Mémoire présenté à

La Faculté des Sciences Dhar El Mahraz Fès

Pour l'obtention du Diplôme de Master

## Web Intelligence et Science des Données (WISD)

**Master en double diplomation avec l'Université Sorbonne Paris Nord**

**Spécialité : Informatique**

**Intitulé :**

# Evaluation Framework for Retrieval-Augmented Generation (RAG) Systems: Bridging Information Retrieval and LLM Performances

**Présenté par: Mohamed Amine EL YAGOUBY**

**Encadrants professionnels: Jean-Pierre Chevallet & Eric Gaussier**

**Encadrant académique: Ismail EL BATTEOUI**

**Soutenu le 10/07/2024, devant le jury**

- Ali YAHYAOUY, Faculté des Sciences Dhar El Mehraz, Fès
- Sabri My Abdelouahed, Faculté des Sciences Dhar El Mehraz, Fès
- Dounia El Bourakadi, Faculté des Sciences Dhar El Mehraz, Fès

Année universitaire 2023-2024

# Contents

# List of Figures

# Acknowledgements

I would like to express my deepest gratitude to those who made my internship experience both valuable and memorable.

First, I extend my sincere appreciation to the ANR GUIDANCE project,[1], for the opportunity to contribute to their initiative and for their financial support, which has been pivotal for this internship.

I am particularly grateful to Jean-Pierre Chevallet and Eric Gaussier, my supervisors, for their guidance, encouragement, and invaluable insights throughout the internship. Their expertise and willingness to share knowledge have been crucial to my learning and professional development.

Acknowledgement is also made to academic supervisor Ismail EL BATTEOUI for his constructive feedback, and to Ali YAHYAOUY, the coordinator of the master's program, for providing a supportive academic environment facilitating this internship opportunity.

[1]GUIDANCE project, `https://guidance.anr.isir.upmc.fr`

# Host organization

**Laboratoire d'Informatique de Grenoble (LIG)** [2] is a prominent research institution affiliated with the Université Grenoble Alpes. LIG focuses on various domains of computer science, from theoretical foundations to practical applications. It is a hub for innovation, fostering interdisciplinary collaborations and contributing to advancements in technology and science.

**Modélisation et Recherche d'Information Multimédia (MRIM) Team** [3], is part of LIG, specializes in the modeling and retrieval of multimedia information. This team is renowned in France for its significant contributions to the fields of information retrieval and access. Their research spans several key areas:

- **Multimedia Data Modeling**: Developing models and algorithms for efficient access to large collections of semi-structured, multimedia, and multilingual data.
- **Textual Data Management**: Implementing concept-based indexing for multilingual documents.
- **Contextual Information Access**: Developing embedded systems for mobile environments, focusing on human interfaces and image processing algorithms.
- **System Evaluation**: Regularly evaluating systems and prototypes through user testing and participating in standard information retrieval evaluation campaigns like TREC, INEX, and CLEF.

The MRIM team also extends its research to various applications, including medical data, news data, and scientific library data, and explores interactions between different internet applications through web services.

[2] Laboratoire d'Informatique de Grenoble (LIG), `https://www.liglab.fr/`

[3] Modélisation et Recherche d'Information Multimédia (MRIM) Team, `https://www.liglab.fr/fr/recherche/equipes-recherche/mrim`

# Abstract

Retrieval-Augmented Generation (RAG) addresses key limitations of Large Language Models (LLMs), including knowledge cutoffs, hallucinations, and outdated information. By integrating retrieval mechanisms with generative models, RAG allows LLMs to dynamically access and utilize external knowledge sources, enhancing performance and reliability. The emergence of LLMs and subsequently RAG has significantly boosted research in this field, this survey [3] summarizes over 100 studies on RAG, detailing various methodologies for combining retrieval and generative components to create optimal RAG frameworks for different use cases. However evaluating RAG systems is challenging due to the intertwined contributions of retrieval and generative components. Existing evaluation methods utilize question-answer datasets and document corpora, assessing performance through either computationally expensive LLM-based scoring [2, 7] or ROUGE scores that penalize correct but differently phrased responses [8].

This research introduces a new evaluation framework for RAG systems, addressing the shortcomings of previous approaches. The first phase involves designing a dataset with extracted relevant parts for each example, representing the necessary information to answer a given question, and proposing an evaluation metric for IR systems based on the presence of these parts in the retrieved content. The second phase explores the relationship between IR and overall RAG evaluations and uses this relationship to predicts the overall RAG performance from its Retrieval component performance. This approach eliminates the need for expensive LLM-generated responses and subsequent evaluations, reducing costs and providing a more comprehensive and robust evaluation framework for RAG systems.

# Introduction

## 0.1 Background

**Retrieval-Augmented Generation (RAG) Systems** combine the capabilities of large language models (LLMs) in language understanding and generation with information retrieval (IR) systems to generate more accurate and contextually relevant responses [3]. The typical RAG process involves retrieving relevant documents based on a user query and using these documents to prompt LLMs to generate more informed responses, as shown in Figure 1. State-of-the-art RAG systems have demonstrated proficiency in decreasing hallucination [10], knowledge-grounding from multiple sources [3], and most importantly, the access to dynamic, up-to-date information [3]. Moreover, RAG provides a better solution for personalization in specific domains than fine-tuning of the LLM [3], especially when dealing with limited datasets, As effective fine-tuning typically requires a large and diverse dataset to ensure the LLM adapts well to the specific domain without the risk of over-fitting. RAG systems are versatile and can be utilized beyond question-answering applications. They can potentially enhance any task performed by LLMs that require extensive knowledge, such as generating long-form texts for example, writing essays, or, summarizing information about some personality (ex: Query: "Write a summary about Joe Biden"). For these tasks, techniques like FLARE [4] are particularly effective in improving the precision of specific entities within the generated text, such as dates, names, and numbers. It achieves this by iterative retrieving information for each newly generated sentence. Assuming that if a sentence contains tokens generated with low probabilities, then the LLM is lack of knowledge regarding the specific information within these tokens, and the IR system should be called to retrieve context about this sentence.

**Evaluating RAGs can be challenging** due to the complexity involved in both the retrieval and generation components, as shown in Figure 1. Unlike traditional lan-

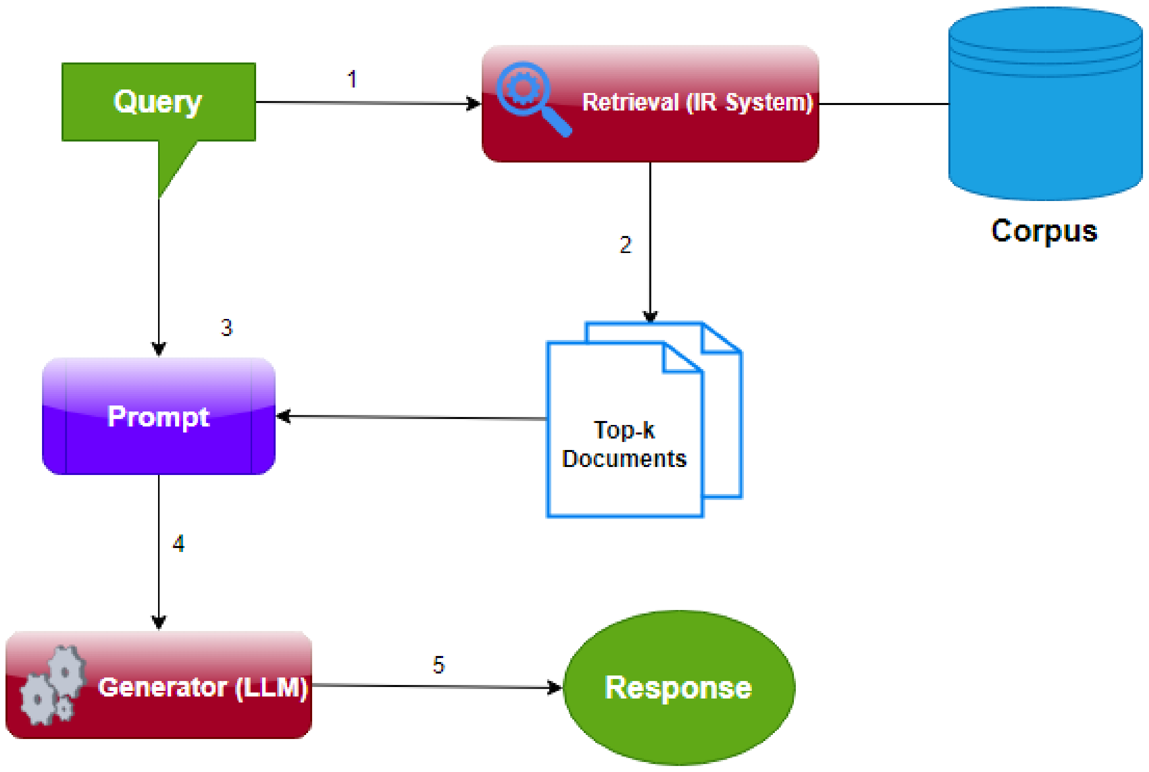


Figure 1: Illustration of the Retrieval Augmented Generation (RAG) basic process

guage models, the performance of a RAG system depends not only on the quality of the generated text but also on the relevance and accuracy of the retrieved documents. Therefore, evaluation metrics must account for both aspects.
IR classical evaluation metrics are precision, recall, and F1 score, these metrics help to assess how well the system retrieves relevant documents in response to a query. Additionally, metrics like Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain (NDCG) are often used to evaluate the ranking quality of the retrieved documents.
Traditional natural language generation (NLG) metrics such as BLEU and ROUGE scores can be used to measure the relevance of the generated text of LLMs. Moreover, human evaluation plays a crucial role in assessing LLMs, providing insights into aspects such as coherence, fluency, and overall quality that automated metrics might miss.
For the overall RAG evaluation, some of these previous metrics are used, such as ROUGE [8], and other techniques based on employing an LLM as a judge to calcu-

late scores that assess the quality of the RAG system on different aspects such as answer faithfulness [7] and context relevance [2].

## 0.2 Objective

After presenting the state-of-the-art of RAG systems and exploring some of its current evaluation frameworks, benchmarks, and datasets in Chapter 1, our main research question is ***how to evaluate the performance of a RAG system?*** This primary question leads to several sub-questions:

1. ***Evaluation of IR: Is it different when used in a RAG?*** Evaluating IR systems involves assessing their ability to retrieve relevant content for a given query. When considering this evaluation in the context of a RAG system, it is important to account for constraints such as the limited input size of LLMs and the chunking method used. These constraints are detailed in section 2.2.

2. ***What is the impact of IR systems in a RAG?*** Evaluating the generated responses that represent the overall performance of a RAG system with multiple IR systems can reveal the influence these systems have on the accuracy and quality of the generated responses.

3. ***Can we estimate the overall performance of a RAG system based on the performance of its IR system?*** Establishing such an estimation can help us create a new evaluation framework for RAG systems. This framework would evaluate the two phases of a RAG system: starting with the Information Retrieval (IR) phase and estimating the Generation phase performance based on the IR performance. By doing so, it provides a comprehensive understanding of how each phase contributes to the final performance, with minimal cost since we do not need expensive LLM response generation and eventual evaluation. This allows for targeted improvements and more transparent assessments.

Addressing these questions will enable us to develop a comprehensible and cost-effective evaluation framework for RAG systems, ensuring that each component's contribution to the overall performance is clearly understood.

## 0.3 Scope

This research focuses on developing a comprehensive evaluation framework for Retrieval-Augmented Generation (RAG) systems, specifically analyzing the impact of Information Retrieval (IR) systems on the overall performance of RAG systems. The scope includes:

1. **Review of Existing Literature**: We will begin by presenting the state-of-the-art in RAG systems, exploring current evaluation frameworks and benchmarks. This review will provide a foundation for understanding the current challenges and limitations in evaluating RAG systems.

2. **Evaluation of IR Systems within RAG and eventual evaluation of the RAG's overall performance (generated responses) using an automatic method detailed in section 2.4**: How to effectively evaluate IR systems within RAG systems. Based on these evaluations:

   (a) Assessing the influence of different IR systems on the accuracy and quality of generated responses.

   (b) Analyzing the relationship between the evaluation of IR systems and the overall performance of RAG systems.

   (c) Investigating whether the performance of a RAG system can be estimated based on the performance of its IR system.

3. **Development of a New Evaluation Framework**: Based on our findings, we aim to develop a new evaluation framework for RAG systems that is more comprehensive and cost-effective. This framework will:

   (a) Evaluate the two main phases of a RAG system: the Information Retrieval phase and the Generation phase.

   (b) Provide a comprehensive understanding of how each phase contributes to the final performance.

   (c) Allow for targeted improvements and more transparent assessments.

4. **Case Studies and Experimental Validation**: We will conduct case studies and experiments to validate the proposed evaluation framework. This will include testing various IR systems within a RAG system using our evaluation framework and comparing the results against their initial overall performance determined by the automatic evaluation 2.4.

This study is limited to specific RAG models currently prevalent in the field, particularly those that use IR systems for retrieving information from a corpus of documents. It does not encompass all possible configurations for retrieving information in RAG systems, such as Knowledge Graphs or retrieving from databases. Additionally, this study does not aim to develop a new RAG system but describes the state of the art of some of these systems in section 1.2.

## 0.4 Importance

The significance of this research lies in the potential to enhance the effectiveness of RAG systems, by relaying on robust evaluation framework. That's aims to improve the accuracy and reliability of generated responses, which is crucial for applications in various domains such as natural language processing, automated customer service, and information synthesis. Understanding the interplay between IR systems and RAG performance can lead to more informed decisions in system design and optimization, ultimately advancing the field of text generation with information retrieval and its practical applications.

# Chapter 1

# Literature Review

## 1.1 Global Overview about State-of-the-Art RAG's Evaluation Methodologies and Their Limitations

The recently updated survey about Retrieval-Augmented Generation [3], which is detailed in Section 1.2, summarizes more than a 100 studies related to RAG systems. This number indicates significant research interest in enhancing these systems, reflecting the importance and potential of improving LLMs by providing real-time knowledge updates, integrating domain-specific information, and enhancing the accuracy and credibility of the generated content. This growing body of research highlights the evolving nature of RAG systems and the continuous advancements being made in this field.

However, current evaluation methods of RAG systems, which are summarized in Table 1.1 and detailed in Section 1.2, rely on datasets consisting of question-answer pairs along with a corpus of documents or passages used by the IR system to retrieve relevant information for the query. These methods assess the RAG system performance in two major approaches. The first approach is by employing an LLM as a judge to calculate different scores, comparing the true answer from the dataset with the RAG's generated answer, or comparing other components such as query-passage pairs for context relevance scores [2] or query-passage-answer pairs for answer faithfulness scores [7]. The second approach calculates a ROUGE score between the two answers [8]. ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a set of metrics used to evaluate text summarization and machine translation quality by comparing the overlap of n-grams, word sequences, and word pairs between the generated text and a reference text.

| **Evaluation Method** | **Corpus** | **Dataset** | **Scoring with LLM as Judge** | **Scoring with ROUGE metrics** | **IR System Performance** | **Overall RAG Performance** |
|---|---|---|---|---|---|---|
| RAGAS | Not used | Not used | Yes | No | Yes | Yes |
| CRAG | Web Pages | Q/A pairs | Yes | No | No | Yes |
| ARES | Wikipedia pages | Synthetic Q/A pairs from passages generated by LLM | Yes | No | Yes | Yes |
| RGB | News articles | Q/A pairs | Yes | Yes | Yes | Yes |
| eRAG | Wikipedia pages | Q/A pairs | No | Yes | Yes | No |

Table 1.1: Summary of State-of-the-Art RAG's Evaluation Methodologies

These evaluation methods have significant drawbacks:

1. For ROUGE Metrics:
   - Sensitivity to Variations: These metrics penalize RAG's generated responses that are correct but phrased differently from the True answer in the dataset. This sensitivity to lexical variations overlooks the diversity of valid expressions and can result in lower scores for otherwise accurate and relevant answers.
   - Lack of Comprehensive Metrics: These Metrics do not adequately assess the multi-dimensional performance of RAG systems. As they focus on linguistic accuracy but do not evaluate other critical aspects such as answer faithfulness, context relevance, and robustness against noise. This gap results in a limited understanding of the overall effectiveness and reliability of RAG systems.
2. For LLM-Based Scoring: Although using LLMs for evaluation represents an advanced approach that addresses the issues of the previous ROUGE scoring methods, it still falls short in providing nuanced judgment akin to human evaluators. and have multiple drawbacks such :
   - Lack of Explainability: LLMs often operate as black boxes, making it difficult to understand the rationale behind their judgments. This lack of transparency can be problematic for evaluating the performance of RAG

systems, as it is important to understand why a particular score was assigned.

- High Cost: The computational resources required for LLM-based evaluation can be expensive, making this approach less feasible for large-scale or frequent evaluations.
- Context Dependency: LLMs' evaluations can be heavily dependent on the context provided, and slight changes in the input can lead to different evaluations. This can result in instability in the scoring process.
- Limited Task-Specific Knowledge: While LLMs are trained on diverse data, they might not have specific domain knowledge required to accurately evaluate specialized RAG systems, leading to less accurate assessments.

## 1.2 Details about State-of-the-Art RAG Systems and Their Evaluation Methodologies

**The survey for Retrieval-Augmented Generation for Large Language Models** [3] emphasizes RAG's potential to significantly enhance LLMs by providing real-time knowledge updates, integrating domain-specific information, and improving the accuracy and credibility of the generated content. It explores the evolution of RAG by summarizing over 100 RAG studies into three main paradigms: Naive RAG, Advanced RAG, and Modular RAG. Each paradigm progressively improves the integration of retrieval, generation, and augmentation techniques. Naive RAG employs a straightforward approach where documents are indexed, retrieved based on semantic similarity, and then used to augment LLM responses. Despite its simplicity, Naive RAG faces challenges such as irrelevant or misaligned retrievals and hallucinations in generated responses. Advanced RAG addresses these issues by optimizing pre-retrieval and post-retrieval processes. Techniques such as query rewriting, query expansion, and metadata attachments are utilized to refine the retrieval process. Also re-ranking retrieved documents and compressing context to ensure only the most relevant information is fed into the LLM. Modular RAG, allows for dynamic integration and replacement of various modules within the RAG framework. It introduces specialized modules like a search module for direct queries across multiple data sources, a memory module for iterative self-enhancement, and a routing module for optimal query processing pathways. Additionally the survey explores iterative and adaptive retrieval processes, enhancing the model's ability to handle complex, multi-step reasoning tasks. The survey highlights key technologies and methodologies used in each stage of the RAG process, for the retrieval stage it includes various indexing strategies, query optimization techniques, embedding models and hierarchical structures to efficiently organize and retrieve relevant data. The generation stage focuses on context curation and fine-tuning of LLMs to ensure that the generated responses are accurate and contextually appropriate, while the augmentation stage incorporates iterative and adaptive retrieval processes to dynamically refine the context provided to the LLM.

The survey also discusses evaluation methods for RAG systems, highlighting the importance of metrics such as context relevance [2], answer faithfulness [2], and robustness against noise [2]. The evaluation frameworks like RAGAS [2] and ARES [7] are used to assess these aspects, ensuring that RAG systems are both effective and reliable.

**The RAGAS (Retrieval Augmented Generation Assessment) framework** [2], provides a method for evaluating RAG systems without relying on reference answers. By focusing on three key dimensions: faithfulness, answer relevance, and context relevance. RAGAS employs another LLM as a judge to assess these dimensions. For faithfulness, it decomposes the generated answer into individual statements and verifies using this judge LLM each statement against the retrieved context, where the faithfulness score is the percentage of the statements that were supported according to the LLM over all statements. Answer relevance is measured by prompting the LLM to generate potential questions based on the answer, then calculating the similarity between these questions and the original question using their embeddings. This helps determine how well the answer addresses the question without redundancy or incompleteness. Context relevance evaluates the focus and necessity of the retrieved context in answering the question. The LLM extracts sentences from the context that are essential for answering the question, and the relevance score is the proportion of necessary sentences within the context. These metrics enable automatic evaluation of RAG systems and facilitating their optimization without relying on pre-existing reference answers.

In contrast, the ARES method [7] has almost similar scoring criteria but takes automated evaluation a step further by incorporating synthetic data generation and fine-tuning LLM judges.

This **Automated Retrieval-Augmented Generation Evaluation System (ARES)** [7] enhances the assessment of RAG systems by automating the evaluation process, unlike traditional methods that require extensive, costly, and time-consuming human annotations. ARES addresses this through a three-stage process: Synthetic Data Generation, Fine-tuning LLM Judges, and Prediction-Powered Inference (PPI). In the Synthetic Data Generation stage, the LLM is prompted to generate synthetic questions and corresponding answers based on given passages. Lightweight LLMs, such as DeBERTa-v3-Large, are then fine-tuned as LLM judges using this synthetic data for three classification tasks:

(a) Context Relevance: The model is fine-tuned to classify query-passage pairs as relevant (positive) or irrelevant (negative). Positive examples include relevant query-passage pairs, while negative examples include irrelevant passages randomly sampled from the same corpus.

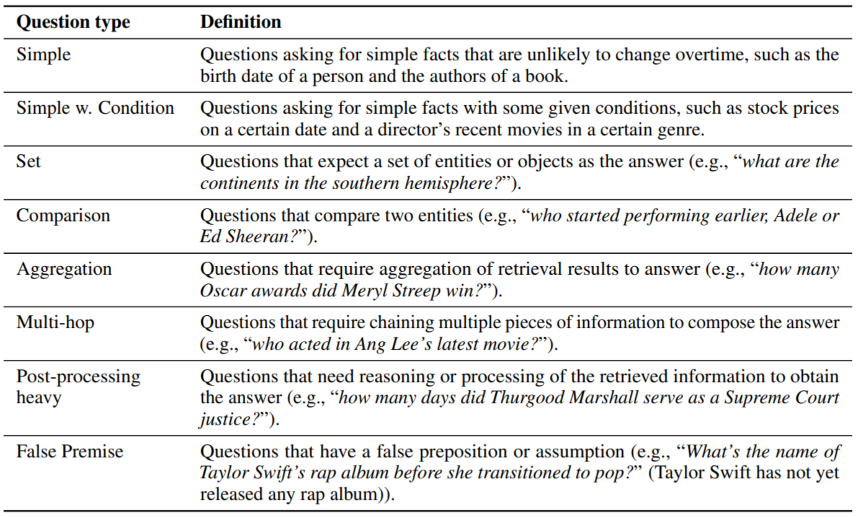
(b) Answer Faithfulness: The LLM is fine-tuned on query-passage-answer triples. Positive examples accurately reflect the passage information, while negative examples contain contradictions or unrelated content. The model checks for consistency and absence of hallucinations, classifying answers as faithful (positive) or not (negative) to the passage.

(c) Answer Relevance: The LLM judge is fine-tuned using query-passage-answer triples, where positive examples are relevant answers derived directly from the passage, and negative examples are irrelevant or do not address the query. The model classifies if the answer directly addresses the query while being grounded in the passage, indicating whether the answer is relevant (positive) or irrelevant (negative).

Prediction-Powered Inference (PPI) enhances ARES evaluation accuracy by integrating human-annotated datapoints (150-300) to validate LLM judges' predictions. PPI learns a rectifier function from this validation set to adjust predictions, improving accuracy on a larger non-annotated dataset. It constructs tighter confidence intervals for evaluation metrics by combining these adjusted predictions with human-annotated data, using a standard 95 % confidence interval. The LLM judges score the RAG system based on these adjusted predictions, and the midpoint of the confidence intervals is used for ranking, enabling precise differentiation and identification of the best-performing configurations.

Various benchmarks Have aroused to evaluate the performance of these systems. Two notable benchmarks in this domain are the Meta Comprehensive RAG Benchmark (CRAG) and the Retrieval-Augmented Generation Benchmark (RGB). CRAG's strength lies in its diverse domains and human-centric evaluations, RGB stands out for its detailed assessment of robustness and information handling capabilities. Here are some details about these benchmarks.

**The Meta Comprehensive RAG Benchmark (CRAG)** [6] [9] by Meta, has a currently available competition that will end on August 26, 2024. It aims to enhance RAG systems by evaluating them across multiple domains and question types. The benchmark's dataset is divided into five domains: Finance, Sports, Music, Movies, and an open-domain encyclopedia. It features eight question types: simple, conditional, set, comparison, aggregation, multi-hop, post-processing, and false premise questions, examples for these question types are shown Figure 1.1.



| Question type | Definition |
|---|---|
| Simple | Questions asking for simple facts that are unlikely to change overtime, such as the birth date of a person and the authors of a book. |
| Simple w. Condition | Questions asking for simple facts with some given conditions, such as stock prices on a certain date and a director's recent movies in a certain genre. |
| Set | Questions that expect a set of entities or objects as the answer (e.g., "*what are the continents in the southern hemisphere?*"). |
| Comparison | Questions that compare two entities (e.g., "*who started performing earlier, Adele or Ed Sheeran?*"). |
| Aggregation | Questions that require aggregation of retrieval results to answer (e.g., "*how many Oscar awards did Meryl Streep win?*"). |
| Multi-hop | Questions that require chaining multiple pieces of information to compose the answer (e.g., "*who acted in Ang Lee's latest movie?*"). |
| Post-processing heavy | Questions that need reasoning or processing of the retrieved information to obtain the answer (e.g., "*how many days did Thurgood Marshall serve as a Supreme Court justice?*"). |
| False Premise | Questions that have a false preposition or assumption (e.g., "*What's the name of Taylor Swift's rap album before she transitioned to pop?*" (Taylor Swift has not yet released any rap album)). |

Figure 1.1: Examples of queries illustrating CRAG benchmark question types taken from the CRAG benchmark paper [9]

This dataset combines 50 pages of web search results and mock Knowledge Graphs (KGs) for each question to simulate real-world retrieval sources. For evaluation, CRAG uses both automated and human evaluations to assess the quality of responses. Responses are scored as perfect, acceptable, missing, or incorrect. The scoring system allocates points as follows: perfect (1 point), acceptable (0.5 points), missing (0 points), and incorrect (-1 point). The benchmark competition employs both automated (auto-eval) and human (human-eval) evaluations. Auto-eval selects the top ten teams for each task by employing rule-based matching and GPT-4 assessment to check answer correctness. It assigns three scores: correct (1 point), missing (0 points), and incorrect (-1 point), and only considers responses that begin within 5 seconds, and limiting them to 50 tokens to promote concise answers. Then complete evaluation of longer responses are done by human-eval to decides the top three teams, where human annotators will give a rating of each response as perfect, acceptable, missing, or incorrect. The overall score is a macro-average across the results, considering the question and its popularity, and the weights of this macro-average are not available to the competing teams.

**The Retrieval-Augmented Generation Benchmark (RGB)** [1] assesses the performance of LLMs on tasks that combine retrieval and generation, focusing on four core abilities: noise robustness, negative rejection, information integration, and counterfactual robustness. RGB uses various metrics to evaluate these abilities. Noise Robustness measures the accuracy of LLMs in extracting correct answers from documents with varying noise levels. This is measured by the Exact Match (EM) score, which compares the model's answers to the correct answers without noise. Information Integration evaluates the LLMs' capability to synthesize information from multiple documents to answer complex questions, again using exact matching for accuracy between the two responses. Negative Rejection assesses whether LLMs can correctly identify when no relevant information is available and refrain from answering, requiring models to output specific rejection phrases. Counterfactual Robustness is evaluated through Error Detection Rate, which measures the LLMs' ability to identify factual errors in documents, requiring specific error detection phrases, and Error Correction Rate, which assesses the proportion of correct answers provided despite incorrect information. ChatGPT further checks for any error detection and rejection beyond exact matching. Each RGB question is supported by up to 30 relevant text chunks from an initial set of 10 web pages, each processed into chunks of up to 300 tokens. The RGB benchmark comprises 1,000 questions, divided into 600 base questions, 200 for information integration, and 200 for counterfactual robustness. These questions are equally split between English and Chinese to ensure a comprehensive evaluation across both languages.

Finally. Evaluating retrievers in RAG systems is crucial to understanding their effectiveness. The eRAG method [8] provides an approach to this evaluation by linking retrieval quality directly with performance on downstream tasks.

**Evaluating Retrievers in RAG Systems (eRAG)**[8] is a recent method for evaluating retrieval quality in RAG systems. It starts with retrieving a ranked list of documents for a given query using a retrieval system. Each document is then individually fed to the LLM along with the query, and the LLM generates an output for each document. These outputs are evaluated using downstream task metrics such as accuracy, exact match, and ROUGE by comparing them against the ground truth labels, resulting in a relevance score for each document. These individual scores are then aggregated using set-based metrics such as Precision, Recall or F1 score, to produce a single evaluation score for the retrieved documents list. This method claims to have a correlation between the retrieval quality and the performance on the downstream task and offers significant computational advantages, reducing runtime and GPU memory consumption as the computational cost of LLMs that are based on the Transformer architecture, scales quadratically with the size of the input.

# Chapter 2

# Framework Design for Evaluating RAG Systems

Our study aims to create a new evaluation framework for RAG systems, starting by evaluating both the IR systems and the overall performance of the RAG system. Additionally, we will explore the relationship between these evaluations and the possibility of estimating the overall performance of a RAG framework based on its IR performance. An illustration of this study is shown in Figure 2.1. Our evaluation framework is presented in Figure 2.2.

## 2.1 Two Main Components of Our Evaluation framework: Dataset and Corpus

To effectively evaluate the performance of IR systems within a RAG framework, it is essential to have a dataset that allows for the assessment of these IR systems in retrieving relevant and necessary information from a corpus to answer a given query. This is because assessing IR systems only based on the final generated responses against the true answers in the dataset may not provide a complete picture of their performance. By focusing solely on the final output, we overlook the critical step of information retrieval, which is foundational to the entire RAG process. Evaluating IR systems on their ability to fetch relevant data ensures that we understand how well these systems can locate and extract the information needed for accurate and effective response generation. Without this, any shortcomings in the retrieval phase could be masked by the generative capabilities, leading to an incomplete and potentially misleading evaluation of the system's overall effectiveness.

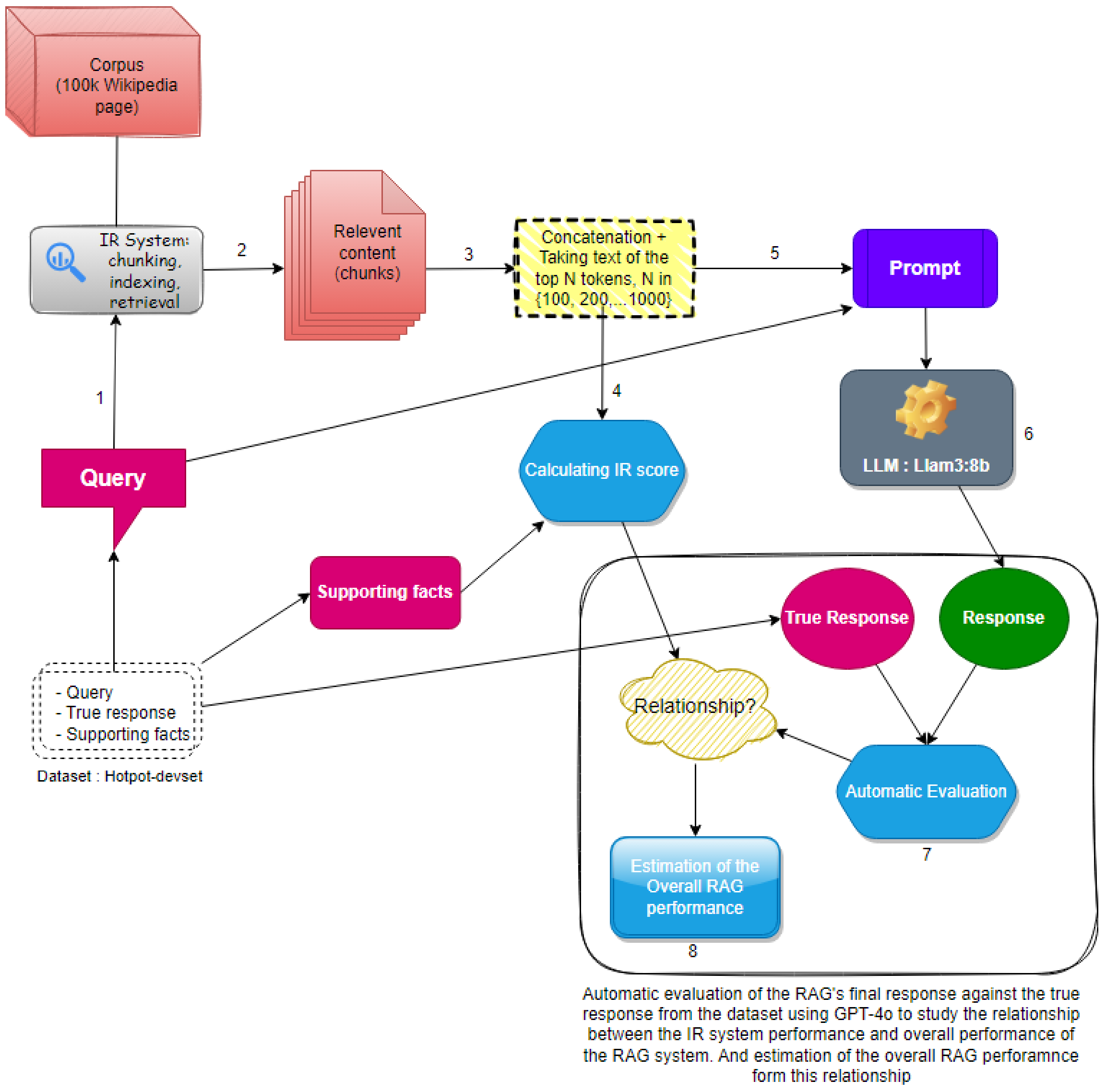

Corpus (100k Wikipedia page)
IR System: chunking, indexing, retrieval
2
Relevent content (chunks)
3
Concatenation + Taking text of the top N tokens, N in {100, 200,...1000}
5
Prompt
1
4
6
LLM : Llam3:8b
Query
Calculating IR score
Supporting facts
True Response
Response
- Query
- True response
- Supporting facts
Dataset : Hotpot-devset
Relationship?
Automatic Evaluation
7
Estimation of the Overall RAG performance
8
Automatic evaluation of the RAG's final response against the true response from the dataset using GPT-4o to study the relationship between the IR system performance and overall performance of the RAG system. And estimation of the overall RAG perforamnce form this relationship

Figure 2.1: Illustration of the study conducted to evaluate Retrieval-Augmented Generation (RAG) systems.

This corpus of documents contains both relevant documents for all the examples in the dataset and non-relevant documents to challenge the IR system's retrieval capabilities.

Our dataset architecture differs from traditional datasets, which typically include

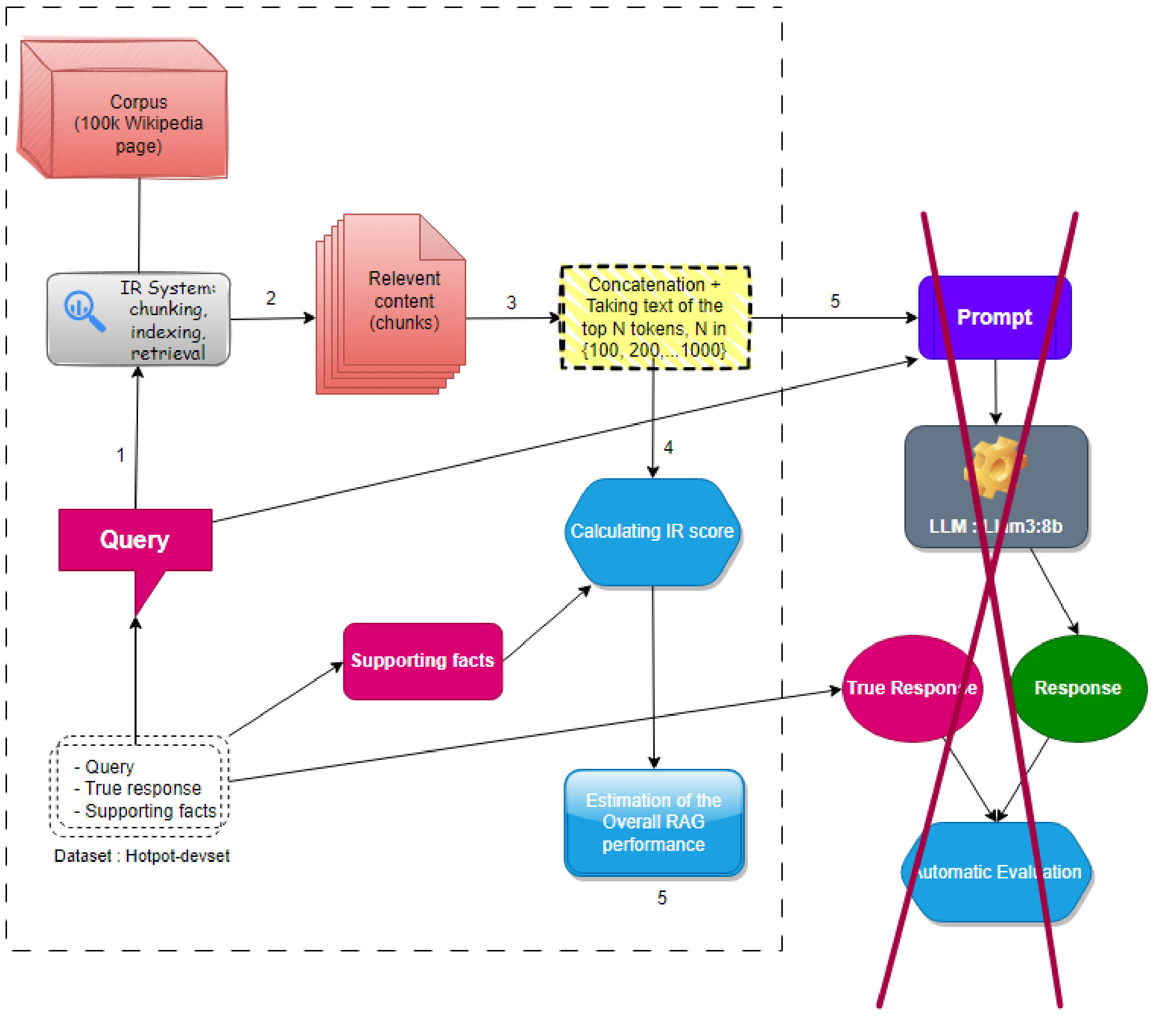
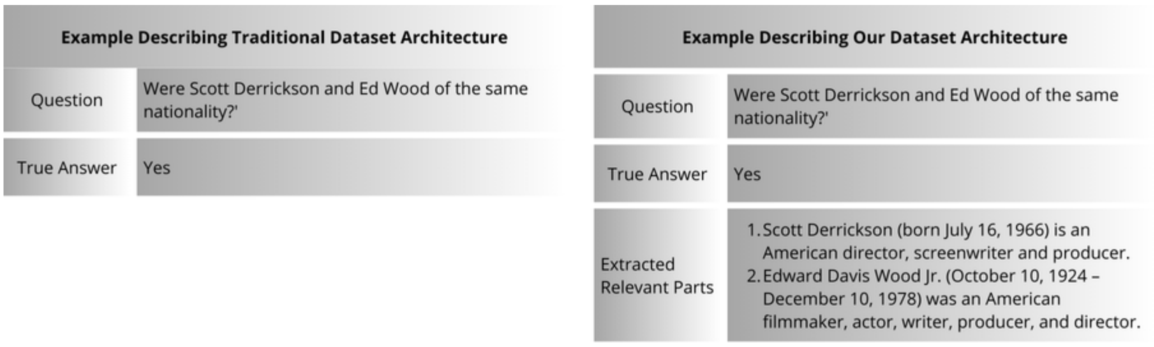

Corpus (100k Wikipedia page)
IR System: chunking, indexing, retrieval
2
Relevent content (chunks)
3
Concatenation + Taking text of the top N tokens, N in {100, 200,... 1000}
5
Prompt
1
4
LLM : Llm3:8b
Query
Calculating IR score
Supporting facts
True Response
Response
- Query
- True response
- Supporting facts
Dataset : Hotpot-devset
Estimation of the Overall RAG performance
5
Automatic Evaluation

Figure 2.2: Illustration of our evaluation framework. It begins by assessing the Information Retrieval (IR) system performance and directly estimates the overall RAG performance from it, eliminating the need for expensive response generation and evaluation.

only question-answer pairs. Instead, our dataset includes, for each example, extracted relevant passages along with the question and answer, as illustrated in the example in Figure 2.3.

In our dataset design, each example is formulated as:

- A question $Q$ and its true answer $A$.



| Example Describing Traditional Dataset Architecture | |
|---|---|
| Question | Were Scott Derrickson and Ed Wood of the same nationality?' |
| True Answer | Yes |

| Example Describing Our Dataset Architecture | |
|---|---|
| Question | Were Scott Derrickson and Ed Wood of the same nationality?' |
| True Answer | Yes |
| Extracted Relevant Parts | 1. Scott Derrickson (born July 16, 1966) is an American director, screenwriter and producer. 2. Edward Davis Wood Jr. (October 10, 1924 – December 10, 1978) was an American filmmaker, actor, writer, producer, and director. |

Figure 2.3: Comparison between Our dataset design againts tradition datasets used for evaluating RAG systems

- A set of extracted relevant passages $P = \{p_1, \ldots, p_n\}$ from documents within the corpus

## 2.2 Constraints about LLMs input size

Before diving into the evaluation method, two constraints regarding the LLM input size should be considered:

1. **The inherent limitation of LLMs regarding the input size**. LLMs, while powerful, can process only a finite number of tokens at a time, which often falls short when dealing with lengthy documents. To overcome this, documents are divided into smaller, manageable chunks. This chunking process ensures that each segment of the document is of an appropriate size that the LLM can handle efficiently, as detailed in Section 2.5.2. This size can vary for each chunk, but it is mandatory to set a maximum size, denoted as $S$, that a chunk can be. Consequently, a corpus of chunks is derived from the corpus of documents. This adaptation led into running the IR system on this corpus of chunks to identify and extract the most relevant chunks from the corpus in response to a query. By retrieving relevant chunks, the most pertinent information is leveraged within the constraints of the LLM's input size.

2. **LLMs prioritize information from the beginning and end of their input sequence while losing crucial context in the middle**. This The well-documented issue, often referred to as "*lost in the middle*", highlights a significant limitation in the processing capabilities of even the most advanced LLMs. Research indicates that LLMs, such as GPT-3.5 and Claude with ex-

tended context windows, tend to perform better when relevant information is located at the beginning or end of the input sequence. This creates a U-shaped performance curve, where the middle portions of the input sequence contribute less effectively to the model's output. This limitation is particularly problematic in scenarios where LLMs are required to handle long contexts. As the length of the input increases, the model's ability to maintain high performance decreases, especially for information situated in the middle of the input. This issue arises because LLMs are inherently designed to focus more on the initial and final tokens due to their architectural biases [5].

## 2.3 Evaluating the IR System in a RAG Framework

Our evaluation method of the IR sytem in a RAG framework considers the two constraints described above in Section 2.2. To address these constraints, the corpus of documents needs to be divided into small chunks. The chunking method of this process is considered a parameter that could vary for each RAG system. Our evaluation method is not sensitive to a specific chunking methodology; the chunking method used in experimentation is described in Section 2.5.2.

Then, we consider only a part of the text corresponding to a fixed number of tokens $N$, denoted $C_N$, from the text concatenation of these retrieved chunks, denoted $C$. This aims to address the previous constraints of LLM input size limits and evaluate different IR systems at the same level.

**Formulation**: The evaluation metric can be formulated as the joint probability of having all the extracted relevant parts, denoted as $P$, contained within the chunks retrieved by the IR system for a question $Q$.

Let $p_i$ represent the $i$-th part from the set of extracted relevant parts $P$ for a given example, and let $C_N$ represent the text corresponding to the first $N$ tokens from the concatenation of the retrieved chunks.

The evaluation metric measures the likelihood that each part $p_i$ is contained within $C_N$. We can express this as the joint probability $\mathbb{P}$ of all parts being contained within $C_N$.

Mathematically, this can be formulated as:

$$\text{Score\_IR} = \mathbb{P}\left(p_1 \in C_N; p_2 \in C_N; \ldots; p_n \in C_N\right) \tag{2.1}$$

In practice, this probability should equals 1 if all the necessary parts in the set $P$

are found in the retrieved chunks and equals 0 otherwise. This should provides a clear and straightforward way to evaluate the performance of the IR system within the RAG framework by quantifying how well it retrieves all necessary parts from the reference documents for each example in the dataset.

Our approach for calculating this probability is by approximating the probability that a part $p_i$ is contained within $C_N$, denoted as $\mathbb{P}(p_i \in C_N)$, by the ratio of the length of the longest common subsequence (LCS) of $p_i$ within $C_N$ to the length of $p_i$. To approximate the joint probability, we take the average of these ratios for all parts $p_i$ in $P$.

$$\text{Score\_IR} = \frac{1}{n} \sum_{i=1}^{n} \frac{\text{len}(\text{LCS}(p_i, C_N))}{\text{len}(p_i)} \tag{2.2}$$

where:

- $\text{LCS}(p_i, C)$ represents the longest common subsequence of $p_i$ within $C_N$.
- $n$ denotes the number of extracted parts in $P$.
- $len(X)$ denotes the length of the string $X$.

Despite the simplicity of this approach, it is effective because it allows us to account for partial matches, thereby providing a more nuanced evaluation of the IR system's performance than a strict binary inclusion measure. By considering partial matches through the LCS, this metric effectively evaluates the IR system's ability to retrieve relevant information. Moreover, this score tend to 1 if all the necessary parts in the set $P$ are found in the retrieved chunks and to 0 otherwise.

## 2.4 Evaluation of the Generator (LLM) with Different IR Systems in a RAG Framework

In this evaluation, the retrieved information for a query is used to prompt the LLM to answer this query. Also the LLM is prompt only with this query so that we can be able to test the impact of adding retrieved information in the prompt. The Python code snippet in Figure 2.4 how this prompting of the LLM (llama3) was done.

The LLM was prompted to answer a query with a one-sentence explanation. This method was used in order to test the LLM's knowledge behind its answer.

For example, given a comparison question such as " *Who is older, Elon Musk or Mark Zuckerberg?*" and the LLM is prompted to generate only a short answer without

RAG Generation

```python
def generate_answer(query , documents=0, llm = ChatOllama(model='llama3:8b', temperature=0.0)):
    if documents :
        prompt = f"""Given the documents below,
        your role is to answer the query using only
        these documents. Respond with a precise,
        one sentence explanation.

        Query:
        {query}

        Documents:
        {documents}

        Answer:"""
    else:
        prompt = f"""Responde with one sentence precise explanation to the given query.
        Query: {query}
        """
    result = llm.invoke(prompt)
    answer = result.content
    return answer
```

Figure 2.4: Generating answers using LLM Llama3 with 8 billion parameters using a given query or using a query and retrieved documents (RAG).

any further explanation, the response might be just "*Elon Musk*", which is correct. However, if it's prompted to respond with a one-sentence explanation, the response might be:

*"Elon Musk is older than Mark Zuckerberg. Elon Musk was born on June 10, 1971, while Mark Zuckerberg was born on May 12, 1984."*

Here, although the response to the comparison is correct, there is hallucination in the details as the correct birth date of *"Elon Musk is June 28, 1971"*, and for *"Mark Zuckerberg is May 14, 1984"*.

this demonstrates that while the LLM can correctly identify who is older, it may still introduce inaccuracies in details (hallucination), which shows its lack of knowledge about these specific details.

**The evaluation of the overall RAG performance** is done by evaluating the LLM responses using an automatic scoring method, where GPT-4-o is prompted to score these LLM responses. The prompt template used for this task is the Figue 2.5.

**Prompt template**

Task: Assess the given candidate's answers based on the true answer, references, and using the scoring criteria. Return only the scores separated by commas.

Question: {question}

True Answer: {true answer}

References: {references}

Candidate's Answers: {candidates answers}

Scoring Criteria:

If the answer says that there is not enough information in documents to answer the question, the score is 1.

If the answer is partially correct but in details you found statements that are incorrect according to the References, the score is 2.

If the answer is partially correct but it doesn't completely answer the question due to lack of information in the documents, the score is 3.

If the answer is fully incorrect, the score is 4.

If the answer is fully correct, the score is 5.

Figure 2.5: The prompt template for scoring answers using GPT-4-o

Given a question, its true answer and its supporting facts from the dataset. and some candidates answers (the LLM answers without or with using different IR systems), GPT-4-o will provide a score for each one of this answers based on this scoring criteria:

- Score 1: Not enough information in documents to answer the question.
- Score 2: Partially correct but contains incorrect statements (Hallucination).
- Score 3: Partially correct but incomplete due to lack of information.
- Score 4: Fully incorrect (Hallucination).
- Score 5: Fully correct

The scoring method used provides a structured way to evaluate the correctness of the LLM's responses. By categorizing responses based on their accuracy and completeness, it becomes clear where the LLM performs well and where it falters. Scores of 2 and 4 highlight areas where the LLM introduces incorrect details and when the response is entirely incorrect, which is crucial for identifying hallucination. Scores of 1 and 3 show the LLM's limitations in generating a comprehensive answer based on available documents but doesn't hallucinate, indicating a need for more robust retrieval mechanisms. A score of 5 indicates a fully correct response, showcasing the system's potential when all components work optimally.

## 2.5 Experimental Setup

### 2.5.1 Data set and corpus collection

**The HotpotQA dataset** created in 2018, is a large-scale question-answering dataset designed to evaluate QA systems with multi-hop reasoning and provide explainable answers. It contains in total 113k question-answer pairs based on Wikipedia articles from the English Wikipedia dump of 2017. The data is collected through crowd sourcing, where workers generate questions that require reasoning over multiple supporting documents, ensuring the necessity for multi-hop reasoning. These questions are divided into two main types: comparison questions that ask about the relationship between two entities and bridge questions that require connecting pieces of information across different texts. An example of a comparison question is the Listing 2.5.1.

Theoretically, these two types of questions can be challenging for IR systems because when information is spread across different documents, for example a comparison question the IR system primarily searches for documents containing the results of the comparison but not the information about each component that is being compared.

Each example in the dataset contains a question, its answer, supporting facts and a context. The context is a mixture of 10 relevant and irrelevant paragraphs designed to provide an answer to the question. Each paragraph includes supporting facts, which are specific sentences necessary to answer the question.

In each example, the context is represented as a list of 10 paragraphs. Each paragraph is formatted as follows: $['\text{document title}', [\text{sen}_1, \text{sen}_2, \ldots, \text{sen}_n]]$. Supporting facts are specified by the document title and the index of the sentence within the paragraph: $['\text{document title}', \text{sentence\_index}]$. An example is illustrated in Figure 2.5.1.

This dataset was used to evaluate QA systems by providing them with the question and the context of 10 paragraphs. The task for the QA systems is to return a response to the question along with the exact supporting facts. The responses and supporting facts are evaluated against those in the dataset using Exact Match, ROUGE, and F1

```
{
  '_id': '5a8b57f25542995d1e6f1371',
  'answer': 'yes',
  'question': 'Were Scott Derrickson and Ed Wood of the same nationality?',
  'supporting_facts': [['Scott Derrickson', 0], ['Ed Wood', 0]],
  'context': [['Ed Wood (film)',
    [...]],
    ['Scott Derrickson',
      ['Scott Derrickson (born July 16, 1966) is an American director,
      ↪  screenwriter and producer.',
        [...]]],
    ['Woodson, Arkansas',
      [...]],
    ['Tyler Bates',
      [...]],
    ['Ed Wood',
      ['Edward Davis Wood Jr. (October 10, 1924 { December 10, 1978) was an
      ↪  American filmmaker, actor, writer, producer, and director.',
        [...]]],
    ['Deliver Us from Evil (2014 film)',
      [...]],
    ['Adam Collis',
      [...]],
    ['Sinister (film)',
      [...]],
    ['Conrad Brooks',
      [...]],
    ['Doctor Strange (2016 film)',
      [...]]],
  'type': 'comparison',
  'level': 'hard'
}
```

Listing 1: HotpotQA Dataset Example with a comparison question.

scores.

**Our dataset** was derived from the development set of the HotpotQA dataset, with the distractor setting[1]. With a total of 7,404 examples, each example contains the question, answer, and the supporting facts represented by their reference document and text, as we don't need all the 10 paragraphs for our task.

Then a **corpus** of 100,000 documents is created from the 2017 Wikipedia dump (the same one used in the HotpotQA dataset). This corpus includes all the documents

[1]HotpotQA dataset, https://hotpotqa.github.io/

used in our dataset plus other irrelevant documents.

### 2.5.2 Chunking details

The chunking method used is from LlamaIndex and called "*Semantic Chunking*", proposed by *Greg Kamradt* in his video tutorial on 5 levels of embedding chunking[2]. Instead of chunking text with a fixed chunk size, the semantic splitter adaptively picks the breakpoints between sentences using embedding similarity. This ensures that a "chunk" contains sentences that are semantically related to each other.

The method works as follows:

1. The text document $D$ is tokenized into individual sentences.
2. A sliding window of three consecutive sentences is created. For each window, an embedding vector is calculated. the next window is by sliding by two sentences to ensure overlapping.
3. The embedding vector of the current window is then compared to the embedding vector of the next window using a similarity metric, such as cosine similarity. The goal is to detect "break points" where the embedding distance between consecutive windows is large. If the similarity between these embeddings falls below a predefined threshold, it is considered a break point, indicating the start of a new semantic chunk.

### 2.5.3 The LLM and IR Systems Evaluated

In our experiment, we used the open-source Llama 3:8b by Meta, a recently developed LLM with 8 billion parameters. This LLM is deployed locally in the server of our Lab, using Ollama[3], a tool for local deployment of LLMs that offers a framework for managing and serving these models on-premises.

For information retrieval, we employed 5 different systems with their indexing strategies. These included:

1. **BM25**: A sparse retrieval method based on TF-IDF that estimates the relevance of documents to a query by considering term frequency, inverse document frequency, and document length normalization. It was employed using PyTerrier[4].

[2]Greg Kamradt, "5 levels of embedding chunking", https://youtu.be/8OJC21T2SL4?t=1933
[3]https://ollama.com/
[4]https://pyterrier.readthedocs.io/en/latest/

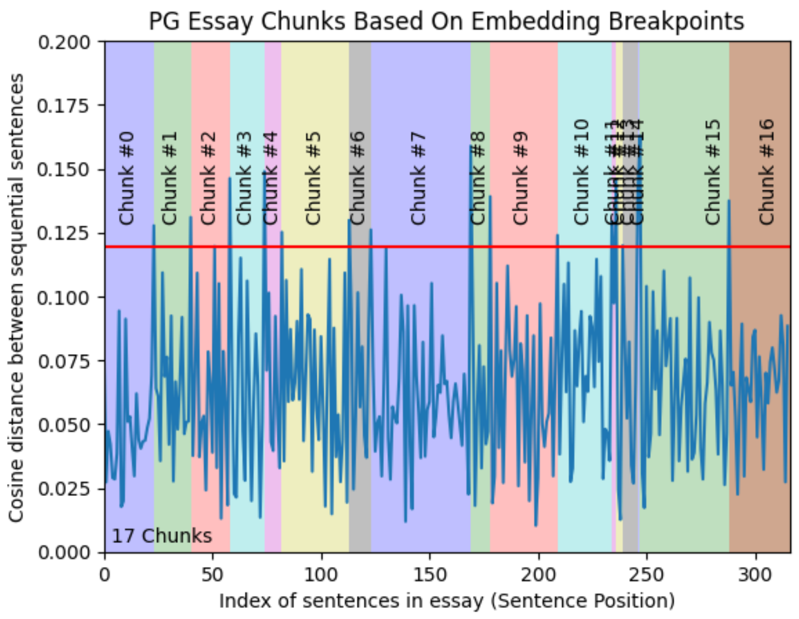


Figure 2.6: Visualization of the chunking of an essay taken from the video tutorial on 5 levels of embedding chunking

2. **Embeddings Similarity Retrieval (SIM)**: A dense retrieval method that aims to capture semantic relationships between words and phrases by estimating relevance between documents and a query as the similarity between there embeddings.

3. **MultiQuery Retriever (MLQ)**: This retriever uses an LLM to generate multiple queries from different perspectives for a given user input query. For each query, it retrieves a set of relevant documents and takes the unique union across all queries to get a larger set of potentially relevant documents.

4. **Maximal Marginal Relevance (MMR)**: A method that aims to balance relevance and novelty, ensuring that the retrieved documents content are not only relevant but also diverse, reducing redundancy.

5. **Long-Context Reorder**: A method that re-order documents retrieved using another retrieval, in our example its Embeddings Similarity Retrieval (SIM) after retrieval to avoid performance degradation.

The last 4 retrievals are implemented in Langchain[5], a framework designed to simplify the development of LLM applications by enabling LLM interactions with various data

[5] https://python.langchain.com/v0.1/docs/modules/data_connection/retrievers/

sources, APIs, and other computational resources to build dynamic language-driven applications. The corpus of chunks for these retrievals is indexed with the FAISS vector store[6] on embedding.

Different RAG methods employ various chunking techniques, resulting in varying chunk sizes. In this experiment, we used a single chunking method, detailed in section 2.5.2. However, the evaluation framework can still accommodate different chunking methods and sizes.

[6] https://faiss.ai/

# Chapter 3

# Results

## 3.1 Results of IR systems Evaluation

Our dataset (with 7,404 examples) was used with the five retrieval systems described above in Section 2.5.3 to retrieve relevant chunks for each query. The retrieved chunks were concatenated into a single text. We evaluated this text by calculating the score_IR for each level of $N$ first tokens of this text, where $N \in \{100, 200, \ldots, 1000\}$ using the formula 2.2.

At each level of $N$ tokens, the average of scores across all the examples in the dataset is calculated. The results are shown in Table 3.1.

| N | bm25 | sim | mmr | mlq | reo |
|---|---|---|---|---|---|
| 100 | **0.40** | 0.34 | 0.34 | 0.29 | 0.16 |
| 200 | **0.46** | 0.39 | 0.38 | 0.33 | 0.19 |
| 300 | **0.49** | 0.41 | 0.39 | 0.35 | 0.20 |
| 400 | **0.52** | 0.44 | 0.41 | 0.37 | 0.21 |
| 500 | **0.55** | 0.46 | 0.42 | 0.39 | 0.22 |
| 600 | **0.57** | 0.47 | 0.43 | 0.40 | 0.23 |
| 700 | **0.58** | 0.48 | 0.44 | 0.41 | 0.24 |
| 800 | **0.60** | 0.50 | 0.44 | 0.42 | 0.25 |
| 900 | **0.61** | 0.51 | 0.45 | 0.43 | 0.26 |
| 1000 | **0.62** | 0.52 | 0.46 | 0.43 | 0.26 |

Table 3.1: Average of IR scores for different retrieval systems across varying levels of $N$

### 3.1.1 Observations

**General Increase in Scores**: Across all retrieval systems, the IR scores generally increase as the number of tokens (N) increases. This indicates that longer passages provide more relevant information, improving the retrieval quality.

For example, BM25 starts at 0.40 for 100 tokens and increases to 0.62 for 1000 tokens. This trend is consistent across all other retrieval systems as well.

**Comparative Performance**: BM25 although its simplicity its consistently outperforms the other retrieval systems across all token lengths, indicating it is the most effective retrieval method for this case among those evaluated.
Sim and MMR perform similarly, with Sim slightly better in most cases.
MLQ shows a steady improvement but lags slightly behind Sim and MMR.
REO consistently has the lowest scores, suggesting it is the least effective retrieval method among those evaluated.

**Rate of Improvement**: The rate of improvement in scores decreases as the token length increases. For instance, BM25's score increases more significantly from 100 to 500 tokens than from 500 to 1000 tokens. This suggests diminishing returns for longer passages, which means that, adding more tokens provides very little to no additional benefit in terms of IR score. This is because the text may become redundant, or the added information might not be as relevant as the initial information.

The diminishing returns observed suggest that further research could focus on enhancing retrieval quality without merely increasing passage length, such as by improving the relevance of retrieved passages or optimizing the concatenation and summarizing processes. Which is important considering the input size limitations the input size limitations of LLMs and the problem of "Lost in the Middle," where important information in the middle of long texts might get less attention. So It's crucial to identify an optimal token length $N$ that balances retrieval quality and practical constraints.

## 3.2 Results of the Generator (LLM) Evaluation with Different IR Systems

### 3.2.1 The Evaluation Cost

Running an automatic evaluation using GPT-4-o as described above in the Section 2.4 on all the dataset examples for all the 5 retrievals and at each $N$ number of tokens, with $N \in \{0, 100, \ldots, 1000\}$, is expensive. We will need 7404 (number of examples) $\times 5$ (retrieval systems) $= 37,020$ API calls, given that in each call 10 responses corresponding to each $N$ number of tokens, for the same query is fed into the GPT-4-o in one prompt. Additionally, to get all the responses with our RAG system using the LLM Llama3 deployed locally, we will need $7404 + 10 \times 7404 \times 5 = 377,604$ iterations.

To reduce costs, we need to specify a fixed $N$ for our evaluations, and to do this, a sample of 100 examples from the dataset was used with BM25 retrieval to analyze the accuracy of the responses and the execution time of our LLM (Llama3) for $N \in \{0, 100, \ldots, 1000\}$. The results are shown in the table 3.2.1

| N | Exec-time (s) | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0.59 | 0 | 19 | 3 | 37 | 41 |
| 100 | 0.77 | 15 | 1 | 24 | 32 | 28 |
| 200 | 0.86 | 12 | 1 | 21 | 31 | 35 |
| 300 | 1.01 | 11 | 0 | 18 | 31 | 40 |
| 400 | 1.03 | 8 | 4 | 17 | 29 | 42 |
| 500 | 1.05 | 12 | 4 | 13 | 32 | 39 |
| 600 | 0.63 | 11 | 2 | 12 | 28 | 47 |
| 700 | 0.65 | 9 | 5 | 10 | 25 | 51 |
| 800 | 0.85 | 10 | 2 | 11 | 28 | 49 |
| 900 | 1.37 | 4 | 5 | 12 | 29 | 50 |
| 1000 | 0.88 | 8 | 1 | 9 | 25 | 57 |

Table 3.2: GPT-4-o evaluation scores of the responses a sample of 100 example from the dataset with the execution time in seconds of our LLM (Llama3) for $N \in \{0, 100, \ldots, 1000\}$.

**The observation** was that, even when $N$ increases, the execution time does not increase proportionally. This can be attributed to the LLM Llama3 employing techniques for handling longer contexts more effectively. These include caching mechanisms, which store intermediate computations such as key-value pairs during the attention mechanism to speed up subsequent token generation, and Grouped-Query Attention (GQA), an efficient representation technique that enhances scalability.

Additionally, as $N$ increases, the number of correct answers with a score of 5 increases, while the number of incorrect answers and hallucinations with scores of 2 and 4 decreases, indicating that the LLM performs better with more contextual information.

As a result, we choose **N=1000** as the optimal number of tokens that provides a balance between a response accuracy which maximum and a manageable execution time.

### 3.2.2 The Evaluation Results

The evaluation results using GPT4-o for Evaluating of the Generator component (LLM) with different IR systems in the RAG framework

of the overall RAG system with only the LLM and with using each one of the 5 retrieval systems within a token limit of N=1000 are shown in the table 3.2.2. This evaluation was conducted across our dataset of 7404 examples, and the percentage of examples with there corresponding scores (from 1 to 5) is presented in this table.

| **Retriever** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| llm only | 0.01 | 0.13 | 0.04 | 0.58 | 0.24 |
| bm25 | 0.08 | 0.05 | 0.13 | 0.31 | 0.42 |
| sim | 0.09 | 0.05 | 0.14 | 0.40 | 0.32 |
| mmr | 0.10 | 0.07 | 0.12 | 0.42 | 0.29 |
| mlq | 0.18 | 0.06 | 0.12 | 0.41 | 0.24 |
| reo | 0.29 | 0.05 | 0.07 | 0.41 | 0.18 |

Table 3.3: Results of the overall RAG system evaluation

The results show that using the LLM without any retrieval system leads to significant issues with hallucinations, as evidenced by high percentages in scores 2 and 4. This underscores the critical role of retrieval systems in enhancing the correctness of information in the generated content.

**Performance Correlation with IR Systems**: The performance of the RAG systems using each of the five IR systems correlates strongly with the individual performance of these IR systems, as evaluated earlier in section 3.1. Their order on the high percentage of fully correct answers (score 5) and low percentage of hallucinations and incorrect answers (scores 2 and 4) matches the order of their earlier evaluations. The bm25 retriever emerges as the most effective among other retrievals, providing accurate and reliable answers with relatively minimal hallucinations.

**LLM Only vs. RAG**: The LLM without retrieval almost always responds to the question, either correctly or incorrectly, and rarely indicates a lack of information in the response (only 1% + 4% for scores 1 and 3). This results in high rates of hallucinations (58% + 13% for scores 4 and 2). Even with lower-performing IR systems like reo, the LLM in this RAG setup can indicate insufficient information (29% + 7% for scores 1 and 3), showing that it is not capable of finding relevant information to answer some queries, and avoiding hallucinations better than the LLM alone.

## 3.3 Estimating the Overall Performance of the RAG System from its IR System Performance

As shown in the previous Section 3.2.2, the RAG system performance is highly correlated with the performance of the IR system used withing it. This lead us to ask the question: *Can the overall performance of the RAG systems be estimated from its IR system performance?* Achieving this would eliminate the need for expensive LLM generation of responses and the subsequent evaluation of these responses.

**Observations**: As we can see in the figure 3.1, answers with high IR scores tend to have the highest accuracy (llm score = 5). Conversely, answers with low IR scores often reflect a lack of information (llm score = 1). And responses with intermediate IR scores frequently contain hallucinations or incomplete information (llm scores = 2, 3, 4). This suggests that when the IR system retrieves only partial information, the RAG system is more likely to produce hallucinated content.

**Estimation Methodology:** Estimating two thresholds, $h$ and $k$, where:

- An IR score under $h$ indicates a high likelihood of the content producing a response that reflects a lack of information (LLM score of 1).
- An IR score above $k$ suggests a high likelihood of generating a fully correct response (LLM score of 5).
- An IR score between $h$ and $k$ suggests a high likelihood of generating partially or fully incorrect responses with hallucination (LLM score of 2,3,4).

We applied the Maximum Likelihood Estimation (MLE) method. The estimation process involves maximizing the likelihood function for each threshold $h$ and $k$. The formulation to estimate $k$ is as follows:

$$\mathcal{L}(k) = -\sum \log\left(p_{\text{ir}} \cdot p_{\text{llm}} + (1 - p_{\text{ir}}) \cdot (1 - p_{\text{llm}}) + 1 \times 10^{-10}\right), \tag{3.1}$$

where $p_{\text{ir}}$ is

$$p_{\text{ir}} = \begin{cases} 1 & \text{if score}_{\text{ir}} > k \\ 0 & \text{if score}_{\text{ir}} \leq k \end{cases} \tag{3.2}$$

and $p_{\text{llm}}$ is

$$p_{\text{llm}} = \begin{cases} 1 & \text{if score}_{\text{llm}} = 5 \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

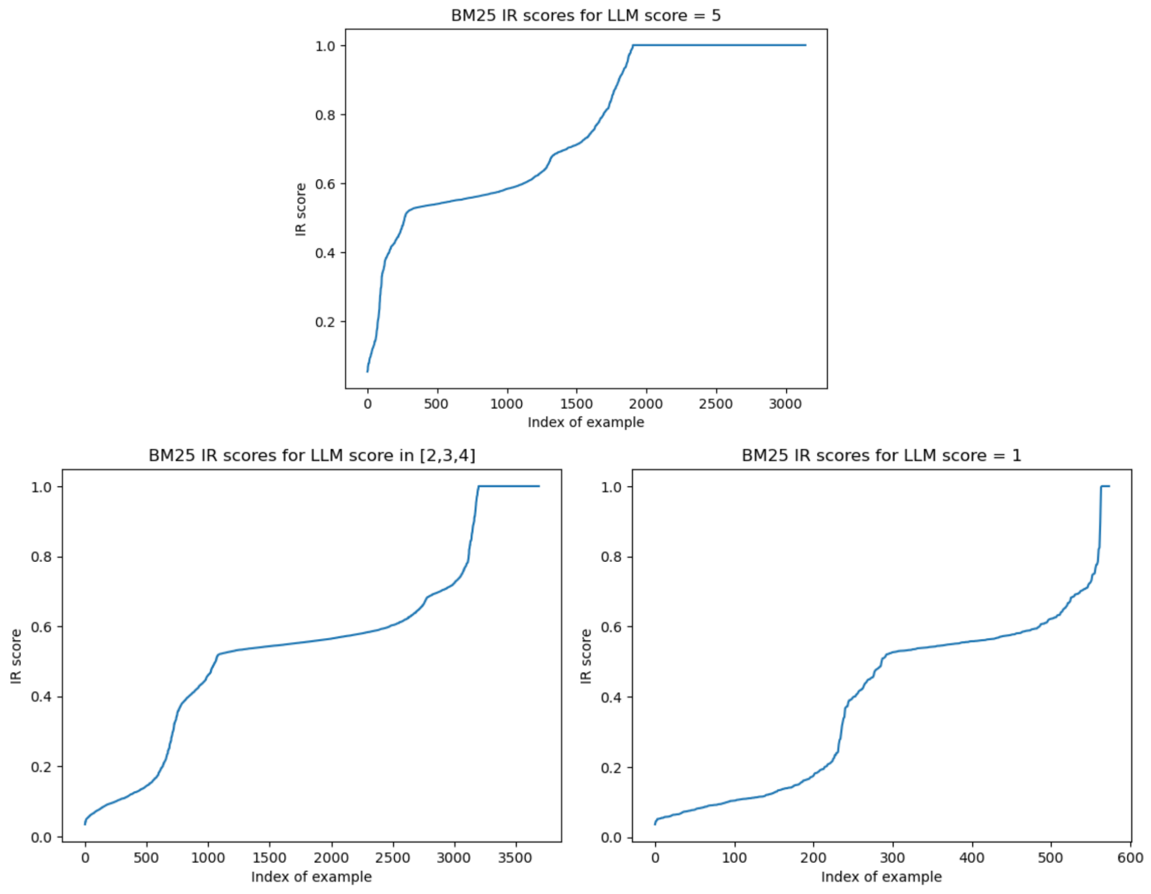


Figure 3.1: BM25 IR Scores corresponding to LLM scores

The likelihood function $\mathcal{L}(k)$ represents the probability of observing a given IR score above $k$ and an LLM score of 5, using data from the automatic evaluation of LLM answers. Given 5 retrievals, this results in $7404 \times 5 = 37020$ (IR score, LLM score) data points. The goal is to find the parameter $k$ that maximizes this likelihood function.

The small constant $1 \times 10^{-10}$ is added to avoid taking the log of zero.

$h$ is estimated using the same method where:

$$p_{\text{ir}} = \begin{cases} 1 & \text{if score}_{\text{ir}} < h \\ 0 & \text{if score}_{\text{ir}} \geq h \end{cases} \tag{3.4}$$

$$p_{\text{llm}} = \begin{cases} 1 & \text{if score}_{\text{llm}} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

**Results**: We found that $\mathbf{h = 0.105}$ and $\mathbf{k = 0.670}$ are the optimal values that maximize the likelihood function $\mathcal{L}(k)$. we apply this threshold on our five retrieval IR scores and the results are illustrated in the figure 3.2.

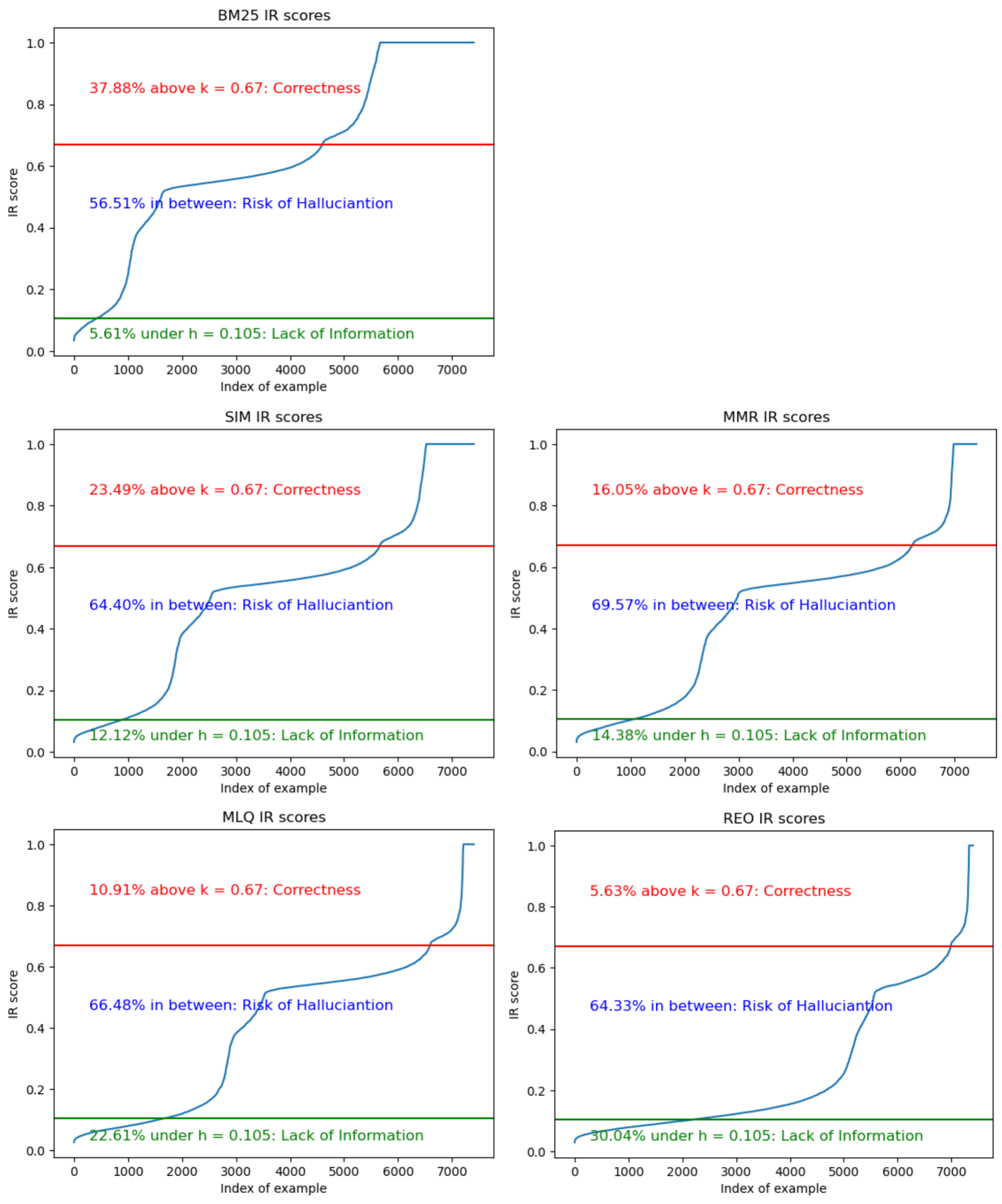


Figure 3.2: Illustration of the estimation of the RAG overall performance from its IR system performance

# Conclusion

This study presents and evaluates a comprehensive framework for assessing the performance of Retrieval-Augmented Generation (RAG) systems, with a particular focus on the influence of Information Retrieval (IR) systems on the overall performance.

Our proposed evaluation framework decomposes performance assessment into two distinct phases: the Information Retrieval phase and the Generation phase. This bifurcated approach enables a detailed analysis of how IR systems affect the quality and accuracy of the final generated responses. Extensive experiments reveal that different IR systems vary in effectiveness, with BM25 consistently outperforming others in retrieval quality across various token lengths. We emphasize the importance of balancing retrieval quality with the practical constraints imposed by the input size limitations of large language models (LLMs). This study concludes by providing an estimation method for predicting overall RAG performance based on the performance of its IR component.

## 3.3.1 Limitations

Despite the robustness of our evaluation framework, there are limitations that must be acknowledged. First, our study focused on specific RAG models and IR systems, which may limit the generalizability of our findings to other configurations of RAG systems. Additionally the sensitivity of the evaluation metric for IR systems to lexical variations. The retrieved content by the IR system may come from documents other than those referenced in the dataset and may contain correct information to answer the question but phrased differently. This limitation is particularly pronounced when working with a real-world, very large corpus like a Wikipedia dump, which contains numerous documents on similar topics. This issue can lead to the underestimation of an IR system's performance when the correct information is retrieved but not recognized due to different phrasing.

### 3.3.2 Future Research

Future research directions include extending our evaluation framework to encompass a broader range of RAG configurations, including those utilizing alternative retrieval mechanisms such as Knowledge Graphs or database queries. Additionally, addressing the sensitivity of evaluation metrics to lexical variations is crucial. Developing more sophisticated evaluation methods that can recognize semantically equivalent information, regardless of phrasing, will improve the accuracy of IR system assessments. Techniques such as advanced natural language understanding and semantic matching could be employed to mitigate this issue.

By addressing these areas, our research aims to contribute to the development of more robust, transparent, and effective evaluation methods for RAG systems, ultimately advancing the field of text generation with information retrieval and its practical applications.

# Bibliography

[1] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762, 2024.

[2] Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*, 2023.

[3] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[4] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*, 2023.

[5] 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. corr abs/2307.03172 (2023), 2023.

[6] Meta. Meta comprehensive rag benchmark: Kdd cup 2024, 2024. Accessed: 2024-05-23.

[7] Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. Ares: An automated evaluation framework for retrieval-augmented generation systems. *arXiv preprint arXiv:2311.09476*, 2023.

[8] Alireza Salemi and Hamed Zamani. Evaluating retrieval quality in retrieval-augmented generation. *arXiv preprint arXiv:2404.13781*, 2024.

[9] Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, et al. Crag–comprehensive rag benchmark. *arXiv preprint arXiv:2406.04744*, 2024.

[10] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023.

# Appendix

All datasets and code used for the experiments detailed in this resarch are available in our GitHub repository, we aim to facilitate replication of our experiments and further research in the field. You can access the repository at `https://github.com/amine-elyagouby/RAG-Evaluation-Framework`