words based on recent contextual history. The SLMs with a fixed context length n are typically known as n-gram models (7). It predicts the next word based on the likelihood of occurrence of a sequence of words and estimates the probability of text as the product of their word probabilities. SLMs have been widely used in many information retrieval (IR) and natural language processing (NLP) tasks. However, natural language is always sparse so n-gram models can not efficiently capture the pattern in diverse and various texts. N-gram models also face the curse of dimensionality where estimation of high-order models becomes computationally infeasible due to the exponential growth in the number of possible word combinations.

- *Neural language models (NLM).* NLMs represent a more advanced phase in the development of language models, where neural networks such as multi-layer perceptron

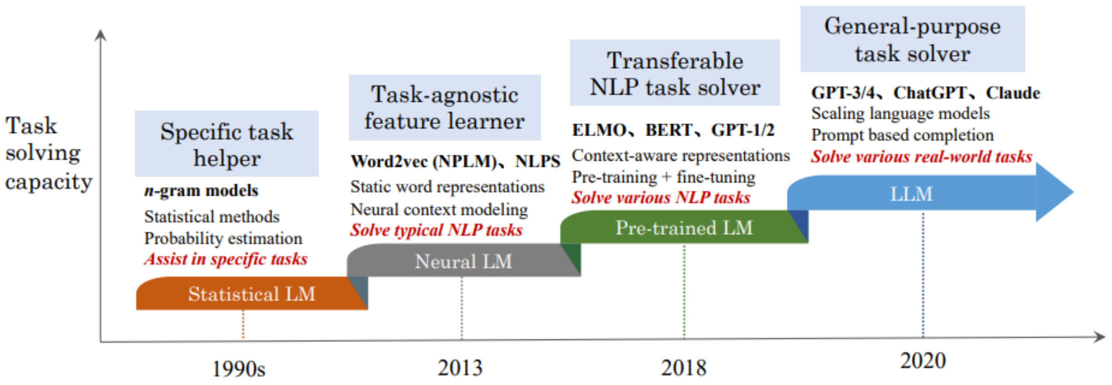


Figure 2.1: Language models (LM) path of evolution

(MLP) and recurrent neural networks (RNNs) (8) are employed. For the shortcomings in n-gram models, the work in (9) solves the curse of dimensionality by introducing a distributed representation of words (commonly referred to as word embeddings). This technique fundamentally changed the way context is processed by aggregating features into distributed word vectors. Building on the idea of distributed representations, word2vec framework (10) was introduced specifically designed to efficiently compute word embeddings using neural networks. These researchers first introduced the use of language models for representation learning (beyond word sequence modeling) and had a huge impact on NLP field.

- *Pre-trained language models (PLM).* Pre-trained Language Models (PLM) have set a new benchmark in language processing with the advent of ELMo (11). Different from word2vec where one word corresponds to one fixed vector, ELMo utilizes a bidirectional LSTM network to generate dynamic contextualized word representations. Furthermore, BERT was invented (12) with transformer (13) architecture, marking a significant evolution by enabling bidirectional training on extensive unlabeled datasets. These models serve as general-purpose semantic features that significantly boost the performance of NLP tasks. Following this study, various models like GPT-2 (14) and BART (15) emerged, establishing a robust "pre-training and fine-tuning" paradigm widely embraced in the field. It is often necessary to fine-tune the PLM to accommodate different downstream tasks.

- *Large language models (LLM).* In PLMs, scaling model size or data size usually improves model ability on downstream tasks following scaling law (16). Even though just increasing model and data sizes and maintaining similar structure, these models achieve unprecedented processing capabilities. They exhibit unique emergent abilities that surpass their predecessors in handling complex tasks, including effective few-shot learning. Consequently, the academic circles decided to name these large PLMs as "large language models". LLMs now specifically refer to these extensive models (17) (18), which continue to attract substantial research interest. Notably, applications like ChatGPT showcase remarkable conversational skills with humans, highlighting the practical potential of LLMs. This surge in research and development is reflected in the increasing volume of related academic publications post the release of such models.

In summary, Large Language Models (LLMs) derive from years of language modeling research than a brand-new concept. Now LLMs show significant problem-solving ability. These days, the AI community is being greatly impacted by LLMs. The problem-solving ability and development of LLMs are revolutionizing the fields of AI research. To a certain extent, LLMs can be used as general-purpose language task solvers in NLP, and the research paradigm is moving in favor of LLM utilization.

2.2 Retrieval Augmented Generation (RAG)

However, despite the success of LLMs, they still face challenges. The issues are, e.g., LLMs knowledge is usually out of date (19), sometimes generates hallucination content (1), also the lack of domain-specific expertise (2).

To address these limitations, The concept of Retrieval-Augmented Generation (RAG) (3) was proposed in 2020 and combines a pre-trained retriever for the first time with a pre-trained seq2seq model with end-to-end fine-tuning. It was after NLP moved into the era of LLMs in 2022 that RAG emerged as a compelling solution, especially after ChatGPT was proposed as shown in Figure 2.2 (20).

In common definition, RAG has two main phases: retrieval and generation. The retriever utilizes an embedding model to retrieve relevant documents based on the problem. The retriever is the core component of the RAG framework, responsible for retrieving relevant information from an extensive knowledge base. The retriever analyzes the user's input query and retrieves the most relevant paragraphs. The generator then uses these retrieved contexts to generate answers based on the LLM. It is responsible for transforming the search results into natural and fluent text. Its input includes not only traditional contextual

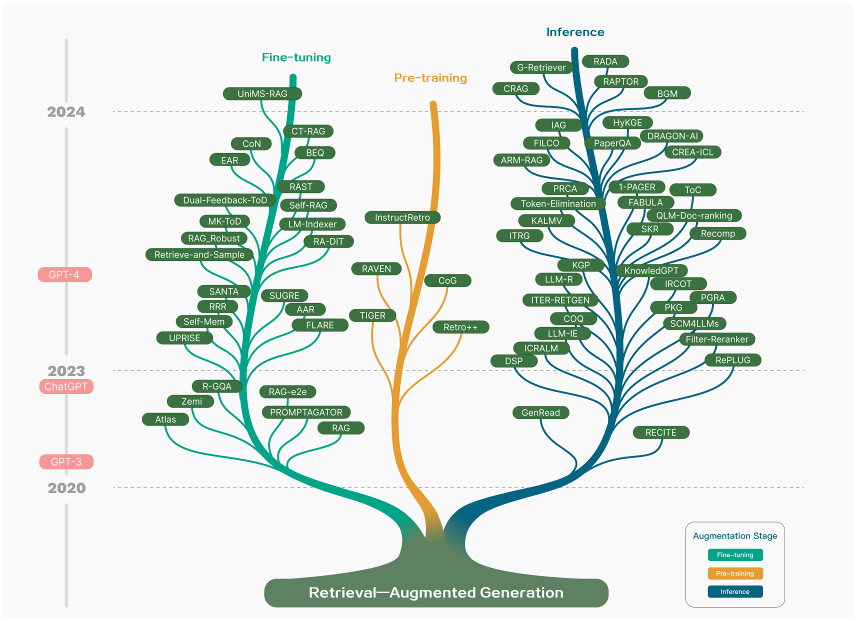


Figure 2.2: Technology tree of RAG research

information but also relevant text snippets obtained by the retriever. This helps LLMs to deepen their understanding of the context of the question and generate richer responses.

Because of the high cost of training high-performance large models, academia and industry have attempted to enhance model generation by incorporating RAG modules in the inference phase to integrate external knowledge in a more cost-effective way. RAG provides a more efficient solution for complex knowledge-intensive tasks in large models by optimizing key parts such as retrievers and generators. RAG improves the relevancy of the answers while decreasing the rate of mistakes in LLMs.

The scope of RAG search is also gradually expanding. Early RAG focused on open-source, unstructured knowledge, e.g., Wikipedia. As the scope of search expands, structured, high-quality data can also be used as a knowledge source. Besides RAG development timeline, in Figure 2.2, the tree illustrates two more aspects of RAG development:

- Augmentation data: Data sources include unstructured data, structured data and LLM-generated content.
- Augmentation stage: Retrieval-Augmented Generation (RAG) can enhance performance at three stages: pre-training, fine-tuning, and inference, represented as three branches in the tree.

Pre-training involves initially training a model on a large-scale generalized dataset to learn a wide range of linguistic patterns and knowledge. The inception of RAG coincided with the rise of the Transformer architecture, aiming to combine broader knowledge with pre-training models for more robust representations. Inference occurs when a trained model generates predictions based on new input data. With the advent of LLMs, RAG research has focused on exploiting the powerful in-context learning (ICL) capabilities of LLMs to tackle knowledge-intensive tasks. By providing better information, RAG enhances LLMs' abilities to handle more complex tasks during inference. Fine-tuning is the further training of a pre-trained model for a specific task or domain to optimize its performance in that particular application. As LLMs have developed and found widespread use, domain-specific retrieval has improved accuracy and relevance by tailoring information to specific tasks.

Overall, RAG ensures that the generated content is not only contextually accurate but also deeply aligned with specialized knowledge, making it particularly suitable for industry scenarios. Moreover, RAG models have achieved great results in multiple tasks such as domain-specific question answering (21), which will be discussed in the following chapter.

2.3 Prompt Engineering

Prompt engineering has been an important method for improving LLM performance. This method makes clear, task-specific clues for computers in natural language, and representative examples are chosen with care to be included in the prompt. Without changing the parameters, the output of LLMs can improved by strategically designed prompts, enabling them to excel across diverse tasks and domains. It is important because of its ability to steer responses. Therefore LLMs can be more flexible and applicable in a wider range of industries. Next, we will provide a brief overview of prompt engineering techniques, spanning from basic to some latest advanced ones.

- *Zero-Shot Prompting.* No need for training on massive amounts of data, LLMs can perform some zero-shot tasks (22). In the prompt, there are no additional examples to guide the model. The model will only leverage its knowledge to produce output based on the prompt.

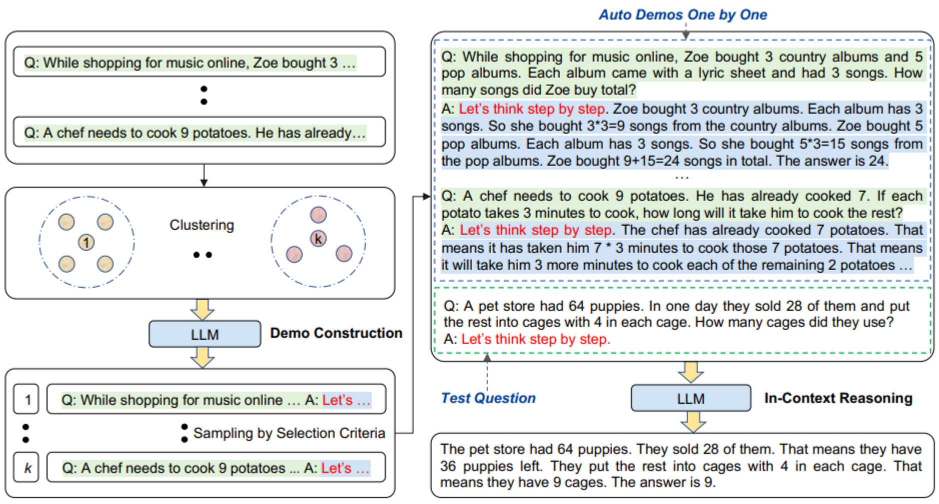
- *Few-Shot Prompting.* Few-shot prompting enhances model understanding by providing a limited number of examples (23). This technique is particularly effective for complex tasks, where even a small number of high-quality examples can significantly improve performance. However, longer inputs lead to increased token consumption, and the impact of prompt selection and composition on the results is still not fully understood.

- *Chain-of-Thought (CoT) Prompting.* The two techniques mentioned above belong to In-Context Learning (ICL). The core idea of ICL is to learn from analogies, mimicking the human learning process. Initially, ICL is given some examples that form the context of the task. These examples are written in natural language templates. Then, ICL connects the query question (i.e., the input) with a contextual presentation (some relevant cases) to form the input with hints, and feeds it into the language model for prediction. However, this method has limitations such as unstable. To compensate for these shortcomings, Chain-of-Thought (CoT) adds thoughts in the middle, in each step (24). Unlike traditional ICL, which provides more input to generate output, CoT involves predicting the "thought process" (referred to as rationale in academic fields) along with the answer. These thought processes are used as hints to get better answers and do not need to be shown for actual use. Instead of rigidly providing sample questions and answers, intermediate reasoning sessions are given so that the model learns the logic of reasoning and thinking in the intermediate process. This method not only mimics human learning process as learning from examples but more closely to the core of human intelligence, human cognitive processes.

  However, high-quality examples which needed in CoT usually involve manual efforts. These could lead to suboptimal results. To mitigate this limitation, Auto-CoT (25) was introduced. As the name suggests, Auto-CoT automatically generates rationales by instructing LLMs with a "Let's think step-by-step" prompt. This automatic process may contain errors, so it is important to build diverse demonstrations. First, Auto-CoT divides questions into clusters. Then it selects a representative question from each cluster to generate its reasoning chain. The process is illustrated in Figure 2.3.

  Auto-CoT has even better results than manual methods. One interpretation is that Auto-CoT does not apply a fixed template. Instead, each task generates its own set of examples. This is because the problem sets differ, leading to more various results.

- *Self-Consistency.* One of the more advanced techniques than CoT prompting is self-consistency (26). First, it uses a set of huamn set CoT examples as prompt as LLMs input. Then it samples a set of candidate outputs from the LLM to generate a set of different candidate inference paths. Eventually, it selects the most consistent



Q: While shopping for music online, Zoe bought 3 …

Q: A chef needs to cook 9 potatoes. He has already…

1 Q: While shopping for music online … A: Let's …

k Q: A chef needs to cook 9 potatoes ... A: Let's …

Q: While shopping for music online, Zoe bought 3 country albums and 5 pop albums. Each album came with a lyric sheet and had 3 songs. How many songs did Zoe buy total?
A: Let's think step by step. Zoe bought 3 country albums. Each album has 3 songs. So she bought 3*3=9 songs from the country albums. Zoe bought 5 pop albums. Each album has 3 songs. So she bought 5*3=15 songs from the pop albums. Zoe bought 9+15=24 songs in total. The answer is 24.
…
Q: A chef needs to cook 9 potatoes. He has already cooked 7. If each potato takes 3 minutes to cook, how long will it take him to cook the rest?
A: Let's think step by step. The chef has already cooked 7 potatoes. That means it has taken him 7 * 3 minutes to cook those 7 potatoes. That means it will take him 3 more minutes to cook each of the remaining 2 potatoes …

Q: A pet store had 64 puppies. In one day they sold 28 of them and put the rest into cages with 4 in each cage. How many cages did they use?
A: Let's think step by step.

The pet store had 64 puppies. They sold 28 of them. That means they have 36 puppies left. They put the rest into cages with 4 in each cage. That means they have 9 cages. The answer is 9.

Figure 2.3: Overview of the Auto-CoT method

answer among multiple reasoning paths. By aggregating multiple responses to the same prompt, self-consistency ensures that the final answer to an input represents a consensus vote, which tends to be more reliable and accurate than simple CoT completions on their own. Even when regular CoT is ineffective, self-consistency still be able to improve results.

2.4 Question-Answer System

Question Answering System (QA system) is a NLP technology designed to automatically answer users' questions. These systems utilize various techniques and algorithms to understand the users and extract relevant answers from databases, documents, or other sources of information.

According to the type of implementation, the QA system can be categorized into different approaches (27) as shown in Figure 2.4.

- *Information retrieval-based QA*. These systems answer questions by utilizing search engines to exploit the relevant documents or paragraphs that contain the target answers. It leverages information retrieval (IR) techniques, e.g. keyword search to identify the most likely relevant context.

- *Knowledge-based QA*. These systems search for answers from structured data sources (as a knowledge base) and exploit database queries to find accurate responses. Because the knowledge is well-curated, this approach is generally more dependable than

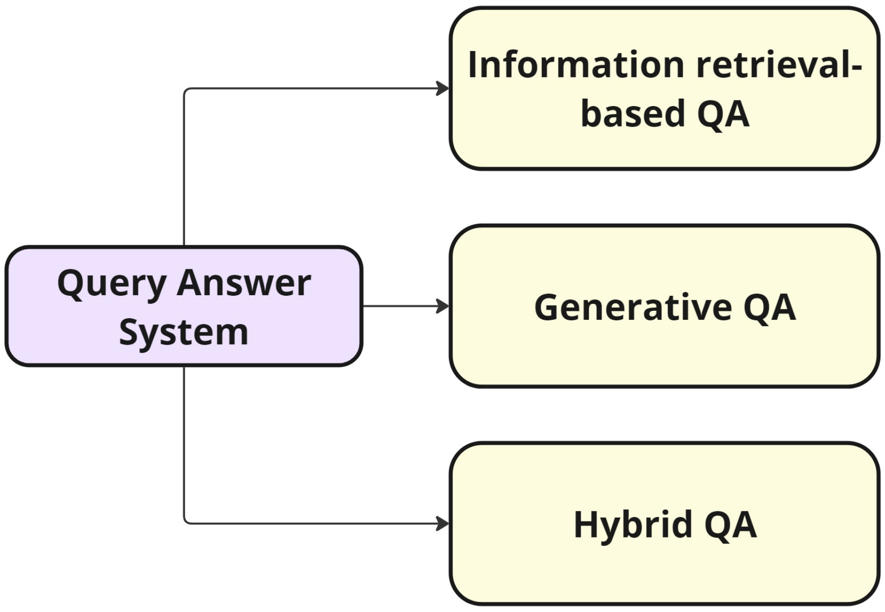


Figure 2.4: Types of QA system

other QA procedures. However, the depth of the knowledge base and the efficiency of the techniques employed to search and retrieve data may be factors limiting its performance. Furthermore, creating and keeping up a knowledge base can be highly costly.

- *Generative QA*. These systems directly generate answers based on users' questions using advanced natural language processing techniques. Because they can provide answers that are more human-like and cover a wide range of queries, generative quality assurance systems are quite strong, making them ideal for use in chatbots. However, the complexity of the model and the caliber and variety of the training data may have an impact on how well they work. It is also important to remember that, to enhance the overall effectiveness, generative QA systems are frequently used in conjunction with other QA methodologies, such as knowledge-based or information retrieval-based QA.

- *Hybrid QA*. These systems use a mix of resources, and they often represent the combination of the previous three listed approaches. Hybrid QA systems leverage the strengths of each method to improve accuracy and response quality. Additionally adaptable, it can adjust to various inquiry kinds and degrees of complexity. However, compared to employing a single QA approach, planning and deploying a hybrid QA system might be more difficult and resource-intensive.

In addition to classifying QA systems based on their implementation approach, their foundational features can be presented from three perspectives: domain type, system type, and question type.

If classified by domain type, QA systems can be close-domain or open-domain. Open-domain systems convert all-natural language questions into structured queries, while closed-domain systems are pre-defined to accept just specific sorts of questions.

From a system-type perspective, analysis shows that whereas open systems are generally accessible and widely used due to their enormous base of global contributors, closed systems are dependent on internal knowledge bases.

When it comes to question types, QA systems can be divided into three categories: chat systems, answer-type systems, and task-oriented systems. Chat systems are designed to provide fun and personalized replies, engaging users in casual and often entertaining conversations. Answer-type systems provide responses based on a knowledge base, delivering specific answers to user queries. Task-oriented systems are used to complete specific user tasks by utilizing semantics ability. Task-based and answer-based systems often feature single-round interactions, while chat systems are characterized by unlimited, ongoing replies. Our INFO system is a domain-specific hybrid QA system based on Retrieval-Augmented Generation (RAG) with Large Language Models (LLMs), typically capable of solving users' problems in single-round interactions.

3

Related Work

As we mentioned before, RAG can alleviate the challenges of LLMs that they have out-of-date knowledge and hallucinate content. However, in real-world scenarios, the retrieved text inevitably contains noise. And lead to inaccurate answers in output. To avoid that and improve the RAG target, we need a comprehensive and systematic evaluation and analysis of RAG with LLM.

Several studies have focused on evaluating the performance of RAG systems in various contexts. For instance, ReQA by Ahmad et al. (2019) (28) offers an open-source, end-to-end testbed for evaluating RAG systems' performance on QA tasks, providing valuable insights into how these systems perform in practical applications.

When evaluating RAG pipelines under white box conditions, it is often useful to assess each component's performance separately. For example, eRAG (29) provides a focused evaluation on the retrieval part of the RAG system, allowing for targeted improvements in this area. provides a separate evaluation only on the retrieval part. However, the structure of RAG is not stable and easy to change. So it is more practical to evaluate in a black box way. However, since the structure of RAG systems can be quite dynamic, it is often more practical to conduct evaluations in a black-box manner, assessing the overall system performance rather than individual components.

Detecting hallucinations in LLM-generated content is a significant area of research. Ji et al. (2023) studied methods to detect hallucinations in LLMs (30). Similarly, Zhang et al. (2023) introduced a method using few-shot prompts to predict the faithfulness of generated content (31). However, standard prompt templates often struggle to detect hallucinations effectively. Advanced models like GPT-4 can be used to enhance this process. For instance, SelfCheckGPT (32) by Manakul et al. (2023) evaluates faithfulness by sampling multiple answers, operating on the principle that factual answers are more stable and semantically similar across different samples.

Evaluation of RAG systems can also be approached from different angles. Chen et al. proposed the RGB framework (33), which includes four assessment dimensions, providing a more nuanced evaluation of RAG systems' performance.

Quantitative evaluation methods are often more robust than purely semantic ones. For example, GPT can be directly prompted to assess specific aspects of an answer by providing a score between 0 and 100 or using a 5-star scale. Wang et al. (2023) demonstrated this method, highlighting its sensitivity to prompt design (34). Seungone et al. (2024) addressed this issue by developing customizable prompt templates, enabling comprehensive evaluations across different aspects of RAG systems(35).

Most approaches rely on the availability of one or more reference answers. RAGAS (36) leverages reference answers to calculate correctness, providing a structured way to assess the accuracy of generated answers. And ARES (37) is an automated evaluation framework that incorporates reference answers to offer a detailed analysis of RAG systems' performance.

4

Methodology

We want to evaluate the performance of INFO and design an evaluation pipeline to achieve this goal. We built a benchmark to measure INFO's performance using several quantitative metrics. This chapter outlines the steps involved in creating the benchmark and dataset, implementing the evaluation metrics, and conducting the experiments.

4.1 System Architecture of INFO

Before diving into the evaluation details, it's essential to understand the structure of INFO. INFO leverages a combination of the GPT-3.5 model and a Retrieval-Augmented Generation (RAG) pipeline to answer user queries. INFO's structure is shown in 4.1. The system consists of two main components: the Retriever, which retrieves the most relevant nodes from the knowledge base. And the Generator, which generates answers based on the context provided by the retriever. Additionally, other components also support these main functions. We will introduce each component in detail in the following sections.

- *Pre-processing phase.* Before querying the system, it is necessary to convert the engineering-relevant support documents into a format suitable for efficient retrieval. The first step in the procedure is to divide the supporting documents into "chunks", which are equal-sized sections. To ensure that each chunk has necessary context information, chunks overlap with each other. The overlap helps to keep context information continuous between adjacent chunks.

  Then we embed these chunks into vector representations, and store them in a vector database, enabling quick and efficient similarity searches.

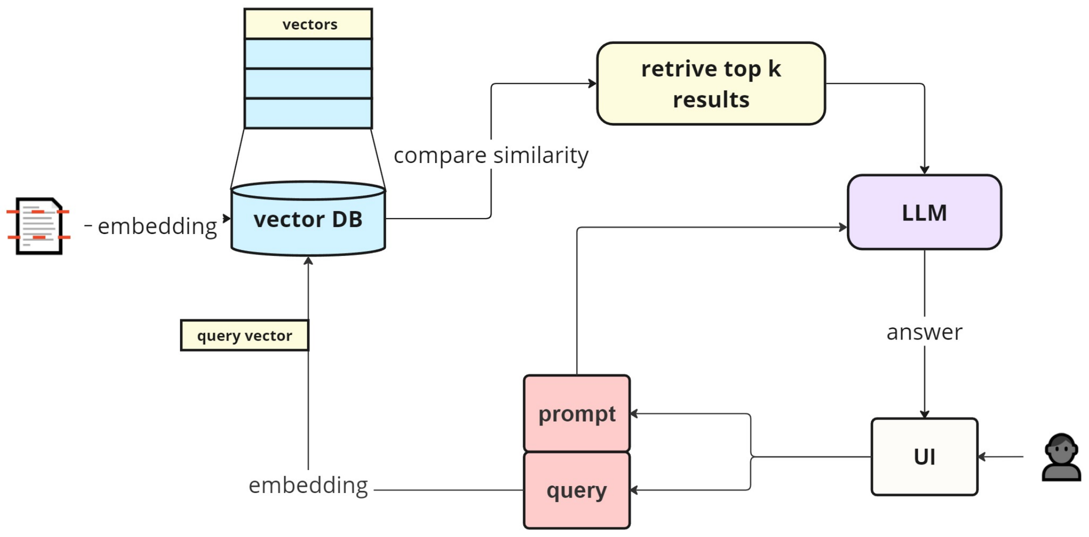


Figure 4.1: INFO structure

- *Retrieval phase.* When using this system, a user should submit a query via the user interface (UI). First, the query is embedded into a vector. The retriever then compares this query vector with all the vectors in the vector database. By calculating the similarity score between vectors, the retriever identifies and retrieves the top-k most relevant document chunks.

- *Generation phase.* The pipeline then feeds the retrieved chunks into LLM along with the user query. To get better performance of LLM, some certain, pre-defined prompt is automatically added to the input. The prompt can include specific instructions such as role setting and format control.

- *Response phase.* With all the mentioned materials, the LLM processes the combined input and generates a final response. This response is then delivered back to the user through the UI.

Table 4.1 are hyperparameters that have been set in advance in the pipeline.

Considering the maximum limits of 4096 tokens of the text input of INFO LLM model GPT-3.5-Turbo (38), we set the chunk size as 1024 tokens to allow the model to process multiple chunks simultaneously. Additionally, we set a 20% overlapping rate which means each chunk shares 20% of its content with the previous one. This overlap ensures that INFO maintains a continuous understanding of the provided text.

Table 4.1: Hyperparameters and their values

Hyperparameter	Value
Chunk size	1024
Overlapping rate	20%
Retrieve top k	3
Embedding model	text-embedding-ada-002
LLM	Microsoft Azure OpenAI GPT 3.5-turbo

For INFO, retrieving more chunks (k) increases the likelihood of capturing the relevant context and obtaining comprehensive information. However, increasing k also consumes more resources. Therefore, to balance performance and efficiency, we set k=3 to retrieve the top 3 most relevant chunks with the highest similarity scores.

Moreover, we choose the text-embedding-ada-002 model as the embedding model. Embedding models are designed to convert text, such as words or paragraphs, into numerical vectors, enabling computers to process and understand natural language. The text-embedding-ada-002 model was published in December 2022. It is particularly efficient, assuming approximately 800 tokens per page and offering 12,500 pages per dollar. It outperforms all old embedding models on multiple tasks such as sentence similarity tasks till December 2023. This makes it the most cost-effective state-of-the-art (SOTA) model available at the time.

Notably, the hyperparameters we selected are preliminary and subject to future optimization. For example, we can increase the number of retrieved chunks (k) to improve the performance of the retrieval phrase. Therefore, we desperately need a robust benchmark to provide quantitative feedback, which will facilitate the evaluation of future adjustments and guide the continuous evolution of INFO. This benchmark will enable us to measure the impact of various parameter changes systematically, ensuring that INFO's performance is optimized for real-world scenarios.

4.2 Dataset Generation

To quantify the performance of INFO, it is essential to build a benchmark, which requires a dataset. However, because INFO is currently in the testing stage, we do not have a real user dialogue history. Thus, we create a tailored dataset derived from specific domain-related documents called Unifi ICT. We need to generate both ground truth QA pairs and

INFO-generated answers for evaluation use. Unifi is the company's internal management system. It is the central nexus for policies, processes, and procedures that define how engineers perform daily work. The formats are a mixture of DOCX and PDF, featuring a blend of textual and tabular content.

It contains 3 different topics: CL, GL and WI. GL stands for guideline, which guides to determine a course of action. Guidelines are supporting documents intended to provide information or commentary on any procedure or process for the completion of a task or an activity identified in procedures or processes. WI means Work Instruction which contains detailed instructions on how to perform a task or an activity. CL means Checklist, which is provided to ensure completeness of a task, activity, or deliverable and is part of the quality control procedure.

In total, there are 58 DOCX files in the source files, and we list 5 example files in the data source in Table 4.2.

Table 4.2: Example names of files

Document Type	Filename	Description
Checklist	ENG-ICT-CL-00001.83_Maintenance	Checklist for routine maintenance inspections
Guideline	ENG-ICT-GL-00001.03_Engineering Procedures	Document outlining standard engineering procedures and practices
Guideline	ENG-ICT-GL-00101.29_Safety Compliance	Document outlining safety standards and compliance protocols
Work Instruction	ENG-ICT-WI-00001.77_Assembly Procedures	Detailed instructions for the assembly of components used
Work Instruction	ENG-ICT-WI-00101.05_Quality Inspection	Instruction for quality inspection of engineering components

Given INFO's typical one-round question-answer (QA) interaction in real scenarios, building question-answer pairs is both efficient and sufficient for evaluating its performance. Generating these QA pairs manually is not feasible due to the extensive time and cost involved, so we use automated generation. GPT-4.0 has notable performance in text generation. It can understand context and generate coherent, contextually accurate responses, making it an ideal model to generate relevant questions and corresponding answers from the source files.

Table 4.3 shows the number of QA pairs derived from each topic of the documents.

Table 4.3: Dataset Compose

Document Type	QA Pair Number
Checklist	105
Guideline	237
Work Instruction	222
Total	564

Therefore, we decided to utilize GPT-4.0 to generate query-answer pairs from the corporate support documents. These pairs, generated by GPT-4.0, serve as the ground truth for evaluation. On the one hand, the ground truth QA pairs are crucial for establishing a reference standard. On the other hand, INFO-generated answers serve as the primary subjects of evaluation. By comparing these responses to the ground truth QA pairs, we can quantitatively assess INFO's performance on different aspects.

1. Input Documents: The corporate support documents, primarily in DOCX format, are input into GPT-4.0.

2. Generate Golden QA Pairs: GPT-4.0 processes these documents to generate several QA pairs relevant to the content of each document. For each document, GPT-4.0 creates queries (questions) based on the information within the document and provides corresponding answers. And these answers are served as ground truth. They are used as the reference standard to evaluate the performance of INFO.

3. Get INFO Answer: The queries are then fed into INFO, and INFO's responses are collected.

The detailed dataset generation pipeline is shown in Figure 4.2.

Combining all the golden QA pairs and answers generated by INFO together, we could get a comprehensive dataset with features prepared for evaluation. The features are shown in Table 4.4.

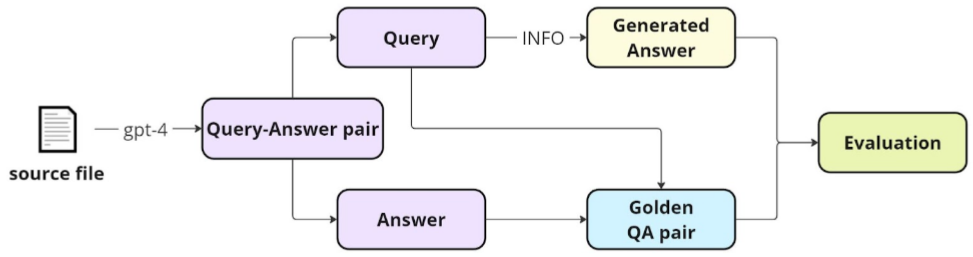


Figure 4.2: Dataset generation pipeline

Table 4.4: Features and Meaning

Column Name	Meaning
query	generated query based on input file
reference$_{c}ontexts$	specific paragraph in input file
reference$_{a}nswer$	golden answer
reference$_{a}nswer_{b}y$	generate golden answer by GPT-4-32k
query$_{b}y$	generate query by GPT-4-32k
query$_{s}ource_{f}ilename$	the input file name
generated$_{a}nswer$	generate answer by InfoWiz
top1$_{c}ontext$	context with top 1 relevancy score
INFO$_{c}ontext_{r}elevancy$	relevancy score of the context used for generating answer

4.3 Evaluation Strategy

Benchmarking is crucial for iterating over RAG systems. Having already created a dataset, the next step is to design suitable metrics to measure the system's performance. As we mentioned in INFO system architecture, for a given question, the system first retrieves some context and then uses it to generate an answer. We focus on particular 3 quality aspects, which are the most important (36). These three aspects can be measured in a fully automated way by prompting an LLM. In our implementation and experiments, all prompts are evaluated using GPT-4.0, which has proven to be a reliable and unbiased evaluator (39).

These three quality aspects form the RAG Triad, which combines three metrics to comprehensively assess a RAG system:

First, *Faithfulness* refers to the consistency of the generated answer with the facts as measured against the given context. This is essential to detect and avoid hallucination, making sure that the retrieved context can serve as a trustworthy resource for the answer. This aspect is crucial in applications where factual consistency is paramount, such as in engineering domains.

Second, *Answer Relevancy* focuses on assessing the relevance of the generated answers to the original query. It ensures that the response can satisfy the user's query, enhancing user experience.

Finally, *Context Relevancy* measures the relevance of the retrieved contexts and calculates the relevancy based on the questions and contexts. It is used to identify how much irrelevant information is in the retried context. This is important due to the cost spent during processing long context in LLMs. Additionally, long passages can reduce the effectiveness of LLMs in utilizing the provided information, particularly for details that are buried in the middle of the passage (40).

Diagrammatically, the RAG Triad (41) looks like Figure 4.3.

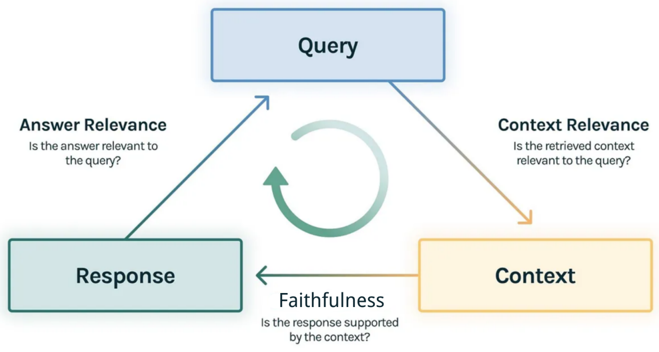

Query
Answer Relevance
Is the answer relevant to the query?
Context Relevance
Is the retrieved context relevant to the query?
Response
Context
Faithfulness
Is the response supported by the context?

Figure 4.3: Evaluation Triad

Moreover, we choose another metric *Correctness* which measures the accuracy of the generated answers compared to the ground truth.

Together, these metrics measure if the RAG system retrieves accurate and relevant information and generates coherent, contextually appropriate, and factually reliable responses. This comprehensive evaluation strategy helps in identifying strengths and areas for improvement in INFO, guiding future iterations and enhancements.

4.4 Evaluation Tool

We automate the measurement of metrics implemented by prompting GPT-4.0. However, it is challenging to find the optimal prompt template for the evaluation.

We need to consider several key aspects to create efficient prompts. First of all, the prompt should contain more reference materials, such as scoring rubrics and examples. To prevent length bias, the prompt should maintain a uniform length among the reference answers for each score. Last but not least, the score should follow a uniform score distribution to prevent decision bias.

To address these challenges, we use llama-index build-in evaluation tools which gain insights from Prometheus (35). Prometheus provides valuable perspectives on the importance of metrics and Chain-of-Thought (CoT) methods. While there are other evaluation frameworks like RAGAS (36), Prometheus prompt templates are designed for detailed and granular evaluation with human-like judgments, making them more intuitive. As we give more attention to the final performance of evaluation than easy implementation, we choose llama-index build-in evaluation tools.

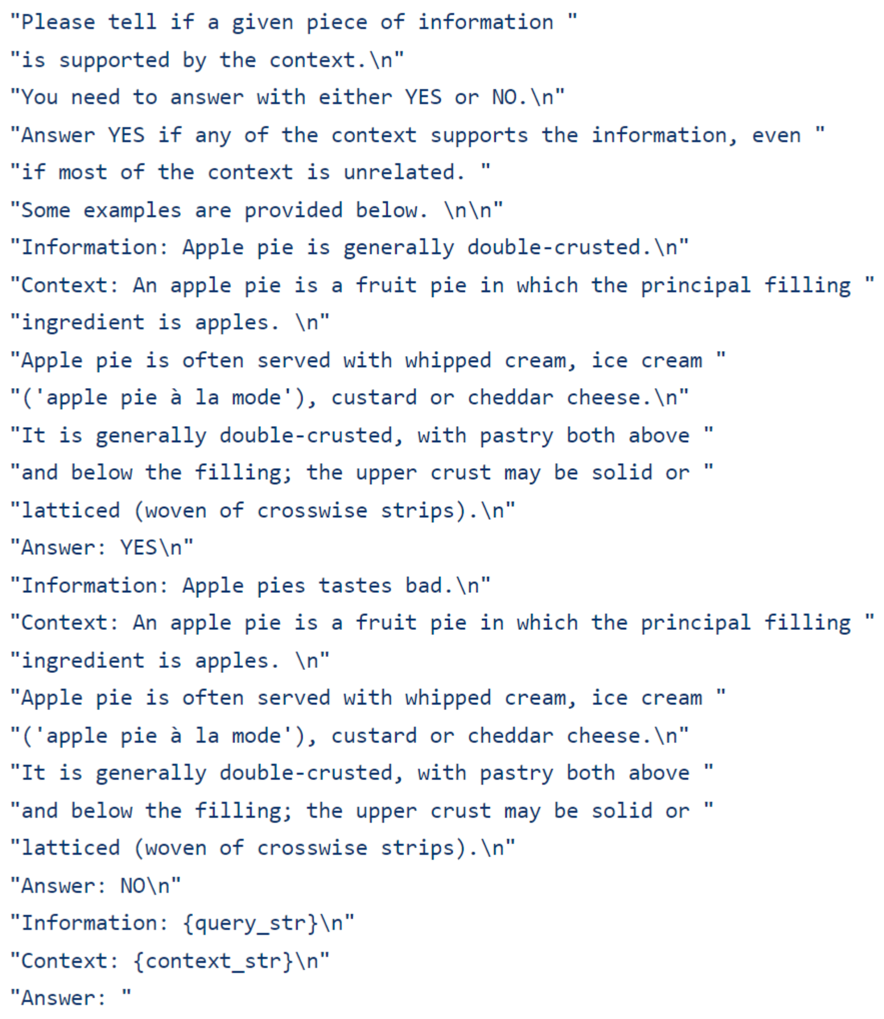
We build each metric evaluation prompt to include four components for the input and two components for the output.

- Input *Instruction*: This component provides the necessary context and directions for the task, including the inputs such as the query or answer. *Evaluation Questions*: These are specific tasks designed to guide the LLM's evaluation process. *Customized Score Rubric*: This rubric includes a description of the evaluation criteria and detailed explanations for each scoring decision, such as binary scores or a scale (e.g., 1-5). *Reference Answer*: Example answers that illustrate the scoring range from lowest to highest, allowing the evaluator to compare the response with the reference to make a scoring decision.

- Output *Feedback*: This provides a rationale for the given score, including a step-by-step Chain-of-Thought (CoT) explanation to enhance the evaluation explainability. *Score*: An integer score for the response, based on the specified criteria.

Faithfulness metrics are a typical example of implementing the template, shown in Figure 4.4.

The first two lines serve as evaluation questions, telling the LLM to determine whether a given piece of information is supported by the provided context.



```
"Please tell if a given piece of information "
"is supported by the context.\n"
"You need to answer with either YES or NO.\n"
"Answer YES if any of the context supports the information, even "
"if most of the context is unrelated. "
"Some examples are provided below. \n\n"
"Information: Apple pie is generally double-crusted.\n"
"Context: An apple pie is a fruit pie in which the principal filling "
"ingredient is apples. \n"
"Apple pie is often served with whipped cream, ice cream "
"('apple pie à la mode'), custard or cheddar cheese.\n"
"It is generally double-crusted, with pastry both above "
"and below the filling; the upper crust may be solid or "
"latticed (woven of crosswise strips).\n"
"Answer: YES\n"
"Information: Apple pies tastes bad.\n"
"Context: An apple pie is a fruit pie in which the principal filling "
"ingredient is apples. \n"
"Apple pie is often served with whipped cream, ice cream "
"('apple pie à la mode'), custard or cheddar cheese.\n"
"It is generally double-crusted, with pastry both above "
"and below the filling; the upper crust may be solid or "
"latticed (woven of crosswise strips).\n"
"Answer: NO\n"
"Information: {query_str}\n"
"Context: {context_str}\n"
"Answer: "
```

Figure 4.4: Faithfulness prompt

The scoring rubric is presented after. Since this is a binary scoring system, the rubric is straightforward, requiring the model to output "YES" if any part of the context supports the information, even if most of the context is unrelated, and "NO" otherwise.

To help LLM understand the task, the prompt includes examples. These examples illustrate how to apply the criteria correctly by providing the concrete materials with "YES" or "NO" answers. For example, the statement "Apple pie is generally double-crusted" is supported by the context and thus correctly answered with "YES." Conversely, the statement "Apple pies taste bad" is not supported by the context, leading to a "NO" answer.

Finally, the prompt template concludes with placeholders for the input query and context. These placeholders are where the actual query and context will be inserted during evaluation.

4.5 Experiment Setup

The experiments were conducted on a Dell laptop with an Intel i7-8650U CPU and 16GB RAM. The development environment included Visual Studio Code (VS Code) and Anaconda for version control. INFO utilized the GPT-3.5-Turbo model from Azure OpenAI API, which has a 4096-token input context window. The embedding model used was text-embedding-ada-002. For the evaluation, we used GPT-4.0-32k, which supports a 32.8K token input context window, also utilizing text-embedding-ada-002.

We implemented Python scripts to automate from dataset generation to metrics evaluation. In addition to the four primary metrics, we tracked the processing time for each step to ensure traceability.

Each query-answer pair was evaluated using the defined metrics. We provide detailed explanations for each metric using the prompt template below.

4.5.1 Answer Relevancy

This metric evaluates how well a response aligns with the user's query. It assesses if the response addresses the subject matter and the specific perspective or focus of the query. Its evaluation questions are as follows:

- Does the provided response match the subject matter of the user's query?
- Does the provided response attempt to address the focus or perspective of the user's query?

Each aspect is scored on a binary scale, contributing to a maximum score of 2 points. This scoring system enables a structured and quantifiable assessment of how relevant and targeted the system's answers are to the initial queries.

4.5.2 Context Relevancy

This metric measures how well the retrieved context from documents matches the user's query and whether it can independently provide a complete answer. Its evaluation questions are as follows:.

- Does the retrieved context match the subject matter of the user's query?
- Can the retrieved context be used exclusively to provide a full answer to the user's query?

Each question is worth 1 point, with a detailed feedback mechanism that guides the evaluator in providing a numeric score and written feedback. Meanwhile, this metric allows for partial marks, so the score scales from 0.125 to 1.

4.5.3 Faithfulness

This metric assesses whether the information provided in a response is supported by the given context, ensuring that the response does not include hallucinated or irrelevant details. Evaluators answer with "YES" or "NO" to indicate whether any part of the context supports the information given in the response.

By utilizing these metrics, we can provide a comprehensive and automated evaluation of INFO's performance.

4.5.4 Correctness

This metric evaluates the accuracy and relevance of the generated answers relative to a reference answer or the factual correctness inferred from context documents.

- Please tell if a given piece of information is supported by the context.

The evaluation process involves a holistic scoring system ranging from 1 to 5. Scores of 1 indicate irrelevance or incorrect information. Intermediate scores (2-3) are given for responses that are relevant but contain factual inaccuracies. Scores of 4 and 5 are reserved for answers that are both relevant and factually accurate, with higher scores indicating greater precision and completeness.

5

Result

5.1 Process Time

In addition to the metrics, we also tracked processing time to reflect the efficiency and scalability of this benchmark.

The query-answer generation phase by GPT-4.0 took 1 hour, 32 minutes, and 1 second. Generating answers by INFO took 41 minutes and 3 seconds. The generation of four metrics involved inputting all 564 QA pairs, with an execution time of 7 hours, 19 minutes, and 40 seconds.

The generation of answers by INFO is relatively fast compared to the overall evaluation process. This suggests that while generating answers is efficient, the evaluation phase requires more computational resources. However, in general, automation still saves a considerable amount of time and money compared to manual efforts.

For instance, generating 564 QA pairs manually would require several human experts, each spending hours reading documents, formulating questions, and crafting accurate answers. Assuming an expert takes an average of 10 minutes to generate a high-quality QA pair, it would take approximately 94 hours of human labor. This does not include the additional time required for quality checks, which could make the whole generation take more than 100 hours.

Additionally, the evaluation phase needs more human resources. Experts would need to assess metrics like correctness and faithfulness of each set. Assuming an evaluation time of 5 minutes per set, this process would require approximately 47 hours. Combined with the generation phase, the total manual effort would amount to around 167 hours, compared to the automated processing time of just around 9 hours.

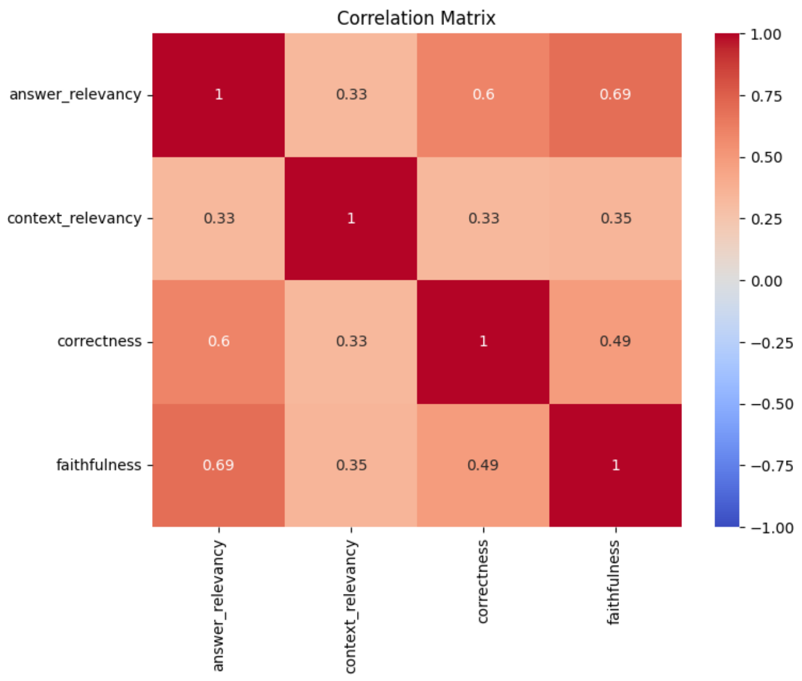


Figure 5.1: Correlation of Metrics

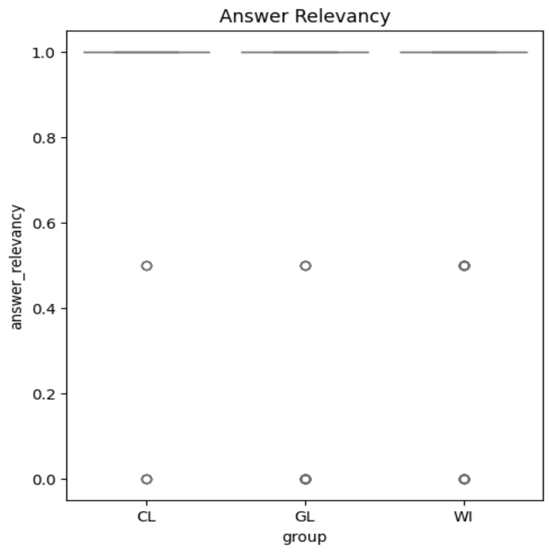
Moreover, human experts are very expensive. If we estimate an hourly rate of €50 for a domain expert, the manual generation and evaluation process would cost around €8,350 (167 hours x €50/hour). In contrast, the automated process would be substantially cheaper, especially considering that the cost of running models like GPT-4.0 is significantly lower, especially considering the scalability of different solutions.

5.2 Metrics Analysis

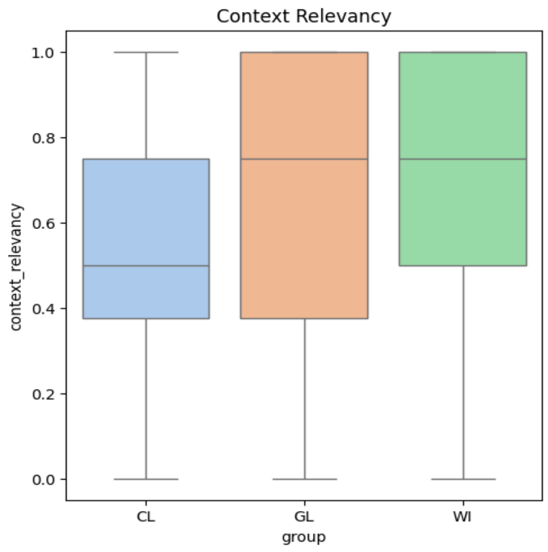
The evaluation of INFO's performance is depicted in the following figures and analyses. We utilized four metrics: Answer Relevancy, Context Relevancy, Correctness, and Faithfulness to measure different aspects of the performance.

Firstly, we checked the correlation between the different metrics to understand their interrelationships. The correlation matrix is shown in Figure 5.1. The scores indicate that the metrics are not highly correlated with each other, implying that they can provide unique insights into different aspects of the performance.

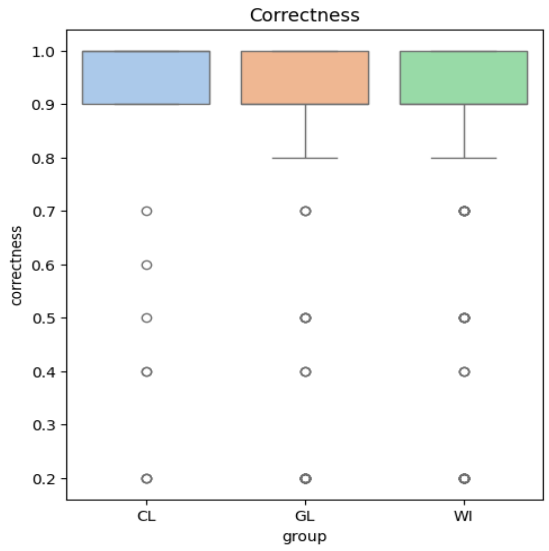
The metrics are representative, and to provide a more detailed view, we visualize the results of each metric using box plots in Figure 5.2. These plots show different metrics across different topics (CL, GL, and WI).



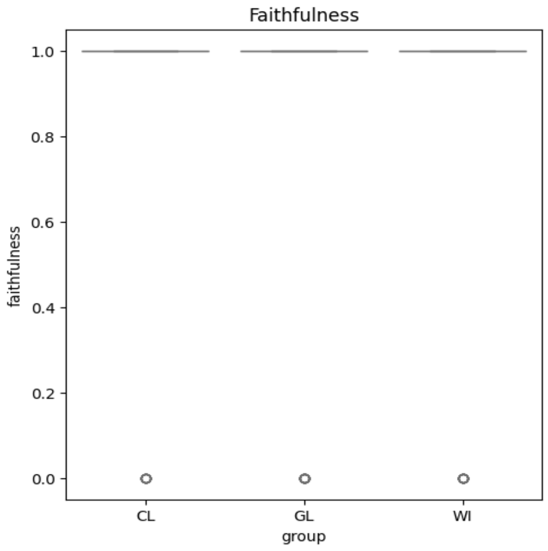
[1]



[2]



[3]



[4]

Figure 5.2: Visualization of Results

- Answer Relevancy: This metric evaluates how well the responses align with queries. As shown, the scores are consistent across the groups with minimal variation. So this indicates that INFO provides uniformly relevant answers regardless of the topic.
- Context Relevancy: This metric assesses the relevance of the retrieved context to the user's queries. The box plot reveals greater variability in scores, particularly in CL topic. This suggests that the retrieval phase may need improvement to capture relevant contexts better.
- Correctness: This metric measures the accuracy of the generated answers. The scores are generally high across all topics, indicating that INFO performs well in generating

factually correct answers.

- Faithfulness: This metric ensures that the answers are grounded in the provided context, avoiding hallucinations. The high scores across all topics indicate that INFO can generate very accurate responses.

5.3 Comprehensive Score

These metrics scores are evaluated individually. However, to gauge overall performance, we need a comprehensive score. In theory, Answer Relevancy, Correctness, and Faithfulness are metrics to evaluate the generation phase, whereas Context Relevancy is for the retrieval phase. If we set the full score as 1, we can split it evenly between the generation and retrieval phases, assigning 0.5 to each. Within the generation phase, we distribute the score evenly among the three metrics. Thus, the full score formula is shown in Equation 5.1.

$$fullscore = \frac{answerrelevancy + correctness + faithfulness}{3} + contextrelevancy \quad (5.1)$$

Using this formula, the total score of INFO is 0.85, indicating a strong performance.

6 Discussion

The INFO evaluation benchmark shows large potential in leveraging AI's ability to automate the process of evaluation. This discussion explores the implications of our findings and the broader applicability of our benchmark.

The benchmark is designed to evolve alongside INFO, allowing it to be reapplied to assess the effects of new features and improvements. By fine-tuning hyperparameters, such as chunk size and overlapping rate, we can optimize context preservation and retrieval accuracy based on specific document structures and query types.

The results of our evaluation have revealed several areas for potential improvement, particularly in the retrieval phase and document parsing:

- Retrieval Phase: The lower scores in Context Relevancy suggest that the retrieval phase needs enhancement. A possible way to improve that is by implementing advanced retrieval techniques, such as rerankers (42) or Hypothetical Document Embeddings (HyDE) (43) or others (44).

  Rerankers: When using vector libraries for approximate semantic searches, results often capture only superficial semantics, prioritizing similarity in character sequences rather than deeper relevance. Rerankers within RAG can further evaluate the relevance, refining results to ensure they are contextually pertinent. This method evaluates the similarity beyond mere character matching, focusing on the contextual relevance of the retrieved documents.

  HyDE is particularly beneficial in this context because it addresses the challenge of zero-shot dense retrieval. It involves generating hypothetical documents that capture the essence of a query, enhancing the retrieval system's ability to find truly relevant information rather than just semantically similar text.

- Document Parsing: The lower scores for the CL topic indicate a need for a better way to solve documents with many tables. Improving parsing methods for documents with complex formats, such as those containing tables, can significantly enhance INFO's ability to process and understand these documents. One potential solution is the implementation of llama-parse, a state-of-the-art tool for parsing tabular data. This enhancement can be quantitatively assessed using the benchmark, providing clear metrics for evaluating improvements.

The benchmark developed for evaluating INFO has broader implications beyond this specific system. It can be generalized to different RAG systems with various datasets, providing a standardized method for performance assessment. This flexibility ensures that the benchmark remains relevant as INFO evolves and new systems are developed.

7

Conclusion

We implemented a benchmark to evaluate INFO's performance and establish a baseline model result, aiding in identifying both strengths and areas for improvement. This benchmark provides valuable insights, particularly highlighting the need for enhancements in the retrieval phase and document parsing methods.

The comprehensive evaluation of INFO using the designed benchmark has yielded significant insights into its performance. The analysis revealed high accuracy and relevance in answer generation while identifying areas requiring enhancement, notably in the retrieval phase. The benchmark's adaptability to other systems and its potential for guiding future improvements underscore its importance as a tool for the ongoing development and optimization of INFO and similar systems. This iterative approach ensures that INFO can continuously evolve, leveraging advanced techniques and fine-tuning parameters to meet the ever-changing demands of its user base.

References

[1] Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. **Factual Error Correction for Abstractive Summarization Models**, 2021. 1, 7

[2] Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. **In ChatGPT We Trust? Measuring and Characterizing the Reliability of ChatGPT**, 2023. 1, 7

[3] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. **Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks**, 2021. 1, 7

[4] Ning Bian, Peilin Liu, Xianpei Han, Hongyu Lin, Yaojie Lu, Ben He, and Le Sun. **A drop of ink makes a million think: The spread of false information in large language models**. *arXiv preprint arXiv:2305.04812*, 2023. 1

[5] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. **Holistic Evaluation of Language Models**, 2023. 2

[6] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. **A Survey of Large Language Models**, 2023. 5

[7] Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. **Class-based n-gram models of natural language**. *Computational linguistics*, **18**(4):467–480, 1992. 5

[8] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. **Recurrent neural network based language model**. **2**, pages 1045–1048, 09 2010. 6

[9] Y. Bengio, Réjean Ducharme, and Pascal Vincent. **A Neural Probabilistic Language Model**. **3**, pages 932–938, 01 2000. 6

[10] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. **Efficient Estimation of Word Representations in Vector Space**, 2013. 6

[11] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. **Deep contextualized word representations**, 2018. 6

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. **BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding**, 2019. 6

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. **Attention Is All You Need**, 2023. 6

[14] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. **Language models are unsupervised multitask learners**. *OpenAI blog*, **1**(8):9, 2019. 6

[15] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. **BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension**, 2019. 6

[16] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. **Scaling Laws for Neural Language Models**, 2020. 6

[17] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. **Training Compute-Optimal Large Language Models**, 2022. 7

[18] Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. **Galactica: A Large Language Model for Science**, 2022. 7

[19] Hangfeng He, Hongming Zhang, and Dan Roth. **Rethinking with Retrieval: Faithful Large Language Model Inference**, 2022. 7

[20] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. **Retrieval-Augmented Generation for Large Language Models: A Survey**, 2024. 7

[21] Jiaxi Cui, Munan Ning, Zongjian Li, Bohua Chen, Yang Yan, Hao Li, Bin Ling, Yonghong Tian, and Li Yuan. **Chatlaw: A Multi-Agent Collaborative Legal Assistant with Knowledge Graph Enhanced Mixture-of-Experts Large Language Model**, 2024. 9

[22] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. **Language Models are Unsupervised Multitask Learners**. 2019. 9

[23] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. **A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications**, 2024. 9

[24] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. **Chain-of-Thought Prompting Elicits Reasoning in Large Language Models**, 2023. 10

[25] Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. **Automatic Chain of Thought Prompting in Large Language Models**, 2022. 10

[26] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. **Self-Consistency Improves Chain of Thought Reasoning in Language Models**, 2023. 10

[27] Neri Van Otten. **What is a question-answering System?** spotintelligence, 2023. `https://spotintelligence.com/2023/01/20/question-answering-qa-system-nlp/#What_is_a_question-answering_System`. 11

[28] Amin Ahmad, Noah Constant, Yinfei Yang, and Daniel Cer. **ReQA: An Evaluation for End-to-End Answer Retrieval Models**. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*. Association for Computational Linguistics, 2019. 15

[29] Alireza Salemi and Hamed Zamani. **Evaluating Retrieval Quality in Retrieval-Augmented Generation**, 2024. 15

[30] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. **Survey of Hallucination in Natural Language Generation**. *ACM Computing Surveys*, **55**(12):1–38, March 2023. 15

[31] Tianhua Zhang, Hongyin Luo, Yung-Sung Chuang, Wei Fang, Luc Gaitskell, Thomas Hartvigsen, Xixin Wu, Danny Fox, Helen Meng, and James Glass. **Interpretable Unified Language Checking**, 2023. 15

[32] Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. **SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models**, 2023. 15

[33] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. **Benchmarking Large Language Models in Retrieval-Augmented Generation**. *Proceedings of the AAAI Conference on Artificial Intelligence*, **38**(16):17754–17762, mar 2024. 16

[34] Jiaan Wang, Yunlong Liang, Fandong Meng, Zengkui Sun, Haoxiang Shi, Zhixu Li, Jinan Xu, Jianfeng Qu, and Jie Zhou. **Is ChatGPT a Good NLG Evaluator? A Preliminary Study**, 2023. 16

[35] Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. **Prometheus: Inducing Fine-grained Evaluation Capability in Language Models**, 2024. 16, 24

[36] Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. **RAGAS: Automated Evaluation of Retrieval Augmented Generation**, 2023. 16, 22, 24

[37] Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. **ARES: An Automated Evaluation Framework for Retrieval-Augmented Generation Systems**, 2024. 16

[38] OpenAI. **GPT-3.5-Turbo**. OpenAI platform, 2023. `https://platform.openai.com/docs/models/gpt-3-5-turbo`. 18

[39] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. **Judging LLM-as-a-judge with MT-Bench and Chatbot Arena**. *arXiv preprint arXiv:2306.05685*, 2023. 22

[40] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. **Lost in the Middle: How Language Models Use Long Contexts**, 2023. 23

[41] TruLens. **The RAG Triad**. TruLens platform, 2023. `https://www.trulens.org/trulens_eval/getting_started/core_concepts/rag_triad/`. 23

[42] James Briggs. **Reranker**. Pinecone, 2023. `https://www.pinecone.io/learn/series/rag/rerankers/`. 33

[43] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. **Precise Zero-Shot Dense Retrieval without Relevance Labels**, 2022. 33

[44] Matouš Eibich, Shivay Nagpal, and Alexander Fred-Ojala. **ARAGOG: Advanced RAG Output Grading**, 2024. 33

Appendix