
Degree Project in Computer Science and Engineering

Second cycle, 30 credits

# Building a Reliable Retrieval-Augmented Generation (RAG) System with Hallucination Awareness

Designing and Evaluating a Robust Pipeline for Grounded Answer Generation and Confidence-Aware Retrieval

**LORETA PAJAZITI**

# Building a Reliable Retrieval-Augmented Generation (RAG) System with Hallucination Awareness

## Designing and Evaluating a Robust Pipeline for Grounded Answer Generation and Confidence-Aware Retrieval

LORETA PAJAZITI

Master's Programme, ICT Innovation, 120 credits
Date: June 12, 2026

Supervisors: Shirin Tahmasebi, Jonas Tillström, Jonas Lindberg
Examiner: Amir H. Payberah
School of Electrical Engineering and Computer Science
Host company: Sigma Technology Insight Solutions
Swedish title: Utveckling av ett tillförlitligt retrieval-augmented generation-system med hallucinationsmedvetenhet
Swedish subtitle: Utformning och utvärdering av en robust pipeline för förankrad svarsgenerering och konfidensmedveten retrieval

# Abstract

Retrieval-augmented generation has become an important approach for improving the factual grounding of large language model outputs by conditioning generation on externally retrieved documents. However, retrieval alone does not guarantee reliable answers. A system may retrieve irrelevant or incomplete evidence, generate unsupported claims, or fail to communicate uncertainty when the available knowledge is insufficient. This thesis investigates how RAG systems can be designed and evaluated with a focus on reliability and hallucination-aware behaviour in a fixed-size knowledge base.

The work follows a system-building and evaluation approach. A modular RAG prototype was developed to support controlled experimentation across different pipeline configurations. The system was evaluated on a research-paper corpus using a custom query dataset containing answerable, ambiguous, and unanswerable questions. Three main retrieval configurations were compared: baseline dense retrieval, query expansion with reranking, and dynamic retrieval based on reranker score gaps. The experiments also compared three embedding models and multiple retrieval depths. The evaluation used retrieval-side metrics, including recall, hit rate, and mean reciprocal rank, together with generation-side metrics such as answer correctness and faithfulness.

The results show that retrieval-side design choices substantially affect system reliability. The choice of embedding model had a strong impact on retrieval quality, while query expansion improved evidence coverage and ranking, especially under smaller retrieval depths. Dynamic retrieval provided a strong trade-off between retrieval quality and context size by selecting a variable number of chunks while maintaining competitive performance. However, the experiments also showed that stronger retrieval metrics did not always lead to proportional improvements in answer correctness or faithfulness. Qualitative error analysis further showed that reliability failures included not only unsupported answers, but also unnecessary refusals and insufficiently calibrated responses to ambiguous questions.

The thesis concludes that hallucination-aware RAG should be treated as a pipeline-level problem. Reliable behaviour requires not only retrieving relevant evidence, but also assessing whether the evidence is sufficient for the specific question and communicating uncertainty when appropriate. The developed prototype and evaluation provide a practical foundation for further work on adaptive, transparent, and trustworthy RAG systems.

## Keywords

Retrieval-Augmented Generation, Large Language Models, Hallucination Mitigation, Information Retrieval, Semantic Search, Retrieval Confidence Estimation, Natural Language Processing, Grounded Text Generation

# Sammanfattning

Retrieval-augmented generation har blivit en viktig metod för att förbättra faktagrundningen i svar från stora språkmodeller genom att låta modellen generera svar baserat på externt hämtade dokument. Samtidigt garanterar inte själva hämtningen att svaren blir tillförlitliga. Ett system kan hämta irrelevant eller ofullständig information, generera påståenden som inte stöds av källmaterialet, eller misslyckas med att kommunicera osäkerhet när kunskapsbasen inte innehåller tillräckligt stöd. Denna avhandling undersöker hur ett RAG-system kan utformas och utvärderas med fokus på tillförlitlighet och hallucinationsmedvetet beteende inom en avgränsad kunskapsbas.

Arbetet följer en systembyggande och experimentell ansats. En modulär prototyp utvecklades från grunden för att möjliggöra kontrollerade experiment med olika konfigurationer av hämtningskedjan. Systemet utvärderades på en korpus bestående av forskningsartiklar och med hjälp av ett egenkonstruerat frågedataset med besvarbara, tvetydiga och obesvarbara frågor. Tre huvudsakliga konfigurationer jämfördes: grundläggande tät hämtning, frågeexpansion med omrangering samt dynamisk hämtning baserad på poängskillnader från en omrangeringmodell. Experimenten jämförde även tre inbäddningsmodeller och flera hämtningsdjup. Utvärderingen använde både hämtningsmått, såsom recall, träffgrad och mean reciprocal rank, och genereringsmått, såsom svarskorrekthet och faithfulness.

Resultaten visar att designval i hämtningsdelen har stor betydelse för systemets tillförlitlighet. Valet av inbäddningsmodell påverkade hämtningskvaliteten tydligt, medan frågeexpansion förbättrade både täckning och rangordning av relevant evidens. Dynamisk hämtning gav en god balans mellan hämtningskvalitet och mängden kontext genom att välja ett varierande antal textstycken för olika frågor. Samtidigt visade experimenten att bättre hämtningsmått inte alltid ledde till motsvarande förbättringar i svarskorrekthet eller grundning. Den kvalitativa felanalysen visade även att tillförlitlighetsproblem inte enbart bestod av ostödda svar, utan också av onödiga avslag och otillräckligt kalibrerade svar på tvetydiga frågor.

Avhandlingen drar slutsatsen att hallucinationsmedveten RAG bör ses som ett problem som omfattar hela systemkedjan. Tillförlitliga svar kräver inte bara att relevant information hämtas, utan också att systemet bedömer om evidensen är tillräcklig för den specifika frågan och kommunicerar osäkerhet när så behövs. Den utvecklade prototypen och utvärderingen ger därmed en praktisk grund för fortsatt arbete med adaptiva, transparenta och mer tillförlitliga RAG-system.

## Nyckelord

Retrieval-Augmented Generation, stora språkmodeller, hallucinationsreducering, informationssökning, semantisk sökning, skattning av hämtningssäkerhet, språkteknologi, grundad textgenerering

# Acknowledgments

I would like to express my sincere gratitude to Prof. Amir H. Payberah for serving as the examiner of this thesis and for contributing to the academic evaluation of my work. I am deeply grateful to my supervisor at KTH, Shirin Tahmasebi, for the valuable guidance, feedback, and support throughout this project. Their help in defining the scope of the thesis and shaping the experimental direction was essential to the development of this work.

I would also like to thank Robert Åberg for making it possible for me to conduct this thesis at Sigma Technology Insight Solutions. I am very grateful for the opportunity to carry out this work in a company setting and to connect the academic goals of the thesis with a practical industrial context.

My sincere appreciation also goes to my supervisors at Sigma, Jonas Tillström and Jonas Lindberg. From the very beginning, they were involved in defining this thesis and have followed my journey with continuous engagement, patience, and encouragement. Their guidance, support, and thoughtful feedback have been invaluable, and I am truly grateful for the time and effort they invested in this project.

Pursuing double-degree studies, first at Aalto University in Finland and then at KTH Royal Institute of Technology in Sweden, has been a life-changing journey. It has been a period of academic growth, personal development, and many meaningful experiences. Throughout this journey, I have come to appreciate how important it is to have supportive people around me. I would therefore like to thank my friends for their companionship, dedication, and for helping make this journey more meaningful.

Most importantly, I would like to thank my family for their constant support, motivation, and encouragement throughout this long journey. Regardless of the distance, they were always only one call away, and have been a constant source of strength throughout my studies. I dedicate this achievement especially to my parents, who have encouraged me from the very beginning and stood by me through every step of the way. Their belief and unconditional support have meant more to me than words could ever express.

Stockholm, June 2026
Loreta Pajaziti

## AI Use Declaration

During the preparation of this thesis, AI-based tools were used as supportive aids for language refinement, structuring, brainstorming, and improving the clarity and coherence of academic text. These tools were not used as a substitute for independent research, critical analysis, system design, implementation, experimentation, or interpretation of results. All substantive academic decisions, including the selection of literature, formulation of arguments, design of the Retrieval-Augmented Generation system, evaluation methodology, and final conclusions, were made by the author.

AI-based tools were also used to assist in generating the evaluation query dataset used in the experimental part of this thesis. This use was carried out under strict conditions: the generated queries were constrained by the research objectives, dataset requirements, and evaluation design defined by the author. After generation, all queries were manually reviewed, verified, and, where necessary, revised to ensure their relevance, correctness, and suitability for the evaluation.

The responsibility for verifying claims, ensuring accurate citation use, validating generated evaluation material, and maintaining academic integrity remains entirely with the author. All AI-generated suggestions and outputs were critically reviewed, revised, and adapted before inclusion in the thesis.

# Contents

# List of Figures

# List of Tables

# Listings

# List of acronyms and abbreviations

| | |
|---|---|
| API | Application Programming Interface |
| BM25 | Best Matching 25 |
| CORAG | Cost-Constrained Retrieval Optimization for Retrieval-Augmented Generation |
| CRAG | Corrective Retrieval-Augmented Generation |
| FLARE | Forward-Looking Active Retrieval |
| JSON | JavaScript Object Notation |
| LLM | Large Language Model |
| NLI | Natural Language Inference |
| NLP | Natural Language Processing |
| PDF | Portable Document Format |
| RAG | Retrieval-Augmented Generation |
| RAG-HAT | Retrieval-Augmented Generation Hallucination Aware Tuning |
| RAGAS | Retrieval-Augmented Generation Assessment |
| ReDeEP | Regressing Decoupled External context score and Parametric knowledge score |
| RGB | Retrieval-Augmented Generation Benchmark |
| UAEval4RAG | Unanswerability Evaluation for Retrieval-Augmented Generation |

# Chapter 1

# Introduction

Large language models (LLMs) have rapidly become a central technology in natural language processing (NLP) due to their strong performance across a wide range of generative and knowledge-intensive tasks [1]. Their ability to produce coherent text, answer questions, summarize documents, and support domain-specific applications has created significant interest in deploying them in practical systems. At the same time, many real-world use cases require more than fluent language generation. They require responses that are accurate, up-to-date, and grounded in relevant sources, especially in settings where users rely on the system to support decision-making or knowledge work [2].

Despite their capabilities, LLMs remain prone to hallucinations, that is, generating content that is false, unsupported, or misleading while still appearing plausible [3]. This problem limits their reliability in real-world deployments and becomes even more critical in knowledge-intensive settings, where outdated model parameters or missing domain knowledge can lead to confident but incorrect answers. As a result, trustworthiness has emerged as a central concern in the practical adoption of LLM-based systems, motivating growing research on factuality, faithfulness, robustness, and hallucination mitigation [4, 5].

Retrieval-augmented generation (RAG) has emerged as a promising approach to address these limitations by combining generation with retrieval from external knowledge sources. By incorporating retrieved documents as contextual input, RAG enables responses to be grounded in explicit evidence and adapted to evolving information [6]. This approach improves the factual basis of generated outputs and extends the applicability of LLMs to knowledge-intensive tasks.

However, retrieval augmentation does not by itself guarantee reliable

behavior. Errors may arise from poor retrieval quality, incomplete or noisy context, or limitations in how the model uses the retrieved information. As a result, even RAG-based systems may produce unsupported claims, fail to handle unanswerable queries appropriately, or generate responses that are not fully grounded in the available evidence [7, 8, 9].

This thesis addresses these challenges by investigating how a RAG system can be designed with a stronger focus on reliability and hallucination awareness. The work takes a system-building and evaluation perspective, where different pipeline configurations are implemented and examined with respect to retrieval quality, answer faithfulness, and hallucination-related behavior. In doing so, the thesis aims to contribute both an engineered prototype and an empirical analysis of how reliability can be improved in practical RAG systems. The remainder of this chapter introduces the background, problem formulation, purpose, goals, methodology, delimitations, and overall structure of the thesis.

## 1.1 Background

LLMs have demonstrated strong capabilities across many language tasks, but their performance in knowledge-intensive settings can degrade when relevant knowledge is rare or insufficiently represented in pretraining data [10]. Because responses are generated partly from knowledge encoded in model parameters, they may produce information that is outdated, unsupported, or factually incorrect. This challenge is commonly discussed in terms of hallucination and has become a central obstacle to the trustworthy deployment of LLM-based systems [3, 5].

RAG addresses this limitation by incorporating an explicit retrieval step that identifies relevant information from an external knowledge source and conditions the generation process on this retrieved context. This design enables models to access up-to-date information and improves their ability to produce grounded responses in knowledge-intensive tasks [6].

A typical RAG pipeline consists of several interconnected components. Documents are first collected from a target knowledge source and transformed into smaller units suitable for indexing and retrieval. When a query is submitted, a retriever identifies the most relevant chunks from the indexed collection, and these retrieved passages are provided to the language model as contextual evidence for response generation. From a system design perspective, RAG can therefore be understood as a modular pipeline in which

retrieval and generation jointly determine the quality of the final output [11, 12].

The effectiveness of such systems depends on the interaction between multiple components, including retrieval quality, context selection, and the model's ability to correctly interpret and use the retrieved evidence. Prior work has shown that RAG systems may still produce unsupported claims, remain unfaithful to retrieved content, and perform poorly on unanswerable queries if they are not carefully designed and evaluated [13, 8, 14, 15, 16].

This background motivates the focus of the present thesis. Since RAG systems are composed of multiple interacting components and their reliability depends on more than retrieval alone, this thesis investigates how a RAG pipeline can be designed and evaluated with explicit attention to hallucination awareness and reliability-related behavior. The background presented here serves to introduce the core concepts and terminology used throughout the remainder of the thesis.

## 1.2 Problem

Although RAG is widely used to improve the factual grounding of LLMs, it does not eliminate the risk of unreliable answers. In knowledge-intensive settings, the quality of generated responses depends on several interacting components, including retrieval effectiveness, context selection, prompt construction, and the model's ability to use retrieved evidence faithfully. Consequently, a RAG system may still generate unsupported or misleading content even when external documents are provided as context [3, 17, 18, 19].

This creates an important problem for both practice and research. From a practical perspective, unreliable answers reduce the usefulness of RAG systems in real applications, especially when users expect responses to be grounded in a known knowledge base. From a research perspective, the problem is not only whether RAG improves performance in general, but also how specific design choices affect hallucination behavior, transparency, and trustworthiness. These issues motivate the focus of this thesis.

### 1.2.1 Practical Problem

In practice, RAG is often presented as a way to reduce hallucinations by grounding generation in retrieved documents. However, existing RAG systems can still hallucinate, misinterpret retrieved evidence, or produce answers that are only partially supported by the available context [14, 15, 3]. This means

that the presence of retrieval alone is not sufficient to guarantee reliable behavior. If the retrieved information is irrelevant, incomplete, or poorly integrated into the answer generation process, the final output may still be incorrect or misleading [7, 15].

A further practical challenge is that reliability is strongly affected by pipeline configuration. Design choices such as how documents are chunked, how retrieval is performed, how many documents are included, and how context is presented to the model can all influence system behavior [20, 21, 7, 8]. As a result, practitioners who want to build dependable RAG applications need more than a general understanding of the method. They need concrete ways of designing and evaluating systems that behave more reliably within the constraints of a specific knowledge base and application setting.

### 1.2.2 Research Problem

From a research perspective, there is a need to systematically examine how different RAG design choices influence reliability-related properties such as retrieval quality, answer faithfulness, and hallucination behavior. While prior work demonstrates that RAG can improve factual grounding, it also shows that performance depends on the interaction between multiple components rather than on retrieval alone [22]. This makes it necessary to study RAG as a pipeline whose parts must be configured and evaluated together.

This thesis addresses that need through the design and evaluation of a prototype RAG system built on a fixed-size knowledge base. The work takes a system-building perspective and investigates how different pipeline configurations affect the behavior of the system, with particular attention to hallucination awareness and reliability. In addition to implementing the system, the thesis evaluates how well different configurations support grounded answers and expose reliability-related strengths and weaknesses.

The research question guiding this thesis is the following: “How can a RAG system be designed to reduce hallucinations while providing transparent and trustworthy answers in a fixed-size knowledge base?”

To operationalise this research question, the thesis investigates three sub-questions:

1. How do different embedding models and retrieval methods affect the relevance and grounding of the retrieved information?

2. How can the system identify cases where retrieved evidence is insufficient to generate a reliable and grounded response?

3. How can the system detect and communicate situations of high uncertainty or insufficient supporting information?

## 1.3 Purpose

The purpose of this thesis is to investigate how a RAG system can be designed to reduce hallucinations while providing more transparent and trustworthy answers within a fixed-size knowledge base. The thesis takes a system-building and evaluation perspective, with the aim of developing a prototype RAG system and examining how different design choices affect its reliability-related behavior. In this way, the work seeks to contribute both practical and analytical insight into the design of more dependable RAG systems.

The project is expected to benefit several groups. For practitioners and developers, the results may provide guidance on how different RAG pipeline configurations influence answer quality, grounding, and hallucination behavior in practice. For researchers, the thesis contributes an empirical case study of how reliability can be examined at the system level rather than only at the level of individual model outputs. More broadly, users of LLM-based systems may benefit indirectly from approaches that improve the transparency and trustworthiness of generated answers, especially in settings where incorrect or unsupported information may reduce confidence in the system.

The project also has ethical and social relevance. Since RAG systems may still generate misleading or insufficiently supported answers, their deployment raises concerns related to user trust, responsible AI use, and the risk of presenting incorrect information with high confidence. Reducing such behavior is therefore important not only from a technical perspective but also from a societal one. In addition, a more systematic understanding of reliable RAG design may support more efficient development practices by reducing unnecessary experimentation and helping practitioners make better-informed engineering decisions.

## 1.4 Goals

The main goal of this degree project is to develop and evaluate a RAG system with a focus on reliability and hallucination awareness.

To achieve this overall goal, the project is divided into the following sub-goals:

1. **Design and implement a prototype RAG system.** The first sub-goal is to develop a working prototype that integrates the core components of a RAG pipeline, including document processing, retrieval, context construction, and answer generation, with explicit attention to reliability and hallucination awareness.

2. **Evaluate how different pipeline configurations affect system reliability.** The second sub-goal is to investigate how design choices in the RAG pipeline influence retrieval quality, answer faithfulness, and hallucination-related behavior through empirical evaluation. This sub-goal is intended to contribute knowledge that is relevant both to engineering practice and to research on reliable RAG systems.

3. **Conduct the project using a systematic engineering and research process.** The third sub-goal is to carry out the work in a scientifically grounded and well-documented manner, including problem formulation, methodology selection, implementation, evaluation, and critical reflection. This also supports the learning objectives of the degree project by demonstrating the ability to independently plan, execute, and report a technically advanced investigation.

The main deliverables of the project are the implemented prototype RAG system, the evaluation results from the tested pipeline configurations, and the written thesis documenting the problem, design choices, methodology, results, and conclusions.

## 1.5 Research Methodology

This thesis is based on a pragmatic research perspective, since it focuses on addressing a concrete engineering problem through the design and evaluation of a working system. The work does not aim primarily to develop new theory, but to generate practically useful knowledge about how a RAG system can be designed to behave more reliably in a fixed-size knowledge base setting. At the same time, the study is empirical, since conclusions are drawn from observations and comparisons made during the implementation and evaluation of the system.

The overall research approach is design-oriented and experiment-based. The thesis includes the development of a prototype RAG system and the systematic evaluation of different pipeline configurations in order to examine how design choices influence retrieval quality, answer faithfulness, and

hallucination-related behavior. This approach is appropriate because the research question concerns both the construction of an artifact and the analysis of its behavior under different technical conditions.

The main research methods used in the thesis are therefore prototype development, controlled experimentation, and comparative evaluation. The prototype serves as the artifact through which design choices can be implemented and tested, while the experiments provide a structured way to compare alternative configurations and assess their strengths and weaknesses. This makes it possible to study reliability not only as a theoretical concept but also as a measurable property of a concrete system.

Other methodological alternatives were considered less suitable for the purpose of this thesis. A purely theoretical study would not have been sufficient, since the research question requires the implementation and testing of an actual system. A qualitative user study was also not chosen as the primary method, since the main focus is on technical system behavior rather than user perception. Likewise, a large-scale deployment study was outside the scope of the project because the thesis is limited to a prototype evaluated in a controlled setting. A more detailed description of the methodology, methods, data collection, evaluation procedure, and quality considerations is presented in Chapter 3.

## 1.6 Delimitations

This thesis is delimited to the design and evaluation of a prototype RAG system operating on a fixed-size knowledge base. The work focuses on how selected pipeline design choices influence reliability-related behavior, with particular attention to hallucination reduction, answer transparency, and trustworthiness. As a result, the thesis does not aim to address the full range of challenges associated with LLMs or retrieval-augmented systems in general.

Several specific delimitations apply to the project. First, the study is limited to a controlled experimental setting and does not include deployment in a real-world production environment. Second, the work focuses on text-based retrieval and generation and does not consider multimodal RAG systems involving images, audio, or video. Third, only a bounded set of pipeline configurations, models, prompts, retrieval strategies, and evaluation methods can be examined within the scope of the degree project. This means that the results should be interpreted in relation to the selected system design rather than as universally valid conclusions for all RAG systems.

In addition, the thesis focuses primarily on technical system behavior

rather than on user-centered evaluation. It therefore does not include large-scale user studies of usability, trust perception, or human-computer interaction. Broader issues such as fairness, privacy, and computational cost are relevant to the responsible use of RAG systems, but they are not investigated in depth in the experimental part of this thesis. These delimitations are necessary in order to keep the project feasible and to allow a more focused investigation of reliability and hallucination awareness within the chosen setting.

## 1.7 Structure of the Thesis

This thesis is organized as follows. Chapter 2 presents the theoretical background and related work on RAG, hallucination behaviour, and evaluation methods. Chapter 3 describes the research methodology, including the experimental design, evaluation dataset, and metrics used in the project. Chapter 4 presents the design and implementation of the modular RAG prototype. Chapter 5 reports and analyses the experimental results across the evaluated pipeline configurations. Chapter 6 discusses the implications of the findings for reliable and hallucination-aware RAG system design. Finally, Chapter 7 concludes the thesis, discusses limitations, reflects on the work, and outlines directions for future development.

# Chapter 2

# Background

This chapter provides the theoretical and technical background necessary to understand the work presented in this thesis. It introduces LLMs and the motivation for RAG, presents key concepts related to reliability and hallucination in RAG systems, and reviews existing approaches for evaluation and improvement. The chapter also summarizes related work and identifies the research gap addressed by this thesis.

## 2.1 LLMs and Knowledge-Intensive Generation

LLMs have demonstrated strong capabilities across a wide range of language tasks and have become increasingly important in practical natural language processing applications [23]. Modern LLMs are largely based on the transformer architecture, which introduced self-attention as a scalable mechanism for sequence modeling [24]. Their generative abilities make them useful for tasks such as question answering, summarization, reasoning, and domain-specific assistance [25, 26].

Prominent examples of such models include systems in the GPT [27], LLaMA [28], and Mistral [29] families, among others, which are available in a range of parameter sizes and have contributed to the widespread adoption of LLM-based applications. At the same time, many real-world applications require more than fluent text generation. They require responses that are factually accurate, grounded in relevant knowledge, and robust when users ask questions that depend on specific external information or specialized domain content [11, 2].

A central challenge in such settings is that LLMs rely heavily on parametric knowledge encoded during pretraining. While this allows them to produce coherent and often highly capable responses, it also means that their performance may degrade when relevant knowledge is rare, insufficiently represented in the training data, or no longer current [30]. Prior work shows that LLMs struggle particularly with long-tail knowledge and may generate outputs that are outdated, unsupported, or factually incorrect [10]. This limitation is closely connected to the broader problem of hallucination, which has become a major obstacle to the reliable deployment of LLM-based systems in knowledge-intensive applications[19, 3, 5].

These limitations have motivated approaches that augment language models with external knowledge sources at inference time. Rather than depending only on information stored in model parameters, such approaches allow the model to access retrieved evidence that is relevant to the current query. Among these approaches, RAG has emerged as one of the most influential paradigms for improving factual grounding and adapting generation to knowledge-intensive tasks [6, 31].

## 2.2 Retrieval-Augmented Generation

RAG was first introduced by Lewis et al. as a framework for addressing knowledge-intensive natural language processing tasks through the combination of neural retrieval and sequence generation [6]. The original idea was to augment a parametric language model with access to an external document collection, so that the model could retrieve relevant passages and use them as supporting context during answer generation. This approach was motivated by the observation that purely parametric models may struggle to provide factual, specific, and easily updateable responses when the required knowledge is not reliably captured in model parameters alone. RAG therefore established an important paradigm in which language generation is grounded in retrieved external evidence rather than depending exclusively on internal parametric knowledge [6].

### 2.2.1 Basic RAG Architecture

In its basic form, a RAG system follows a retrieve-then-generate pipeline. A user first submits a query to the system, after which a retrieval component searches an external knowledge source in order to identify information that is likely to be relevant to the query. The retrieved information is then

incorporated into the input of a large language model, which generates a response conditioned on both the user query and the retrieved context. In this way, the system combines two complementary capabilities: retrieval provides access to explicit external knowledge, while generation provides the ability to synthesize that knowledge into fluent natural language output [6, 2, 11].

The original formulation by Lewis et al. introduced RAG as a combination of a neural retriever and a sequence-to-sequence generator for knowledge-intensive natural language processing tasks [6]. The retriever is responsible for locating candidate passages from an external corpus, while the generator uses those passages as supporting evidence during answer generation. This design addresses an important limitation of purely parametric language models: although such models may encode substantial knowledge in their parameters, they do not provide reliable or transparent access to that knowledge and may struggle when the required information is rare, recently updated, or insufficiently represented in training data [10]. By retrieving supporting documents at inference time, a basic RAG system can ground its answers in explicit external content rather than relying only on internal parametric memory [31]. Figure 2.1 illustrates this basic retrieve-then-generate architecture.

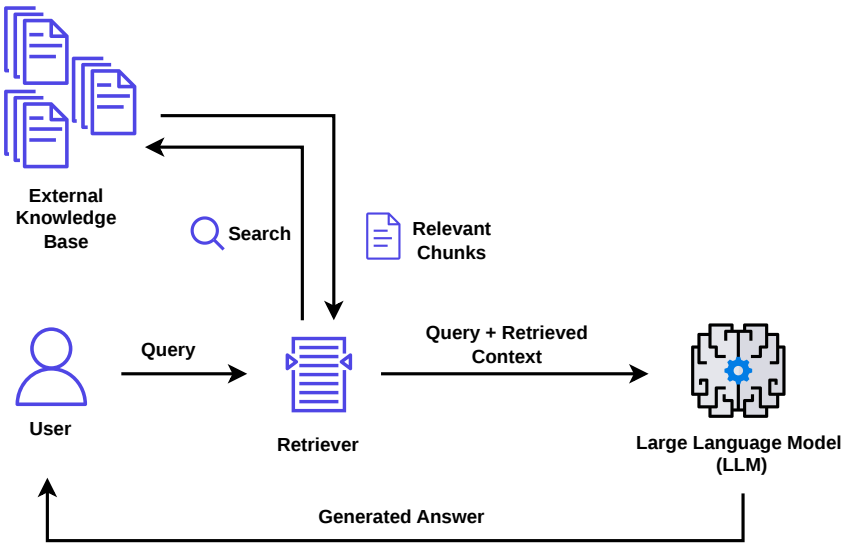


Figure 2.1: Basic RAG architecture.

A basic RAG pipeline typically contains several core components. The first is the *knowledge source*, which may consist of documents, passages,

manuals, reports, web pages, or other textual resources relevant to the target domain. Since entire documents are often too large and too heterogeneous to retrieve effectively, the source material is usually preprocessed into smaller textual units, commonly referred to as *chunks* [21]. These chunks are then indexed so that they can be searched efficiently during inference. In many modern systems, indexing is performed using vector representations, where chunks and queries are embedded into a shared semantic space and similarity search is used to identify relevant candidates [32, 33, 34].

The second major component is the *retriever*. When a user submits a query, the retriever transforms the query into a suitable representation and compares it against the indexed collection in order to retrieve the most relevant chunks. In a basic architecture, this often results in a top-$k$ set of retrieved passages that are assumed to provide useful evidence for answer generation [6, 12, 21]. The quality of this step is critical, because the generator can only use information that has actually been retrieved. If the retrieved passages are irrelevant, incomplete, or noisy, the generated answer may still be misleading even though the system is nominally retrieval-augmented. Retrieval quality therefore plays a foundational role in the overall performance of a RAG pipeline [7, 35, 36].

The third component is the *generator*, typically a large language model that receives both the original query and the retrieved passages as input. The retrieved content is commonly inserted into the prompt as context, often together with formatting or system instructions intended to guide the model toward using the evidence appropriately. The model then generates an answer that ideally reflects the retrieved material while also providing a coherent and contextually appropriate response to the user. In principle, this allows the system to produce answers that are more factually grounded and more adaptable to the knowledge source than answers generated purely from model parameters [6, 31].

An important feature of the basic RAG architecture is the separation between *knowledge access* and *language generation*. In a purely parametric system, both functions are implicitly combined in the model parameters. In a RAG system, by contrast, retrieval is handled by a separate component that accesses explicit external evidence. This separation offers several advantages. First, the knowledge source can be updated without retraining the language model. Second, retrieved passages can in principle make the basis of the answer more transparent, since the system can expose or cite the supporting context. Third, the architecture is modular, meaning that individual components such as the chunking strategy, retriever, reranker, prompt

construction method, or language model can be modified independently [11, 12, 2].

At the same time, even a basic RAG architecture involves several non-trivial design decisions. The system designer must decide how documents should be segmented, how chunks should be represented, which retrieval method should be used, how many passages should be included, and how the retrieved context should be presented to the model. These choices influence both the relevance of the retrieved evidence and the model's ability to use that evidence effectively. As a result, the performance of a RAG system depends not only on the quality of the language model, but also on the interaction between retrieval, context construction, and generation [7, 22, 12].

### 2.2.2 Formal Formulation

Formally, let $q$ denote an input query and let $D = \{d_1, d_2, \ldots, d_n\}$ denote the external document collection. The retrieval component selects a subset of relevant documents or passages from $D$ based on their estimated relevance to $q$. In a basic top-$k$ setting, this can be written as

$$R(q) = \{d_{(1)}, d_{(2)}, \ldots, d_{(k)}\},$$

where $R(q)$ is the set of retrieved passages and $d_{(i)}$ denotes the $i$-th highest-ranked passage according to the retrieval function.

The generation component then produces an answer $a$ conditioned on both the query and the retrieved context. This can be expressed as

$$a \sim p(a \mid q, R(q)),$$

where $p(a \mid q, R(q))$ denotes the conditional probability distribution over possible answers given the query and the retrieved evidence.

In dense retrieval settings, both queries and documents are typically mapped into a shared vector space. Let $e_q$ denote the embedding of the query and $e_d$ the embedding of a document chunk. Retrieval can then be based on a similarity function $s(e_q, e_d)$, such as cosine similarity or inner product, and the retriever selects the top-$k$ passages with the highest similarity scores.

### 2.2.3 From Naive RAG to Advanced and Modular RAG

The development of RAG can be understood as a progression from a simple retrieve-then-generate pipeline toward increasingly sophisticated and

configurable architectures. A common starting point in the literature is *naive RAG*, which represents the most direct implementation of the original idea introduced by Lewis et al. In naive RAG, a user query is used to retrieve a fixed number of passages from an external knowledge source, and these passages are then provided to the language model as context for answer generation [6]. This formulation captures the central intuition behind RAG, namely that generation can be improved by grounding the model in retrieved external evidence rather than relying exclusively on parametric knowledge [2, 31]. Figure 2.2 illustrates this standard naive RAG pipeline, in which retrieval is performed once before the retrieved passages are passed to the generator.

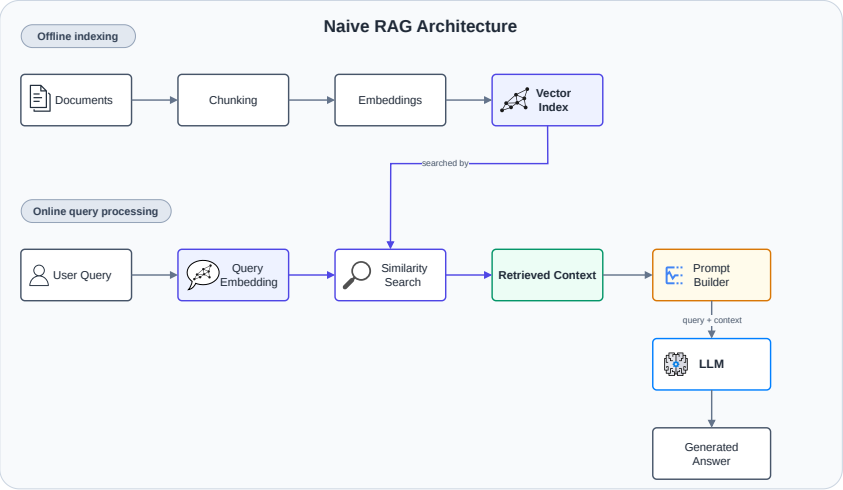


Figure 2.2: Naive RAG architecture.

Although naive RAG provides a useful baseline, it is often insufficient for more demanding applications. Its effectiveness depends heavily on the quality of the initial retrieval step, the relevance of the selected passages, and the model's ability to use the retrieved evidence appropriately during generation [15, 36, 30]. In practice, a single retrieval step may fail to capture all necessary evidence, while retrieved context may contain noise, redundancy, or only partial support for the answer [37]. These limitations motivated the development of *advanced RAG*, in which retrieval and context construction are refined through additional processing stages.

A useful way to understand advanced RAG is to distinguish between *pre-retrieval* and *post-retrieval* enhancements. Pre-retrieval methods aim to improve what is retrieved in the first place. They include decisions and techniques related to chunking, vectorization, indexing, query reformulation,

hierarchical retrieval, hypothetical query generation, and hybrid or fusion retrieval strategies [2, 31, 38, 39, 21]. In this stage, the goal is to improve the alignment between the user query and the indexed knowledge representation before the final evidence is passed to the generator. This perspective is also reflected in practical overviews of advanced RAG pipelines, which treat chunking, indexing, search structure, and retrieval strategy as core components of advanced retrieval design. Figure 2.3 summarizes this expanded pipeline by showing how advanced RAG introduces additional processing around the basic retrieval and generation stages.

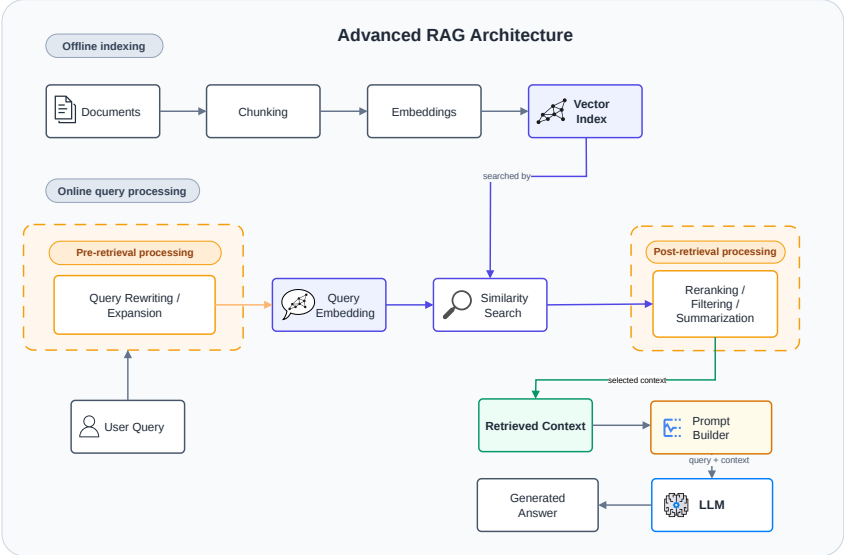


Figure 2.3: Advanced RAG architecture.

Post-retrieval methods, by contrast, operate on the material that has already been retrieved. Their purpose is to improve the quality of the context that will ultimately be given to the language model. Typical post-retrieval steps include reranking, filtering, context compression, context enrichment, and other transformations that attempt to remove noise, prioritize the most relevant evidence, or reconstruct a more useful context window for generation [2, 30, 20]. In this sense, advanced RAG departs from the assumption that retrieved top-$k$ passages can simply be forwarded directly to the generator. Instead, retrieved evidence is treated as an intermediate result that may need to be refined before generation begins.

A further development in this progression is the idea of *modular RAG*. While advanced RAG adds sophistication to the pipeline, modular RAG reframes the system more fundamentally as a composition of separable

functional modules [12, 11, 2]. In this perspective, a RAG system is not treated as one fixed architecture, but as a flexible framework composed of modules such as query processing, retrieval, reranking, context construction, generation, and post-generation verification. This makes it possible to analyze the pipeline as a configurable system in which different components can be adapted or replaced depending on the requirements of the task. Figure 2.4 illustrates this modular view by presenting RAG as a set of configurable components rather than a single fixed retrieve-then-generate pipeline.

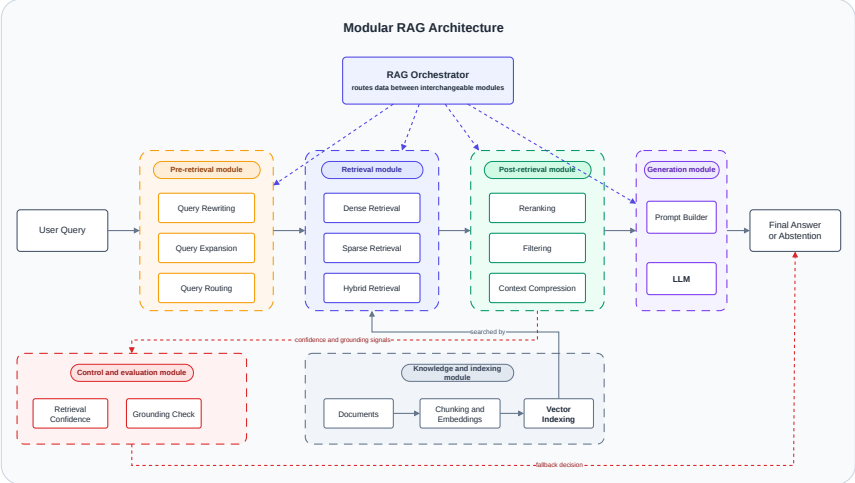


Figure 2.4: Modular RAG architecture.

The modular perspective is particularly useful because many recent RAG methods can be understood precisely as modifications or extensions of individual pipeline modules [12, 40, 30, 41, 39]. Some methods focus on query reformulation and retrieval planning, while others focus on reranking, context selection, retrieval quality estimation, generation-time self-correction, or post-generation verification . In this sense, modular RAG provides a broader architectural framework for understanding how modern RAG systems are designed and why they differ from the earlier naive formulation.

The progression from naive to advanced and modular RAG therefore reflects a broader maturation of the field. What began as a relatively simple retrieve-then-generate mechanism has developed into a richer design space in which retrieval, context construction, generation, and verification can be adapted to the needs of the application. This evolution is especially important for reliability-oriented systems, since it highlights that improved performance and reduced hallucination are often achieved through coordinated pipeline design rather than through a single retrieval step [15, 16].

### 2.2.4 Applications of RAG

RAG has been applied across a growing range of domains in which access to external, up-to-date, or domain-specific knowledge is essential. Surveys of the field show that RAG is no longer limited to open-domain question answering, but is increasingly used in applications that require stronger factual grounding, better interpretability, and adaptation to specialized data sources [2, 31]. This broad adoption reflects the practical value of combining language generation with external retrieval, especially in settings where unsupported or outdated answers would significantly reduce the usefulness of the system.

One important application area is healthcare and medicine. In these settings, RAG is attractive because medical decision support, diagnostic reasoning, and clinical question answering often depend on accurate access to external knowledge rather than on parametric memory alone. Recent work on MedRAG shows how RAG can be combined with knowledge-graph-elicited reasoning to support healthcare copilots and reduce misdiagnosis in diagnostic tasks [42]. Large-scale evaluations in medicine further demonstrate that carefully designed medical RAG systems can substantially improve performance across medical question answering tasks, while also revealing that component choices such as corpora, retrievers, and backbone models strongly affect results [43]. Related work in nephrology likewise highlights the potential of combining LLMs with RAG to support both patient care and education in a specialized medical field, while emphasizing the importance of reliability, prompting, and domain adaptation in real deployments [44].

RAG has also shown promise in educational contexts. A recent systematic survey on RAG for educational applications describes how RAG can support interactive learning systems, educational content generation, assessment, and large-scale educational ecosystems [45]. The same line of work argues that external retrieval is particularly valuable in education because it improves factual accuracy, supports dynamic knowledge updates, and makes LLM-based tools more suitable for scenarios in which learners and teachers depend on current and verifiable information [45].

Another practical area is summarization and information-intensive content processing. Work on benchmarking LLMs for news summarization illustrates the importance of factual consistency and robust use of source material in summarization tasks [26]. In addition, the broader RAG literature has emphasized applications in AI-generated content and document-grounded generation, where retrieved evidence can improve factual grounding and make generation more suitable for real-world content production workflows [31].

This is closely related to application scenarios in which the system must synthesize information from external sources while maintaining accuracy and relevance.

RAG has also shown promise in the automotive industry, where engineering and organizational work depends on efficient access to extensive technical knowledge. A case study from an industrial automotive setting shows that RAG-based search can improve task efficiency, support more complex information needs, and increase correctness compared with traditional search tools, highlighting its potential for knowledge-intensive engineering workflows [46].

RAG is also being extended to multimodal and specialized settings, including code, graphs, and other structured or semi-structured knowledge forms [2]. Examples from the literature include graph-based RAG approaches for textual graphs and networked information, as well as multimodal RAG methods that extend the same principles beyond plain text [47, 48, 49]. These developments suggest that RAG should be understood not merely as a technique for a narrow set of benchmark tasks, but as a broader paradigm for grounding generation in external knowledge across diverse application areas.

The growing range of applications is particularly relevant to this thesis. It shows that the importance of RAG lies not only in improved performance on isolated tasks, but also in its potential to support more reliable and practically useful LLM systems in domains where factual grounding, transparency, and trustworthiness are essential.

## 2.3 Reliability and Hallucination in RAG Systems

Although RAG is designed to improve factual grounding by incorporating external knowledge, it does not eliminate the risk of unreliable outputs. In practice, RAG systems may still generate unsupported, misleading, or contextually unfaithful answers due to failures in retrieval, weaknesses in context use, or limitations in the generation process itself [3, 17, 4]. For this reason, understanding reliability in RAG systems requires more than a general discussion of model performance. It also requires careful attention to how hallucination is defined, what causes it in retrieval-augmented pipelines, and which system properties contribute to more trustworthy behavior.

### 2.3.1 Hallucination, Confabulation, and Related Terminology

Hallucination is commonly used as an umbrella term for cases in which a language model produces content that is false, unsupported, or misleading while still appearing plausible [3, 5]. In the context of RAG systems, hallucination typically refers not only to fabricated claims in the traditional sense, but also to answers that are insufficiently grounded in the retrieved evidence, inconsistent with the provided context, or unjustified given the available information [14, 15]. This makes hallucination a particularly important concept in retrieval-augmented systems, where the expectation is that external knowledge should constrain the generation process.

At the same time, some researchers have argued that the term *hallucination* may be imprecise or even misleading when applied to language models. In particular, the term *confabulation* has been proposed as an alternative for cases in which a model generates coherent and plausible statements without adequate evidential support, especially when no perception-like process is involved [50]. This distinction is intended to emphasize that such errors do not necessarily imply random fabrication, but may instead reflect the model's tendency to produce fluent outputs even when its knowledge is incomplete, conflicting, or poorly grounded.

Related terminology is also important for clarifying what kind of failure is being discussed. *Factuality* generally concerns whether a generated statement is correct with respect to the world or an accepted source of truth, whereas *faithfulness* concerns whether the generated answer is properly supported by the evidence or context provided to the model [15, 9]. In a RAG setting, these concepts are closely related but not identical: an answer may be factually correct in general while still being unfaithful to the retrieved documents, or it may appear consistent with the retrieved context while remaining misleading or incomplete. Similarly, unsupported claims, contradictions to retrieved evidence, and failures to abstain on unanswerable queries can all be understood as reliability problems, even when they do not take the form of explicit fabrication [8, 14].

In this thesis, the term *hallucination* will be used as the primary term, since it remains the dominant term in the literature on LLMs and RAG systems. However, the broader discussion also recognizes that some forms of hallucination in retrieval-augmented systems are better understood as failures of grounding, faithfulness, or evidential support rather than as simple fabrication. This distinction is important because the goal of a reliable RAG

system is not merely to avoid obviously false statements, but to generate answers that are appropriately supported, transparent, and trustworthy in relation to the available evidence [3, 4].

### 2.3.2 Causes of Hallucination in RAG Pipelines

Hallucinations in RAG systems are not caused by a single failure point, but can emerge from multiple stages of the pipeline. A useful way to understand these failures is to distinguish between retrieval-related problems and generation-related deficiencies [3]. Retrieval failures concern cases in which the system does not provide the model with the right evidence, while generation deficiencies concern cases in which the model fails to use the available evidence appropriately. This distinction is important because a RAG system may still produce unreliable answers even when retrieval is present, and improving reliability therefore requires attention to the full pipeline rather than to retrieval alone.

One group of causes arises in the retrieval stage itself. If the knowledge source is incomplete, outdated, poorly structured, or misaligned with the user's information need, the retriever may fail before the generation stage even begins. In addition, query formulation, chunking strategy, embedding quality, retriever design, and retrieval strategy all influence whether the retrieved passages are actually relevant and sufficient for answering the query [3, 21, 38, 39]. In such cases, the model may receive irrelevant, partial, or low-quality evidence, which increases the risk of unsupported or misleading answers.

A second group of causes arises from the quality of the retrieved context. Even when retrieval returns documents related to the query, the retrieved set may still contain noise, redundancy, contradictions, or only partial support for the answer. Retrieved passages may also overwhelm the model if too much weakly relevant material is included, making it harder to identify the most important evidence. Prior work has shown that noisy or irrelevant retrieved documents can actively mislead the model, while incomplete retrieval can leave crucial evidence missing from the context altogether [36, 30, 20]. Recent work such as Debate-Augmented RAG (DRAG) illustrates that hallucination mitigation in RAG must address both retrieval and generation. Its retrieval-debate stage targets incomplete or poorly covered evidence, while its response-debate stage targets failures in reasoning over the retrieved context. This distinction supports the view that hallucinations may arise not only from poor document retrieval, but also from inadequate interpretation or integration of

otherwise available evidence [51]. These problems illustrate that retrieval quality should not be understood only in terms of document ranking, but also in terms of the usefulness and coherence of the final context provided to the generator.

A third group of causes concerns the generation stage. Even when relevant evidence is present in the context window, a language model may still fail to use it faithfully. The model may ignore retrieved passages, mix them incorrectly with its own parametric knowledge, overgeneralize beyond the available evidence, or generate claims that are only partially supported by the retrieved context [15, 14]. This problem is especially important in RAG systems because retrieval does not automatically guarantee faithfulness. A model may produce an answer that appears plausible and well-formed while still being poorly grounded in the evidence that was retrieved.

Another important challenge concerns cases in which the available evidence is insufficient to support an answer at all. In such situations, a reliable RAG system should ideally abstain, indicate uncertainty, or explicitly state that the answer cannot be derived from the available knowledge base. However, many systems still attempt to generate an answer even when the retrieved evidence is inadequate, which increases the risk of hallucination [8, 36]. This highlights that hallucination in RAG is not only a matter of incorrect content generation, but also of when the system should decline to answer.

Taken together, these observations show that hallucinations in RAG pipelines can originate from the source collection, the retriever, the retrieved context, the generator, or the interaction among these components. The problem is therefore best understood as a system-level reliability issue rather than as a weakness of the language model alone. This perspective is particularly important for the present thesis, since it motivates evaluating RAG pipelines not only in terms of output quality, but also in terms of retrieval behavior, faithfulness to retrieved evidence, and handling of insufficient or noisy context [3, 4, 17].

### 2.3.3 Trustworthiness, Transparency, and Faithfulness

In the context of RAG systems, reliability is closely related to the broader concept of *trustworthiness*. A trustworthy system is not defined only by whether it produces correct answers in isolated cases, but by whether its behavior can be relied upon across different queries, contexts, and knowledge conditions [4]. This includes the ability to provide grounded answers when

sufficient evidence is available, to avoid unsupported claims when evidence is weak or conflicting, and to respond appropriately when the available information is insufficient. Trustworthiness in RAG systems is therefore a system-level property that depends on both retrieval and generation, rather than on either component alone [17, 3].

A closely related concept is *transparency*. In retrieval-augmented systems, transparency refers to the extent to which the basis of an answer can be inspected, traced, or communicated to the user. Because RAG systems retrieve explicit external documents, they can in principle expose the passages, sources, or contextual evidence used during generation. This makes them potentially more interpretable than purely parametric systems, even though transparency by itself does not guarantee correctness [4]. A system may show retrieved passages and still produce an answer that misinterprets them, blends them incorrectly with parametric knowledge, or overstates what the evidence actually supports.

For this reason, faithfulness is a central property in RAG systems. Since generated answers are expected to be grounded in external evidence, reliability depends not only on plausibility but also on whether the response is supported by the retrieved context. A fluent answer may still overstate, distort, or selectively use the available evidence. Reliable RAG systems should therefore produce responses that remain aligned with the retrieved material and are justified by the provided context [15, 14, 9].

## 2.4 Evaluation of RAG Systems

Evaluating RAG systems is more complex than evaluating standalone language models, because the overall quality of a RAG system depends on both the retrieval component and the generation component, as well as on the interaction between them. A system may retrieve highly relevant evidence but still generate an unfaithful answer, or it may generate a plausible answer despite weak or incomplete retrieval. For this reason, recent work has emphasized that RAG evaluation should not be reduced to final answer quality alone, but should instead assess multiple dimensions, including retrieval quality, answer faithfulness, hallucination behavior, and the handling of unanswerable queries [9, 52, 8].

This multi-dimensional perspective is especially important in reliability-oriented RAG systems. Frameworks such as RAGAS (Retrieval Augmented Generation Assessment) explicitly separate retrieval-related and generation-related evaluation, while benchmark-oriented studies show that current

RAG systems continue to struggle with challenges such as noisy context, information integration, and negative rejection [9, 52]. In addition, unanswerability-focused evaluation highlights that a reliable RAG system must not only answer correctly when evidence is available, but also refrain from unsupported answering when the knowledge base does not contain sufficient information [8]. These considerations motivate a closer examination of retrieval quality as a distinct and foundational aspect of RAG evaluation.

### 2.4.1 Retrieval Quality

Retrieval quality is a central component of RAG evaluation because the generator can only make use of the evidence that has actually been retrieved. If the retrieved passages are irrelevant, incomplete, weakly ranked, or noisy, even a strong language model may fail to produce a reliable answer. In this sense, retrieval quality concerns not only whether the system returns passages that are related to the query, but also whether those passages provide sufficient and appropriately prioritized evidence for downstream generation [9, 52].

A useful evaluation perspective is therefore to consider retrieval quality in terms of relevance, sufficiency, and robustness. Relevance concerns whether the retrieved passages genuinely match the information need expressed by the query. Sufficiency concerns whether the retrieved evidence contains enough information to support a correct answer. Robustness concerns whether retrieval remains useful in more difficult settings, such as when distractor documents are present, when the query requires integrating multiple pieces of evidence, or when the correct system behavior is to reject an unanswerable request rather than force an answer [52, 8]. This broader view is important because top-ranked retrieval alone does not guarantee that the retrieved set will be adequate for reliable generation.

Recent evaluation work reinforces this point. RAGAS treats the retrieval component as independently measurable and emphasizes the importance of identifying relevant and focused context passages for the generator [9]. The RGB benchmark further shows that current LLMs in RAG settings still face substantial limitations in noise robustness, negative rejection, information integration, and counterfactual robustness, indicating that effective retrieval must be judged in relation to the demands of the task rather than by simple matching alone [52]. Similarly, UAEval4RAG shows that no single RAG configuration consistently performs best across answerable and unanswerable requests, highlighting that retrieval quality must also be assessed in relation to knowledge-base coverage and abstention behavior [8].

Taken together, these findings suggest that retrieval quality should be understood as a foundational but not self-sufficient component of RAG evaluation. High-quality retrieval makes reliable answer generation possible, but it does not by itself guarantee faithfulness or freedom from hallucination. For this reason, retrieval quality is best evaluated as one part of a broader reliability-oriented framework.

### 2.4.2 Benchmarks and Evaluation Frameworks

Because RAG systems combine retrieval and generation, their evaluation requires benchmarks and frameworks that capture multiple aspects of system behavior rather than a single end-to-end score. Existing evaluation resources differ in what they emphasize: some focus on retrieval and grounded answer quality, others on hallucination detection, multi-hop reasoning, or unanswerability, and some provide broader diagnostic views throughout the RAG pipeline [9, 52, 8]. As a result, benchmark selection has a direct influence on what kinds of strengths and weaknesses become visible during evaluation.

One influential framework is RAGAS, which provides a reference-free evaluation approach for RAG pipelines and separates retrieval-related and generation-related dimensions such as context relevance, faithfulness, and answer quality [9]. This is useful in practice because it supports faster evaluation cycles without requiring fully annotated gold answers for every query. However, as later work has shown, evaluator outputs can vary substantially depending on the evaluation setting, especially when abstention or refusal behavior is involved [53]. This suggests that framework outputs should be interpreted as informative signals rather than as unquestionable ground truth.

Alongside benchmark and framework development, recent work has also focused on methods for detecting hallucinations in generated outputs. SelfCheckGPT approaches this problem through response consistency, using disagreement across multiple sampled outputs as a signal that a claim may be non-factual [5]. ReDeEP (Regressing Decoupled External context score and Parametric knowledge score), by contrast, analyzes internal mechanisms in RAG models and detects hallucinations by examining how the model balances retrieved context against parametric knowledge [54]. These methods highlight that evaluation in RAG is not limited to benchmark scores alone, but also includes post-generation analysis aimed at identifying unreliable answers before they reach the user.

The RGB benchmark operationalizes several reliability stress tests for

RAG-enabled LLMs, including the ability to handle noisy context, reject unsupported queries, integrate information from multiple sources, and remain robust under counterfactual evidence. Its results show that these abilities remain challenging for current systems [52]. UAEval4RAG focuses specifically on unanswerable requests and introduces a taxonomy and metrics for evaluating whether a system correctly rejects questions that cannot be answered from a given knowledge base [8]. Other benchmarks such as RAGTruth provide manually annotated hallucination data for analyzing unsupported claims in RAG outputs, while diagnostic frameworks such as RAGChecker aim to provide more fine-grained measurements across both retrieval and generation modules [14, 55].

These developments show that no single benchmark or framework is sufficient for evaluating all relevant aspects of RAG reliability. Instead, effective evaluation typically requires a combination of retrieval-oriented metrics, faithfulness or groundedness assessment, hallucination detection, and unanswerability-aware testing. For a thesis concerned with reliable and hallucination-aware RAG, this implies that evaluation design must be aligned with the specific reliability properties under investigation, rather than relying on a single summary score or a single benchmark alone.

## 2.5 Methods for Improving Reliable RAG

Because hallucinations in RAG systems can arise from multiple stages of the pipeline, methods for improving reliability have also developed along multiple directions. Some approaches focus on improving retrieval so that the model receives more relevant and sufficient evidence in the first place, while others focus on how the generator interprets, integrates, or critiques retrieved information during answer generation. A further group of methods emphasizes detection and verification, aiming to identify unsupported claims or reduce hallucinations through post-generation analysis or corrective intervention [3, 17, 12].

### 2.5.1 Retrieval-oriented Improvements

Retrieval-oriented methods aim to improve the quality, relevance, and usefulness of the evidence that is passed to the generator. A basic direction in this area is to improve query formulation and retrieval diversity. For example, RAG-Fusion reformulates the original query into multiple related queries and then fuses the resulting rankings, with the goal of capturing

relevant evidence from different semantic perspectives rather than relying on a single retrieval query [38]. In a related direction, Auto-RAG treats retrieval as an iterative decision process in which the model plans successive retrieval steps and refines its information gathering over multiple turns, instead of assuming that one retrieval step is sufficient [39]. Active Retrieval Augmented Generation extends the retrieval process beyond a single initial search step by allowing the system to decide when additional retrieval is needed during generation. In particular, FLARE (Forward-Looking Active Retrieval) iteratively anticipates upcoming content, retrieves relevant documents for low-confidence continuations, and regenerates the sentence when necessary, thereby improving the integration of external knowledge in long-form generation [56].

Other methods focus on improving the retrieval unit or the post-retrieval selection process. LongRAG addresses limitations of short chunk-based retrieval by using longer retrieval units together with long-context language models, thereby preserving more contextual information and reducing the mismatch between retrieval granularity and answer generation [21]. LightRAG improves retrieval through a graph-based indexing approach and a dual-level retrieval strategy that supports both specific and more abstract information needs, showing how retrieval-oriented improvements can target not only relevance but also efficiency and adaptability in evolving knowledge sources [57]. CORAG (Cost-Constrained Retrieval Optimization for Retrieval-Augmented Generation) approaches retrieval as an optimization problem and selects chunk combinations under cost constraints, explicitly modeling correlations among chunks and the fact that adding more retrieved passages does not always improve performance [20]. Corrective Retrieval Augmented Generation (CRAG) further strengthens retrieval by introducing a retrieval evaluator that assesses the quality of retrieved documents and triggers corrective actions, including retrieval extension and selective filtering of irrelevant information [30]. Together, these methods illustrate that improving retrieval reliability involves not only finding relevant passages, but also shaping how evidence is gathered, combined, and filtered before generation.

### 2.5.2 Detection and Verification Frameworks

A third line of work improves reliable RAG by detecting, verifying, or correcting hallucinations after retrieval and generation have taken place. LettuceDetect is a dedicated hallucination detection framework for RAG applications that performs token-level classification over context-question-

answer triples, allowing unsupported claims to be identified efficiently even in relatively long contexts [16]. ReDeEP approaches the problem from a mechanistic perspective, analyzing how the model balances external context against parametric knowledge and detecting hallucinations when the model over-relies on internal knowledge instead of properly using retrieved evidence [54]. SelfCheckGPT provides a more general black-box alternative, using inconsistencies across multiple sampled outputs as a signal that a statement may be non-factual [5].

Some methods go beyond detection and incorporate correction into the pipeline. RAG-HAT (Hallucination Aware Tuning) uses hallucination detection outputs to construct preference data and then fine-tunes the model through hallucination-aware tuning, reducing hallucination rates while improving answer quality [58]. In addition, datasets and diagnostic resources such as RAGTruth support the development of these systems by providing manually annotated hallucination corpora tailored to retrieval-augmented settings [14]. Taken together, detection and verification frameworks provide an additional reliability layer for RAG systems by identifying unsupported content, enabling correction, and making hallucination-aware monitoring more practical in real-world deployments.

## 2.6 Related Work and Research Gap

Prior work on RAG can be broadly grouped into several closely related directions. A first line of research established the conceptual and architectural foundations of RAG, beginning with retrieve-then-generate formulations that combine parametric language models with external document retrieval, and later extending these foundations into more advanced and modular pipeline designs [6, 2, 31, 12]. A second line of research has focused on improving individual parts of the pipeline, including retrieval quality, context selection, conflict handling, adaptive retrieval, and generation-time self-reflection [30, 38, 39, 21, 41, 15, 59]. Together, these works show that RAG performance depends on coordinated pipeline design rather than on retrieval alone.

A substantial body of work has also addressed hallucination, trustworthiness, and reliability in retrieval-augmented systems. Reviews and surveys emphasize that RAG can reduce hallucinations but does not eliminate them, since failures may still arise from noisy retrieval, incomplete evidence, weak use of retrieved context, or conflicts between retrieved information and parametric knowledge [3, 4, 17]. Complementing these broader analyses, more specialized methods and resources such as RAGTruth,

LettuceDetect, ReDeEP, and SelfCheckGPT target hallucination detection, evidence-grounded diagnosis, and reliability monitoring at the output level [14, 16, 54, 5]. This work has made reliability a central concern in RAG research, but it has also highlighted that different studies often focus on different failure modes, metrics, and assumptions.

A third important direction concerns evaluation. Frameworks such as RAGAS treat retrieval quality, answer grounding, and generation quality as separate but interacting dimensions of RAG performance, while benchmarks such as RGB, UAEval4RAG (Unanswerability Evaluation for Retrieval-Augmented Generation), MultiHop-RAG, and RAGChecker expose more specific weaknesses related to noise robustness, rejection of unanswerable queries, multi-hop evidence integration, and fine-grained pipeline diagnostics [9, 52, 8, 37, 55]. These studies collectively show that no single benchmark or metric is sufficient for assessing reliable RAG behavior across all settings. Instead, evaluation must be aligned with the reliability properties that a system is expected to satisfy, especially when hallucination awareness and abstention behavior are important design goals.

Adjacent work has also compared RAG with fine-tuning as alternative strategies for incorporating new or specialized knowledge into LLMs. These studies suggest that while fine-tuning can be effective in some scenarios, retrieval-based approaches often provide greater flexibility and can be a more reliable choice for knowledge injection, particularly when the target knowledge is evolving, domain-specific, or infrequently represented in pretraining data [60, 61]. This comparison is relevant because it clarifies that improving knowledge access in LLM systems is not limited to a single method. However, the present thesis focuses specifically on RAG, since its goal is not to compare all possible knowledge-injection strategies, but to investigate how the design of a RAG pipeline affects reliability, transparency, and hallucination-aware behavior in a fixed-size knowledge base.

Against this background, the research gap addressed in this thesis can be stated as follows. Existing work provides strong foundations, numerous pipeline improvements, and an expanding set of evaluation methods, but there remains a need for system-level studies that examine how concrete RAG design choices interact in practice when the objective is to build a reliable and hallucination-aware system for a bounded knowledge source. Much of the literature either proposes individual methods, benchmarks particular capabilities, or studies RAG in broader survey form. Fewer works take a prototype-centered perspective that combines pipeline design, comparative configuration testing, and reliability-oriented evaluation within a single fixed

application setting. This thesis addresses that gap by designing and evaluating a prototype RAG system with explicit attention to retrieval quality, answer grounding, transparency, and hallucination-related behavior. In addition, it contributes an evaluation dataset generated for the bounded knowledge source, which can support systematic testing of the prototype and may also serve as a reusable basis for evaluating similar RAG systems in future work.

## 2.7 Summary

This chapter presented the background and related work relevant to reliable RAG. It introduced the role of LLMs in knowledge-intensive settings, explained the foundations and architectural variants of RAG, and reviewed the main reliability challenges that remain despite retrieval augmentation. The chapter also discussed key evaluation perspectives, including retrieval quality, hallucination-aware assessment, and benchmark-based evaluation, as well as methods proposed in prior work for improving reliable RAG.

The literature reviewed in this chapter shows that reliable RAG depends on the interaction of multiple pipeline components rather than on retrieval alone. It also shows that, although many methods and evaluation frameworks have been proposed, there is still a need for system-oriented studies that examine how practical design choices affect reliability, transparency, and hallucination-related behavior in bounded knowledge settings. This motivates the methodology and experimental design presented in the next chapter.

Table 2.1 summarizes the main research directions discussed in this chapter together with their main strengths and limitations. This overview helps clarify how prior work approaches reliability in RAG systems and highlights the aspects that are most relevant for the present thesis.

Table 2.1: Summary of major RAG research directions relevant to this thesis

| **Category** | **Main idea** | **Strengths** | **Limitations / relevance to this thesis** |
|---|---|---|---|
| Naive RAG | Retrieve a fixed set of passages and use them directly for generation | Simple and effective baseline | Sensitive to retrieval quality and weak context use |
| Advanced / modular RAG | Introduce additional modules for retrieval, reranking, context construction, or verification | More flexible and better suited to reliability-oriented design | More complex to configure and evaluate |
| Retrieval-oriented improvements | Improve query formulation, indexing, retrieval strategy, or chunk selection | Can improve evidence quality before generation | Good retrieval alone does not guarantee grounded answers |
| Generation-time improvements | Improve how the model interprets and uses retrieved evidence during generation | Can reduce unsupported or weakly grounded responses | Often adds inference complexity |
| Detection / verification frameworks | Detect hallucinations or verify whether outputs are supported by evidence | Adds a reliability layer for monitoring and correction | Typically complements rather than replaces good pipeline design |
| Evaluation frameworks | Measure retrieval, grounding, hallucination behavior, and unanswerability | Enables systematic comparison of RAG systems | Different frameworks emphasize different aspects of reliability |

# Chapter 3

# Methodology and Experimental Design

This chapter describes the methodology and experimental design of the thesis. It first presents the research approach and methodological choices underlying the study. It then outlines the research process, the design of the evaluation, and the collection of evaluation data. In addition, the chapter discusses issues of validity, reliability, reproducibility, and ethics, and explains the methods used for analyzing the results. The purpose of the chapter is to clarify how the study was structured and how the empirical investigation was conducted, thereby providing the methodological basis for the system development and evaluation presented in the subsequent chapters.

## 3.1 Research Approach and Methodology

This thesis adopts a design-oriented and experiment-based research approach. The work is design-oriented because it centers on the development and study of a prototype RAG system intended to address a concrete engineering problem, namely how to improve reliability and hallucination awareness in a bounded knowledge-base setting. It is experiment-based because the prototype is not only constructed, but also evaluated under multiple pipeline configurations in order to examine how different design choices influence retrieval quality, answer grounding, and hallucination-related behavior.

The methodological choice is motivated by the nature of the research question. Since the thesis investigates how a RAG system can be designed to reduce hallucinations while providing transparent and trustworthy answers, it is necessary to combine artifact construction with systematic empirical

evaluation. A purely theoretical study would not be sufficient, because the research problem concerns the behavior of an implemented system in practice. Similarly, a purely user-centered study would not be appropriate as the primary method, since the main object of investigation is the technical behavior of the RAG pipeline rather than user perception alone.

The study therefore combines two complementary methodological elements. The first is prototype development, through which a concrete RAG system is designed and configured. The second is comparative evaluation, through which different pipeline variants are tested in a controlled setting. Together, these elements make it possible to generate practical and analytically relevant knowledge about how reliability in RAG systems is affected by design choices across the pipeline. This approach is consistent with design science research, where knowledge is produced through the construction and evaluation of purposeful artifacts designed to address identified problems [62, 63].

## 3.2 Research Paradigm

The thesis is grounded in a pragmatic research paradigm. A pragmatic perspective is appropriate because the study is driven by a practical engineering problem and focuses on producing useful knowledge through the design and evaluation of a working system. Rather than aiming primarily at theory development, the thesis seeks to understand which technical design choices contribute to more reliable behavior in a RAG pipeline operating on a fixed-size knowledge base.

At the same time, the study has an empirical orientation. Knowledge is generated through observation of system behavior under different experimental conditions, and conclusions are drawn from comparative analysis of the resulting outputs and evaluation measures. In this sense, the thesis treats the prototype RAG system as an artifact through which the research question can be investigated in a systematic and evidence-based manner.

This paradigm is well suited to the present work because it allows the thesis to combine engineering construction with analytical evaluation. The focus is not only on building a functioning system, but also on using that system to study reliability-related properties such as retrieval quality, answer grounding, and hallucination-aware behavior. The research paradigm therefore supports both the practical and the investigative aims of the thesis. This is also aligned with design science research methodology, which emphasizes both artifact creation and evaluation as central activities in producing practically relevant

and empirically grounded knowledge [63].

### 3.2.1 Research Design

The research design of this thesis is an artifact-centered comparative study. The central artifact is a prototype RAG system developed for a fixed-size knowledge base, and the study is structured around evaluating how different pipeline configurations affect reliability-related behavior. Rather than treating the system as a single static solution, the design of the study allows multiple configurations to be examined and compared under controlled conditions.

This comparative structure is important because the research question concerns how design choices influence system behavior. The study therefore does not focus only on whether the prototype can generate answers, but on how different choices in retrieval, context construction, prompting, and answer handling shape retrieval quality, groundedness, and hallucination-aware performance. In this sense, the research design combines constructive system development with empirical comparison.

The study is also bounded in scope. It is conducted on a fixed-size knowledge base and a deliberately constructed evaluation query set, which makes it possible to investigate reliability in a controlled and well-defined setting. This bounded design supports more systematic comparison than would be possible in a broad open-domain setting, while still allowing meaningful observations about the interaction between pipeline components and reliability-related outcomes.

### 3.2.2 Rationale for Methodological Choices

The chosen methodology is motivated by the objectives of the thesis. Since the work aims to investigate how a RAG system can be designed to reduce hallucinations and provide more trustworthy answers, it is necessary to study the behavior of a concrete implemented system rather than rely solely on theoretical reasoning. A design-oriented and experiment-based methodology makes it possible to examine how reliability emerges from the interaction of multiple pipeline components in practice.

A comparative prototype study was selected because it allows the thesis to move beyond the evaluation of a single end result. By comparing different pipeline configurations, the study can identify how specific design choices affect retrieval quality, answer grounding, and hallucination-related behavior. This is more informative for the research question than a simple demonstration

of one working system, since the goal is not only to build a prototype but also to understand which design decisions contribute to more reliable behavior.

Other methodological alternatives were considered less suitable. A purely theoretical or conceptual study would not have provided empirical evidence about system behavior. A user study focused on perceived trust or usability would have addressed a different research problem, since the primary focus of this thesis is technical reliability rather than user experience. Similarly, a production deployment study was outside the scope of the degree project due to the time and resource constraints of the thesis. The selected methodology therefore offers an appropriate balance between engineering feasibility, empirical rigor, and relevance to the research question.

## 3.3 Research Process

The research process in this thesis combined literature-informed system design with iterative experimental evaluation. The work began with the identification of the research problem, namely how a RAG system can be designed to behave more reliably in a fixed-size knowledge-base setting. This was followed by a literature review covering foundational RAG architectures, hallucination and reliability in LLMs, evaluation frameworks, and methods for improving reliable RAG. The review served as the basis for selecting the main design dimensions and evaluation perspectives used in the study.

Based on this foundation, a prototype RAG system was developed as the central artifact of the thesis. The purpose of the prototype was not only to demonstrate a working system, but also to provide a controlled basis for comparative evaluation. In parallel with the system development, an evaluation strategy was defined in order to examine how different pipeline configurations affected retrieval quality, answer behavior, and hallucination-related performance. This included the construction of a dedicated query set designed to test multiple aspects of RAG reliability in a bounded knowledge setting.

The empirical part of the study was then carried out through controlled experiments in which multiple pipeline configurations were tested on the constructed evaluation data. The results were collected and analyzed with respect to the evaluation criteria defined earlier in the chapter. This made it possible to compare the strengths and weaknesses of different configurations and to examine how reliability-related behavior emerged from the interaction of multiple pipeline components rather than from a single design choice in isolation.

The overall workflow of the thesis can be summarized in the following stages:

**Step 1** define the research problem and scope of the study,

**Step 2** review relevant literature and identify key design and evaluation dimensions,

**Step 3** design and develop a prototype RAG system,

**Step 4** generate the evaluation dataset by constructing a query set for the selected knowledge source and defining the experimental setup,

**Step 5** conduct controlled experiments on multiple pipeline configurations, and

**Step 6** analyze and interpret the results in relation to reliability and hallucination awareness.

## 3.4 Evaluation Design

The evaluation in this thesis is designed as a controlled comparative study of three RAG pipeline configurations. The purpose of the evaluation is not only to assess whether the prototype can generate answers, but to examine how different retrieval and pipeline design choices influence reliability-related behavior. In particular, the evaluation is structured to compare how the tested configurations affect retrieval effectiveness, answer quality, and hallucination-aware performance within a fixed-size knowledge-base setting.

The compared pipeline configurations are illustrated at a high level in Figure 3.1. Their detailed implementation is described in the following chapter, while the present section focuses on the logic of the comparison, the dimensions being varied, and the criteria used for assessment.

### 3.4.1 Evaluation Objectives

The main objective of the evaluation is to investigate how different pipeline configurations influence the reliability of a RAG system. More specifically, the evaluation is intended to examine whether retrieval improvements and more adaptive retrieval behavior lead to better evidence selection and more dependable final answers than a simpler baseline configuration.

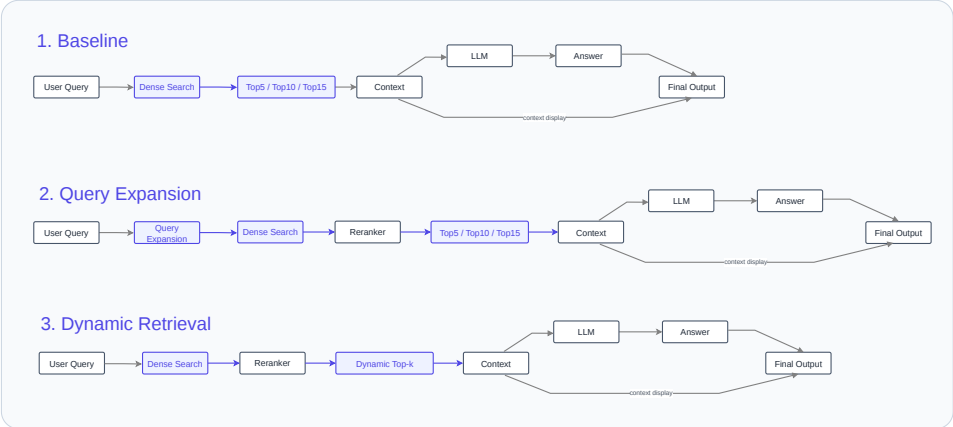


Figure 3.1: Overview of the three RAG pipeline configurations compared in the evaluation.

A second objective is to study how retrieval depth affects system behavior. Since the amount of retrieved context can influence both answer support and noise exposure, the evaluation compares different retrieval sizes in order to examine the trade-off between providing more evidence and introducing potentially distracting or less relevant context.

A third objective is to assess whether more advanced pipeline variants improve hallucination-aware behavior. In this thesis, reliability is understood not only in terms of retrieving relevant information, but also in terms of supporting answer generation in a way that reduces unsupported or misleading outputs. The evaluation is therefore designed to compare configurations with respect to both retrieval-side and answer-side behavior.

### 3.4.2 Compared Dimensions

The evaluation compares three pipeline configurations. The first configuration serves as a baseline and represents a relatively standard retrieve-generate pipeline. In this setting, dense retrieval is used as the primary retrieval strategy and the number of retrieved passages is varied using fixed top-$k$ values of 5, 10, and 15. This configuration provides the reference point against which the more advanced variants are assessed.

The second configuration extends the baseline by introducing a query expansion step before retrieval. The purpose of this variation is to examine whether reformulating or expanding the user query improves the relevance and usefulness of the retrieved evidence. As in the baseline configuration,

retrieval is evaluated at fixed top-$k$ values of 5, 10, and 15 in order to support direct comparison between the two settings.

The third configuration introduces dynamic retrieval. Instead of relying on a fixed retrieval depth, this pipeline adaptively determines the amount of retrieved context used for generation. The purpose of this variant is to investigate whether a more adaptive retrieval strategy can improve the robustness of the system relative to the other two configurations.

In addition to these pipeline-level variations, all three configurations are evaluated using three different embedding models: `BAAI/bge-large-en-v1.5` [64], `Qwen/Qwen3-Embedding-0.6B` [65], and `sentence-transformers/all-mpnet-base-v2` [34]. These models are applied during the indexing phase and therefore influence how document chunks are represented and retrieved. The models were selected to provide variation in model family, scale, and intended retrieval use. BGE represents a retrieval-oriented embedding model family commonly used for dense retrieval, Qwen3-Embedding provides a more recent general-purpose embedding model from the Qwen model family, and all-mpnet-base-v2 serves as a widely used Sentence-Transformers baseline. This selection is not intended to exhaustively benchmark all available embedding models, but to examine whether retrieval and generation behavior remain stable across different embedding representations. This setup makes it possible to analyze not only how pipeline design affects system behavior, but also how embedding model choice interacts with retrieval performance and downstream answer generation.

Taken together, these experimental factors make it possible to compare a standard baseline, a retrieval-enhanced variant based on query reformulation, and a more adaptive reliability-oriented pipeline across multiple retrieval depths and embedding representations. This design supports analysis of how indexing choices and pipeline design jointly affect the overall behavior of the system.

### 3.4.3 Metrics and Assessment Criteria

The compared configurations are assessed using a combination of retrieval-oriented and answer-oriented metrics. Since the experiments vary both the pipeline configuration and the embedding model used during indexing, the evaluation is designed to capture not only differences in answer behavior, but also differences in retrieval effectiveness that may arise from alternative document representations.

Retrieval performance is evaluated using standard ranking-based measures, including recall at $k$, precision at $k$, hit rate at $k$, and mean reciprocal rank. Such retrieval-focused measures are commonly used to assess whether a RAG system can identify and rank relevant evidence before answer generation, which is a necessary condition for reliable downstream responses [37, 56, 21, 6]. These metrics are used to examine whether the system retrieves relevant evidence, how highly such evidence is ranked, and how consistently the retriever provides useful support for downstream answer generation. In addition, the mean number of retrieved passages is recorded in order to compare fixed top-$k$ settings with the dynamic retrieval configuration.

At the answer level, the evaluation includes metrics for answer correctness, correctness rate, faithfulness, and hallucination rate. Answer correctness is used to assess the overall quality of generated responses, while correctness rate provides a more direct indication of how often the system produces acceptable answers across the evaluation set. Faithfulness is included to examine whether generated answers remain supported by the retrieved context, as faithfulness or groundedness is commonly used in RAG evaluation to assess whether generated responses are justified by the provided evidence [53, 54]. Hallucination rate is used to capture the extent to which the system produces unsupported or misleading outputs despite retrieval augmentation.

Taken together, these metrics allow the evaluation to consider both retrieval effectiveness and answer reliability. This is important because a configuration may retrieve relevant documents without producing grounded answers, or may produce plausible answers despite weak retrieval. The assessment is therefore based on multiple complementary measures rather than on a single aggregate score.

## 3.5 Data Collection and Evaluation Data

This section describes the data used in the evaluation. It defines the scope of the knowledge base, explains the construction of the evaluation query set, and clarifies how the data was organized to support the analysis of retrieval quality, answer behavior, and hallucination-aware performance.

### 3.5.1 Knowledge Base Scope

The evaluation in this thesis is conducted on a bounded knowledge base rather than in an open-domain setting. The bounded knowledge base used in the evaluation consists of 60 PDF documents selected for the prototype

system. This means that the prototype system retrieves only from a predefined document collection selected for the purpose of the study. The scope of the knowledge base is defined by the document collection included in the prototype at indexing time. In this setting, system performance depends not only on the retrieval and generation components, but also on the extent, structure, and coverage of the available source material. A bounded knowledge base is therefore useful for evaluation because it makes it possible to distinguish between cases where the answer should in principle be recoverable from the indexed collection and cases where the correct behavior is to acknowledge that the available evidence is insufficient.

This delimitation also supports more meaningful analysis of answerable and unanswerable queries. Because the retrieval source is fixed, the study can evaluate whether errors arise from retrieval failure, weak evidence use, or generation behavior rather than from uncontrolled variation in external information sources. At the same time, the results must be interpreted in relation to this bounded setting, since the aim is not to evaluate general open-domain question answering, but to study reliability-oriented RAG behavior under explicitly defined knowledge conditions.

A more detailed description of the documents included in the knowledge base and their preparation for retrieval is provided in the following chapter, where the implementation of the prototype system is described.

### 3.5.2 Query Dataset Construction

The evaluation is based on a query dataset constructed specifically for the bounded knowledge-base setting of the prototype. Existing benchmark datasets were not used because their questions and reference corpora are tied to external document collections that do not correspond to the 60 PDFs indexed in this study. Using such benchmarks would therefore make it difficult to determine whether an unsupported answer resulted from a system failure or from the absence of the required information in the prototype's knowledge base. A custom dataset was instead created to align the evaluation queries with the actual indexed documents and to support controlled testing of answerable, ambiguous, and unanswerable cases.

The query dataset was generated from the chunks stored in the vector database. For each source document, the corresponding indexed chunks were extracted and used as input to a large language model, which generated questions in three categories: answerable, ambiguous, and unanswerable. These categories were defined with respect to the bounded knowledge base.

A question was considered answerable when the indexed corpus contained sufficient evidence for a grounded answer, ambiguous when the corpus contained related or partial evidence, and unanswerable when the corpus did not contain sufficient information to answer the question.

For answerable and ambiguous questions, supporting chunk identifiers were stored where applicable. These references were later used to compute retrieval-oriented metrics such as recall at $k$, precision at $k$, hit rate at $k$, and mean reciprocal rank. For unanswerable questions, no reference chunk identifiers were assigned, since these questions were intended to evaluate whether the system avoided unsupported answer generation when sufficient evidence was absent.

The final dataset consisted of 200 questions: 100 answerable, 50 ambiguous, and 50 unanswerable. Table 3.1 summarises the composition of the dataset and the role of each category in the evaluation.

Table 3.1: Composition of the evaluation query dataset

| **Query category** | **Number** | **Purpose in the evaluation** |
|---|---|---|
| Answerable | 100 | Evaluates whether the system can retrieve relevant evidence and generate correct, faithful answers when the answer exists in the knowledge base. |
| Ambiguous | 50 | Evaluates how the system behaves when the corpus contains related or partially supporting evidence, but not enough information for a definitive answer. |
| Unanswerable | 50 | Evaluates whether the system avoids unsupported generation when the answer is not present in the knowledge base. |
| Total | 200 | Complete evaluation query set. |

Examples of queries from each category are provided in Appendix A to clarify how answerable, ambiguous, and unanswerable cases were distinguished.

## 3.6 Validity, Reliability, and Ethical Considerations

This section discusses the validity and reliability of the study and outlines the ethical considerations relevant to the development and evaluation of the proposed RAG system.

### 3.6.1 Validity

The validity of the study depends on whether the chosen evaluation design is appropriate for the research question. Since the thesis investigates how different RAG pipeline configurations influence reliability-related behavior, the evaluation must capture properties that are directly relevant to that objective. For this reason, the study does not rely on a single overall performance score, but instead evaluates retrieval quality, answer correctness, faithfulness, and hallucination-related behavior separately. This supports stronger alignment between the research question and the empirical assessment.

A further aspect of validity concerns the bounded knowledge-base setting used in the experiments. The fixed-size document collection makes it possible to define answerable, ambiguous, and unanswerable cases in a controlled way, which strengthens the internal validity of the study. At the same time, this also limits the scope of the conclusions. The results should therefore be interpreted as findings about reliability-oriented RAG behavior in a bounded knowledge setting rather than as claims about open-domain question answering in general.

The manually constructed query set also affects validity. Because the evaluation queries were designed specifically for the thesis objectives, they make it possible to test behavior that is directly relevant to reliability and hallucination awareness. However, this also means that the usefulness of the evaluation depends on the quality and balance of the constructed cases. To address this, the query set is structured to include different knowledge conditions rather than only straightforward answerable questions.

### 3.6.2 Reliability and Reproducibility

Reliability in this study concerns the consistency of the evaluation procedure and the extent to which the experiments can be repeated under the same conditions. To support reliable comparison, all tested pipeline configurations are evaluated on the same knowledge base and the same query set. This

ensures that differences in performance can be interpreted in relation to the experimental factors under study rather than to variation in the evaluation material.

Reproducibility is further supported by the use of explicitly defined pipeline configurations, fixed retrieval settings where applicable, and a documented experimental procedure. The comparison across embedding models, retrieval settings, and pipeline variants is therefore carried out within a controlled framework intended to make the evaluation traceable and repeatable. In addition, the use of multiple complementary metrics reduces dependence on any single measurement and provides a more stable basis for interpretation.

At the same time, some limitations remain. As with many experiments involving LLMs, some outputs may vary depending on model behavior and inference conditions. The results should therefore be understood as observations obtained under a specific experimental setup. Reliability in this context is strengthened not by assuming perfect determinism, but by keeping the evaluation conditions consistent and by interpreting results across multiple dimensions rather than in terms of a single run or metric.

### 3.6.3 Ethical Considerations

Although the thesis does not involve human-subject experimentation, it still raises ethical considerations related to the design and evaluation of AI systems. A central issue is the risk of producing unsupported or misleading answers. Since RAG systems may still hallucinate despite retrieval augmentation, evaluating reliability and hallucination-aware behavior is itself part of addressing an ethical concern: reducing the likelihood that users are presented with confident but insufficiently supported information.

Another ethical consideration concerns transparency and responsible system behavior. In bounded knowledge settings, a reliable system should not only answer correctly when evidence exists, but also avoid overstating what can be concluded when the available information is incomplete or insufficient. This is particularly important in ambiguous and unanswerable cases, where overconfident generation may create a false impression of certainty.

Finally, the work is conducted with awareness that design decisions in RAG systems can influence user trust. Methods that improve grounding, faithfulness, and hallucination detection are therefore relevant not only from a technical perspective but also from an ethical one. The goal of the thesis is not to claim that the developed prototype eliminates these risks entirely, but

to contribute to a more responsible understanding of how RAG systems can be designed and evaluated for greater reliability.

## 3.7 Data Analysis Methods

The data analysis in this thesis is based on comparative evaluation across the tested pipeline configurations, retrieval settings, and embedding models. The purpose of the analysis is to examine how these experimental factors influence retrieval effectiveness, answer quality, and hallucination-aware behavior within the bounded knowledge-base setting. Rather than relying on a single summary measure, the analysis considers multiple complementary metrics in order to capture different aspects of system performance.

The analysis is conducted in two main stages. First, retrieval-oriented metrics are used to compare how effectively the tested configurations identify and rank relevant evidence. This part of the analysis focuses on the extent to which the system retrieves useful supporting material for downstream answer generation. Second, answer-oriented metrics are used to examine correctness, faithfulness, and hallucination-related behavior in the generated responses. This two-level analysis makes it possible to distinguish between failures caused primarily by retrieval weaknesses and failures that arise at the answer generation stage.

The results are analyzed comparatively across the different configurations rather than in isolation. In particular, the analysis examines how the baseline, query expansion, and dynamic retrieval pipelines differ with respect to the selected metrics, and how these effects interact with retrieval depth and embedding model choice. The aim is not to identify a single strongest overall configuration, but rather to analyze how different configurations influence system behavior and to understand why the observed results arise across retrieval quality, answer support, and hallucination-aware performance.

The interpretation of the results is therefore guided by the research question and by the reliability-oriented focus of the thesis. Configurations are assessed in terms of whether they improve the grounding and dependability of the generated answers, whether they reduce unsupported outputs, and whether they do so consistently across different evaluation cases. This analytical approach supports a more nuanced understanding of reliable RAG design than would be possible from a single aggregate performance score alone.

# Chapter 4

# System Design and Implementation

This chapter presents the design and implementation of the RAG system developed in this thesis. The purpose of the chapter is to describe the concrete system artifact, the main components of the implemented pipeline, and the design decisions made to support reliability and hallucination awareness. While the previous chapters established the theoretical and methodological foundation for the work, this chapter focuses on how these ideas were translated into an operational prototype.

The implemented system follows a modular RAG architecture consisting of document processing, chunking, embedding, vector-based retrieval, prompt-based generation, and post-generation reliability assessment. Each component was designed so that different pipeline configurations could be tested and compared in later experiments. This modular structure makes it possible to isolate the effects of individual design choices, such as retrieval settings, prompt structure, context handling, and hallucination-awareness mechanisms.

A central objective of the implementation was to move beyond a basic retrieve-then-generate pipeline by incorporating mechanisms that make the system aware of potential unsupported or insufficiently grounded answers. For this reason, the chapter also describes how retrieval confidence, answer grounding, and abstention behavior were considered in the system design. These mechanisms are intended to support the broader thesis goal of building a RAG system that not only produces answers, but also provides signals about their reliability.

In addition to the system itself, this chapter describes the construction

of the query dataset used for evaluation. The dataset was designed to test answerable, ambiguous, and unanswerable queries, making it possible to evaluate how the system behaves when sufficient supporting evidence is available and when it is not. The chapter therefore documents both the implemented RAG prototype and the evaluation material required to assess it systematically.

## 4.1 System Requirements and Design Goals

The system developed in this thesis was designed as a prototype for evaluating how RAG can be used to improve the reliability of large language model outputs while making potential hallucinations more visible. The purpose of the implementation was therefore not only to build a functional question-answering system, but also to create a configurable experimental environment in which different RAG pipeline configurations could be compared. The system requirements were derived from the overall aim of the thesis: to investigate how a RAG system can be designed with attention to answer grounding, retrieval quality, and hallucination awareness.

A first requirement was that the system should generate answers grounded in an external knowledge base. Instead of relying only on the parametric knowledge of the language model, the system had to retrieve relevant document passages and use them as the primary basis for answer generation. This requirement follows from the central motivation behind RAG, namely that external retrieval can provide access to task-specific, updatable, and inspectable knowledge that is not necessarily encoded in the language model itself. However, prior work also shows that the use of retrieval does not eliminate hallucination entirely, since generated answers may still be unsupported, inconsistent with the retrieved context, or affected by irrelevant or incomplete retrieval results [9, 14, 3, 18]. This motivated the need to treat retrieval grounding as a design requirement rather than as an automatic guarantee of reliability.

A second requirement was transparency. The system should make the retrieved context available for inspection, either directly or through stored metadata such as document identifiers, chunk identifiers, and source references. This was necessary because the evaluation of a RAG system depends not only on the final generated answer, but also on whether the retrieved evidence was relevant and sufficient. By preserving the connection between user queries, retrieved chunks, and generated answers, the system enables later analysis of whether failures originate from retrieval, generation,

or the interaction between the two.

A third requirement was configurability. Since the thesis evaluates the behavior of different RAG pipeline configurations, the implementation had to support controlled variation of key components and parameters. These include the document processing strategy, chunking parameters, embedding model, number of retrieved chunks, prompt design, and hallucination-awareness mechanisms. A modular design was therefore chosen so that individual components could be modified or replaced without changing the entire system. This makes it possible to compare a baseline language model, a standard RAG pipeline, and additional reliability-oriented variants under similar experimental conditions.

A fourth requirement was hallucination awareness. In this thesis, this refers to the system's ability to signal whether an answer is sufficiently supported by the retrieved context. This does not imply perfect hallucination detection, but rather mechanisms for identifying weak retrieval, insufficient context, or unsupported claims. Such mechanisms can include retrieval confidence thresholds, context sufficiency checks, faithfulness evaluation, and abstention when the available evidence is inadequate. This goal aligns with recent RAG evaluation work, which emphasizes that reliable systems must assess whether generated outputs are grounded in retrieved evidence, not only retrieve and generate text.

A fifth requirement was reproducibility. The implemented system had to support systematic experimentation using a fixed knowledge base, a defined query dataset, documented pipeline configurations, and consistent evaluation procedures. This requirement is important because the thesis aims to compare system behavior across different configurations rather than demonstrate a single isolated implementation. Therefore, inputs, retrieved contexts, generated answers, and evaluation outputs were stored in a structured way to support later analysis.

The resulting system requirements and design goals are summarized in Table 4.1. Together, they guided the implementation toward a modular, traceable, and reproducible RAG prototype with explicit support for grounded generation and hallucination-aware behaviour.

## 4.2 System Architecture

The implemented system follows a modular RAG architecture consisting of three main phases: offline indexing, online query processing and generation, and evaluation with hallucination-awareness components. The architecture

Table 4.1: System requirements and corresponding design goals

| **Requirement** | **Design goal** |
|---|---|
| Grounded generation | Retrieve relevant passages from the knowledge base and use them as the primary context for answer generation. |
| Transparency | Preserve links between queries, retrieved chunks, source metadata, and generated answers. |
| Configurability | Implement the system as a modular pipeline in which retrieval, generation, and evaluation components can be varied. |
| Hallucination awareness | Include mechanisms for identifying weak retrieval, insufficient context, or unsupported generated claims. |
| Reproducibility | Store inputs, configurations, retrieved evidence, generated answers, and evaluation outputs in a structured format. |

was designed to transform a collection of PDF documents into a searchable knowledge base, retrieve relevant information for a user query, generate an answer using a large language model, and evaluate the reliability of the generated response.

The system begins with a document collection consisting of 60 PDF documents stored in Azure Blob Storage. These documents form the external knowledge base used by the RAG pipeline. Before they can be used for retrieval, the documents are processed and converted into machine-readable representations. The Unstructured library is used to parse the PDF files and extract different content types, including text, tables, and images. Textual content is preserved as plain text, tables are represented using Markdown formatting, and images are represented through OCR-extracted text. Although a multimodal language model could potentially be used to produce richer summaries of images, only OCR-based extraction was applied in this implementation due to resource limitations.

After parsing, the extracted content is divided into smaller chunks suitable for embedding and retrieval. These chunks are encoded using embedding models and stored in a Qdrant vector database. Qdrant was deployed locally through Docker, which provided a controlled and reproducible environment for indexing and retrieval during development. Each stored chunk is associated with its vector representation and relevant metadata, allowing retrieved results to be traced back to their original document source.

When a user submits a query, the query is embedded using the same

embedding model as the indexed document chunks. The embedded query is then compared against the stored document embeddings using cosine similarity. The most relevant chunks are retrieved from Qdrant and inserted into a prompt together with the user query. This prompt is passed to a large language model, which generates an answer conditioned on the retrieved context.

In addition to the core RAG pipeline, the system includes components for evaluation and hallucination awareness. RAGAS is used to evaluate the faithfulness of generated answers to the retrieved evidence. An additional natural language inference (NLI) layer was also implemented as a hallucination-detection mechanism. This layer was independently tested within the pipeline, but it was not included in the controlled experiments using the query dataset. The reason for this separation was to keep the main experiments focused on comparable RAG configurations and to avoid introducing an additional post-processing component that could make the effects of retrieval and generation design choices harder to isolate.

Overall, the system architecture separates the construction of the knowledge base from the query-time generation process and from the evaluation layer. This separation supports modularity, reproducibility, and controlled experimentation. It also makes it possible to analyze different failure points in the RAG pipeline, including document parsing errors, retrieval failures, insufficient context, unsupported generation, and limitations in automatic evaluation. Figure 4.1 presents an overview of the implemented system architecture and its main components.

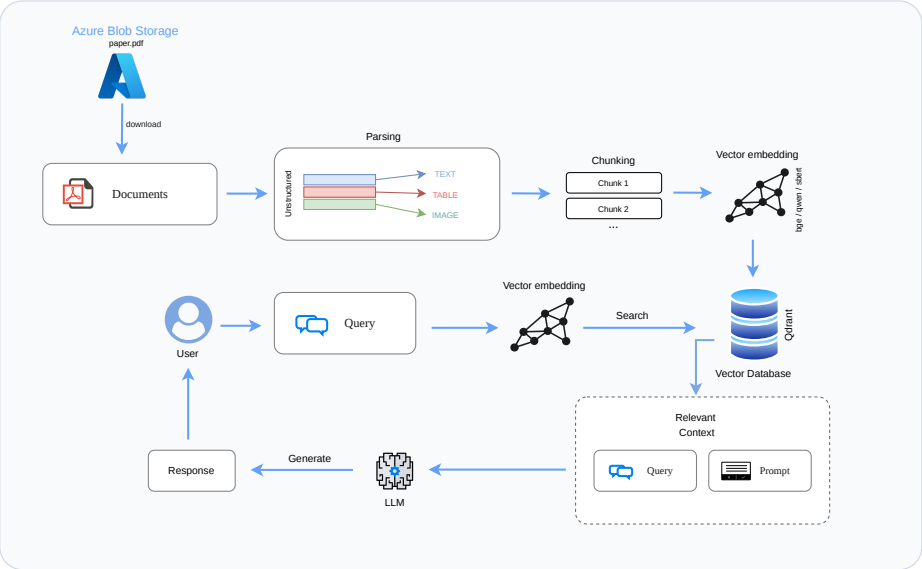


Figure 4.1: System architecture diagram.

### 4.2.1 Offline Indexing Pipeline

The offline indexing pipeline converts the document collection into a searchable vector-based knowledge base. This stage is executed before any user query is processed and consists of five main steps: document retrieval from cloud storage, PDF parsing, normalization of extracted elements, chunk aggregation and filtering, and vector indexing. The purpose of this pipeline is to transform the original research papers into retrieval units that can later be used as grounded context for answer generation.

The input corpus consisted of approximately 60 research papers stored as PDF files in Azure Blob Storage. Azure Blob Storage was used as the document storage layer because it provides cloud-based object storage for unstructured data, including text and binary files such as PDFs [66]. During ingestion, the system listed the available blobs in the configured container, downloaded each PDF to a local directory, and assigned each document a unique document identifier. These identifiers were preserved throughout the later processing steps so that generated chunks could be traced back to their original source documents. Figure 4.2 provides an overview of this offline indexing process and shows how raw PDF documents are transformed into indexed vector representations.

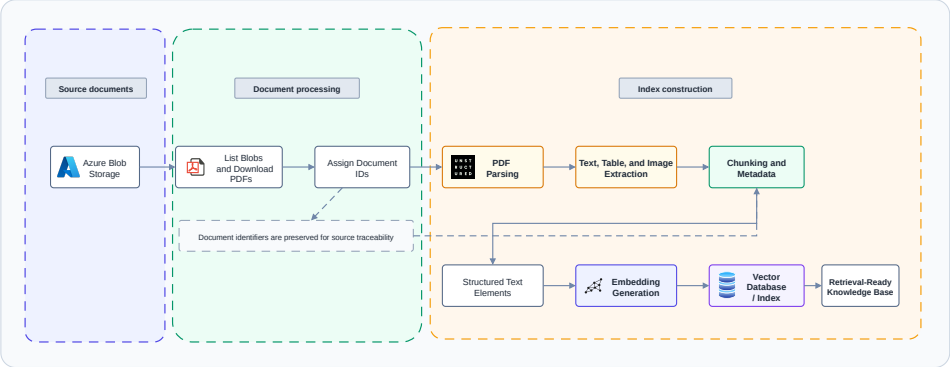


Figure 4.2: Offline indexing pipeline diagram.

After downloading, each PDF was parsed using the Unstructured library. Unstructured provides partitioning functions that convert raw documents into structured elements, such as titles, narrative text, lists, tables, and other document components [67]. In this implementation, the PDF parser was configured to use a high-resolution parsing strategy when automatic OCR handling was selected. Table structure inference was enabled so that tables could be extracted as structured content rather than being treated only as plain text. This is important for research papers, where tables often contain compact experimental results, model comparisons, parameter settings, or evaluation metrics. Unstructured supports table extraction for PDFs through the `infer_table_structure` option, which is designed to preserve table structure during parsing [68].

The extracted elements were then normalized into a common internal representation. Textual elements were stored as text chunks, table elements were stored using their Markdown representation, and image or figure elements were represented through OCR-derived text. This representation allowed different content modalities to be included in the same retrieval pipeline while still preserving their original type. For example, table chunks could later be distinguished from ordinary text chunks, and image-derived chunks could be identified as originating from visual content. Although a multimodal language model could potentially have been used to generate richer semantic summaries of figures and images, this implementation only applied OCR-based extraction for images. This choice was made because the controlled prototype had limited computational and development resources, and because OCR provided a simpler and more reproducible way to include image-derived textual information in the knowledge base.

The normalized chunks were then aggregated into retrieval-sized units.

Since PDF parsing often produces small fragments, adjacent text chunks were merged into larger chunks before indexing. The implemented aggregation strategy only merged textual chunks and preserved non-text chunks, such as tables and image-derived chunks, as separate retrieval units. Text chunks were not merged across different section titles, document identifiers, or ingestion versions. This constraint was used to preserve document structure and avoid combining unrelated content from different sections. The default chunking strategy was recursive character-based splitting, where larger text blocks were split using progressively smaller separators such as paragraph breaks, line breaks, spaces, and finally fixed character boundaries. The target chunk size was set to 1500 characters with an overlap of 200 characters. The overlap was used to reduce the risk that relevant information would be separated across chunk boundaries.

Because the source documents were research papers in PDF format, an additional chunk quality filtering step was included. PDF extraction can produce low-value artifacts such as page numbers, isolated years, arXiv identifiers, headers, footers, figure-only captions, citation-heavy fragments, or sections that are not useful for question answering, such as references and acknowledgements. To reduce the amount of noisy content in the vector database, the system applied deterministic filters and a simple quality score before retaining chunks for indexing. Text chunks below a minimum length were removed, and chunks from predefined low-value sections such as references, bibliography, acknowledgements, checklists, and ethics statements were excluded. Additional heuristics considered the ratio of alphabetic characters, word count, symbol density, unusual word length, and citation density. Tables and image-derived chunks were handled separately so that structured or visual information was not removed only because it did not resemble ordinary prose.

After aggregation and filtering, the retained chunks were embedded using one of three selected embedding models: `BAAI/bge-large-en-v1.5` [64], `Qwen/Qwen3-Embedding-0.6B` [65], and `sentence-transformers/all-mpnet-base-v2` [34]. These models were chosen to support comparison between different retrieval configurations while keeping the rest of the indexing pipeline unchanged. Each chunk was converted into a dense vector representation and stored in a Qdrant collection together with its metadata. Qdrant was used as the vector database because it is designed for storing, searching, and managing vectors with associated payloads, which makes it suitable for semantic retrieval in RAG systems [69]. In this system, the payload stored with each vector included metadata such as the document

identifier, chunk identifier, modality, page numbers, and section information where available. This metadata was necessary for traceability, since retrieved chunks needed to be linked back to the original document context during evaluation and error analysis.

The Qdrant database was run locally through Docker. Docker containers provide isolated runtime environments containing the files and dependencies needed to run an application, which supports reproducible deployment across environments [70]. Running Qdrant locally in a Docker container made the indexing and retrieval setup easier to reproduce during development and avoided dependence on an external hosted vector database. The resulting vector collection served as the knowledge base used by the online query pipeline described in the next subsection.

### 4.2.2 Online Query and Generation Pipeline

The online query and generation pipeline is executed when a user submits a question to the system. While the offline indexing pipeline prepares the document collection for retrieval, the online pipeline uses the indexed knowledge base to identify relevant chunks and generate an answer grounded in those chunks. The main steps are query processing, optional query expansion, dense retrieval, optional reranking, context construction, and answer generation. Figure 4.3 provides an overview of this query-time pipeline and shows how a user query is transformed into retrieved context and then into a generated answer.

When a query is submitted, the system first normalizes and embeds the query using the same embedding model that was used to embed the document chunks during indexing. This is necessary because dense retrieval depends on comparing the query vector and document vectors in the same embedding space. The embedded query is then sent to the Qdrant vector database, where the system performs similarity search over the indexed chunk embeddings. Qdrant supports nearest-neighbor vector search and can use cosine similarity as a distance metric for comparing vectors [71]. In this implementation, cosine similarity was used to retrieve the chunks that were semantically closest to the user query.

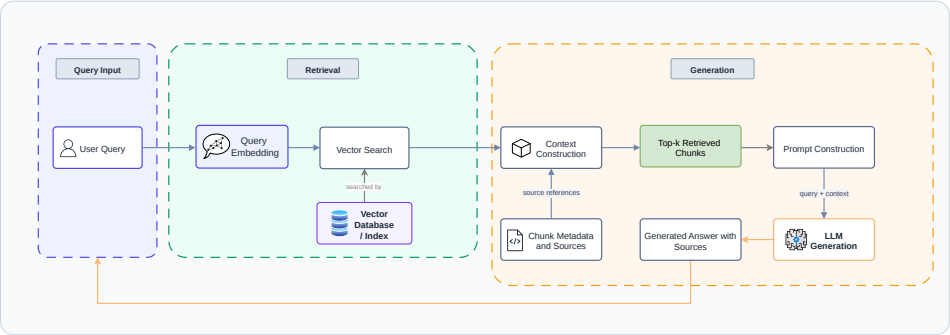


Figure 4.3: Online query and generation pipeline diagram.

The default retrieval setup follows a dense retrieval approach. For each query, the system retrieves the top-ranked chunks from the Qdrant collection and returns both the chunk content and its associated metadata, including document identifier, chunk index, page number, modality, and section title when available. The retrieval output therefore contains not only the text used for answer generation, but also provenance information that can later be used for citation, inspection, and evaluation. The retrieved chunks may originate from ordinary text, Markdown-formatted tables, or OCR-derived image text, depending on which indexed content is most relevant to the query.

In addition to basic dense retrieval, the system was designed to support query expansion. In this mode, the original user query is expanded into several related search queries using `gpt-4.1-mini`. The expanded queries are embedded separately, and each query is used to search the vector database. The retrieved results are first fused using Reciprocal Rank Fusion (RRF), which assigns each retrieved chunk a score based on its rank position across the expanded-query result lists. Chunks that appear highly ranked in multiple result lists receive higher fused scores, while duplicate chunks are merged under the same identifier. The resulting RRF-ranked list forms an expanded candidate set that can be passed to the later reranking and context construction stages. This approach was included to improve recall for queries that may be expressed differently from the terminology used in the source documents. For example, a user query may refer to a concept using a general phrase, while the documents may use a more specific technical term. Query expansion can help bridge this gap by retrieving evidence from multiple semantic formulations of the same information need.

The query expansion step used a fixed prompt template. The purpose of this prompt was to generate alternative formulations of the original query that

preserved the same information need while increasing the chance of retrieving relevant passages expressed with different technical terminology. The model was instructed to return a JSON array only, which made the output easier to parse automatically. The prompt template used for query expansion is shown in Listing 4.1.

Listing 4.1: Prompt template for query expansion

```
Generate 2 alternative search queries for retrieving
technical documents.

Requirements:
- Preserve the exact meaning and scope of the original
   query.
- Do not make the query broader, narrower, or more
   speculative.
- Keep all named entities, method names, benchmark
   names, dataset names, acronyms, and technical terms
   exactly as written.
- Do not introduce new methods, datasets, model types,
   domains, mechanisms, or assumptions that are not
   implied by the original query.
- Use concise search-style phrasing, not long
   explanatory questions.
- Prefer terms that are likely to appear in technical
   reports or research papers.
- Each query should be lexically different, but still
   answer-seeking for the same fact.
- If the original query is already specific, make only
   small lexical variations.

Original query: {query}
Return a JSON array of 2 strings and nothing else.
```

The implementation was also built to support alternative retrieval strategies, including BM25-based retrieval and hybrid retrieval. BM25 represents a sparse keyword-based retrieval approach, while hybrid retrieval combines lexical and dense semantic signals [72]. These alternatives were tested during development, but they were not included in the controlled experiments reported in this thesis. The controlled experiments focused on dense retrieval configurations in order to keep the comparison more constrained and to isolate the effects of embedding model choice, retrieval settings, and generation behavior. This decision was also appropriate for

the selected corpus, which consisted of research papers on a relatively coherent technical topic. In this setting, dense semantic retrieval was considered more central to the thesis objective, since many relevant passages may express similar concepts using different formulations rather than exact keyword overlap. BM25 and hybrid retrieval therefore remained implemented extensions rather than part of the final controlled evaluation.

After the first-stage retrieval step, the system can optionally apply a reranking component. The reranker takes the initial retrieved candidates and scores each query–chunk pair using a cross-encoder model. Unlike dense retrieval, where the query and document chunks are embedded independently and compared through vector similarity, a cross-encoder jointly encodes the query and candidate text and produces a relevance score for the pair. This makes reranking more computationally expensive, because each candidate pair must be evaluated separately, but it can provide more fine-grained relevance estimates for a smaller set of candidate chunks. Sentence Transformers describes cross-encoders as models that score predefined sentence pairs and are commonly used to rerank top-k results from a first-stage retriever [73]. In this implementation, the reranker used the `BAAI/bge-reranker-base` model through the Sentence Transformers `CrossEncoder` interface. The `BAAI/bge-reranker-base` model is a text-ranking model from the BGE family and is intended for reranking retrieved candidates [74]. The reranker was applied after candidate retrieval and before final context construction, so that only the most relevant retrieved chunks were passed to the generation model.

The selected chunks are then converted into a context pack. The context pack concatenates the retrieved chunk content into a structured prompt context while preserving citation metadata for each chunk. It also records the source metadata associated with each chunk, enabling the system to display or store which documents and pages were used to support the generated answer.

The final generation step uses `gpt-4.1-mini` to produce an answer conditioned on the user query and the retrieved context. The generation prompt was designed to make grounding an explicit behavioral constraint. Rather than simply asking the model to answer the question, the system prompt instructed it to use only the provided context, avoid prior knowledge, avoid assumptions, state uncertainty when support was partial, and abstain when the context did not contain sufficient evidence. This design reflects the reliability objective of the thesis: the system should not only produce fluent answers, but should produce answers whose relationship to the retrieved evidence can be inspected and evaluated.

The purpose of this prompt was to constrain the language model to the retrieved context and to encourage abstention when the context did not contain sufficient evidence. The retrieved context and the user question were then passed as the user message. This separation made the system-level instruction stable across queries, while the user message changed depending on the retrieved evidence and the input question. The system prompt used for answer generation is shown in Listing 4.2.

Listing 4.2: System prompt for context-grounded answer generation

```
You are a reliable and cautious assistant for a
   Retrieval-Augmented Generation (RAG) system.

You must follow these rules:
1. Only use the provided context to answer the question.
2. Do not use prior knowledge or make assumptions.
3. If the context does not contain direct or reasonably
   inferable information, respond with:
  'There is insufficient evidence in the provided
     context to answer this question.'
4. If the answer is partially supported, clearly state
   uncertainty.
5. Do not hallucinate or fabricate any details.
6. Base your answer strictly on the retrieved text.
7. When possible, reference or quote relevant parts of
   the context.

Your goal is to produce answers that are accurate,
   grounded, and transparent.
```

This prompt design was intended to reduce unsupported generation by explicitly instructing the model to rely only on retrieved evidence. It also introduced an abstention behaviour for cases where the retrieved context was empty or insufficient. If no context was retrieved, the system returned a fixed insufficient-evidence message directly. Otherwise, the language model was instructed to generate a grounded answer or state that the provided context was insufficient.

Overall, the online pipeline transforms a user query into an evidence-grounded answer through semantic retrieval and context-conditioned generation. The query is embedded in the same vector space as the indexed chunks, relevant chunks are retrieved from Qdrant using cosine similarity, optional query expansion and reranking can improve retrieval coverage and relevance,

and the final answer is generated from a structured context pack. This design supports the broader reliability objective of the thesis by preserving the connection between the user query, retrieved evidence, generated response, and later evaluation outputs.

### 4.2.3 Evaluation and Hallucination-Awareness Components

The system includes several components that support hallucination awareness and later evaluation. In this thesis, hallucination awareness is not treated as a single isolated module, but as a property supported by multiple design choices across the pipeline. These include provenance-preserving chunk metadata, a context-grounded generation prompt, optional abstention mechanisms, an NLI grounding layer, and automated evaluation using the RAGAS framework. Together, these components make it possible to inspect whether answers are supported by retrieved evidence and to identify cases where the system may generate unsupported or weakly grounded responses.

A first hallucination-awareness mechanism is introduced during document ingestion and chunking. Each indexed chunk is stored together with metadata describing its source document, page number, chunk id, chunk index, modality, and section title. This metadata is important because a RAG system should not only retrieve text, but also preserve the provenance of the evidence used for generation. By maintaining a link between each retrieved chunk and its original document location, the system can later expose which sources contributed to a generated answer. This supports grounding, traceability, and error analysis. For example, if a generated answer is incorrect, the stored metadata makes it possible to determine whether the error originated from irrelevant retrieval, insufficient retrieved context, or unsupported generation.

The second mechanism is the generation prompt itself. As described in the previous subsection, the system prompt instructs the language model to answer only from the provided context, avoid prior knowledge, avoid assumptions, and abstain when the retrieved context does not contain sufficient evidence. This prompt design is intended to reduce the likelihood of unsupported generation by making context adherence an explicit requirement. It also encourages the model to communicate uncertainty when the retrieved evidence only partially supports an answer. Although prompt instructions cannot guarantee complete elimination of hallucinations, they provide an important first layer of control over the model's behavior.

The system was also designed to support an abstention mechanism before

generation. In this setting, the system can reject or stop the generation process when the retrieved context is empty or judged insufficient. This type of mechanism is relevant for hallucination-aware RAG systems because some user questions may not be answerable from the available knowledge base. In such cases, forcing the language model to generate an answer may increase the risk of unsupported output. During development, an abstention mechanism was tested as a possible reliability layer before passing retrieved chunks to the generator. However, it was not included in the controlled experimental evaluation in order to keep the main experiments focused on comparable retrieval and generation configurations. Instead, abstention was treated as an implemented extension that could be evaluated more systematically in future work.

In addition to prompt-level grounding and abstention behavior, the system includes an NLI-based hallucination detection layer. NLI is commonly formulated as the task of determining whether a hypothesis is entailed by, contradicts, or is neutral with respect to a given premise [75]. The purpose of this layer is to check whether the generated answer is entailed by the retrieved context. NLI models classify the relationship between a premise and a hypothesis, commonly as entailment, contradiction, or neutral. In this implementation, the retrieved context is treated as the premise and the generated answer, or individual sentences from the answer, is treated as the hypothesis. If the NLI model determines that the answer is not sufficiently entailed by the retrieved context, the answer can be flagged as ungrounded or replaced with an insufficient-evidence response. Figure 4.4 shows the NLI-based hallucination detection layer.

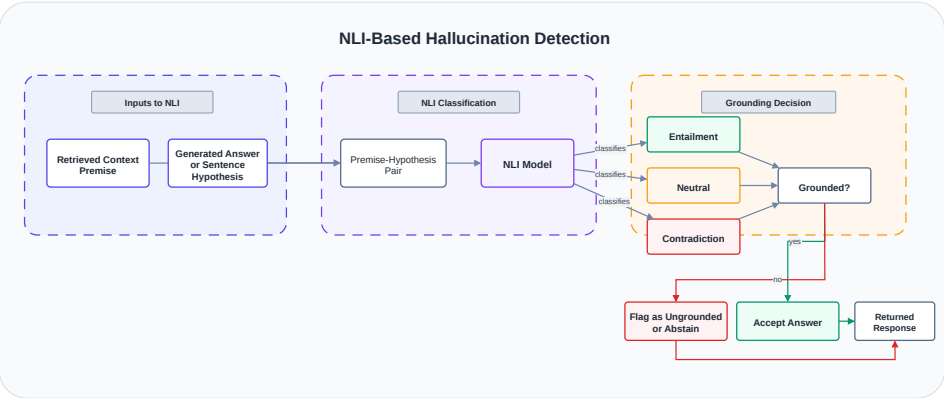


Figure 4.4: NLI-based hallucination detection layer.

The NLI layer was implemented as a post-generation grounding filter. It

split generated answers into sentence-level units, compared them with relevant retrieved context windows, and used entailment scores with configurable thresholds to assess grounding. The purpose was to detect fluent but unsupported answer content. Although the NLI layer was implemented and tested within the pipeline, it was not included in the controlled query-dataset experiments in order to keep the main comparison focused on retrieval configuration, embedding model choice, and generation behaviour. It is therefore presented as a hallucination-awareness extension rather than a primary experimental configuration.

The evaluation component of the system uses the RAGAS framework.* RAGAS is an evaluation framework designed for RAG pipelines and provides metrics for assessing both retrieval and generation quality [9]. This is useful because evaluating a RAG system requires more than judging whether the final answer appears correct. The retrieved context must also be evaluated for relevance, and the generated answer must be evaluated for faithfulness to that context. In this thesis, RAGAS is used through its faithfulness metric to support systematic evaluation of the generated answers and their relationship to the retrieved evidence. The RAGAS evaluation was configured to use an LLM-as-judge setup, with `gpt-4.1-mini` used as the judge model. This allowed the evaluation process to assess answer quality and groundedness in relation to the retrieved contexts. This aligns with the broader motivation of hallucination-aware RAG, where reliability depends not only on answer fluency, but also on whether the answer can be justified by retrieved sources.

*https://github.com/vibrantlabsai/ragas

# Chapter 5

# Results and Analysis

This chapter presents and analyses the experimental results of the evaluated RAG system. The results reported in this chapter are based on the evaluation design described in Chapter 3 and the system implementation described in Chapter 4. All configurations were evaluated using the same indexed research-paper corpus, query dataset, and evaluation procedure. This controlled setting allows differences in the results to be interpreted primarily in relation to retrieval strategy, retrieval depth, and embedding model choice.

## 5.1 Evaluated Configurations

The experiments compared three RAG pipeline configurations. The first configuration was a simple baseline RAG pipeline. In this configuration, the user query was passed directly to dense retrieval, a fixed number of top-ranked chunks was selected, and the retrieved context was then used for answer generation. From this point onward, this configuration is referred to as *Baseline RAG*. This configuration served as the reference point for evaluating whether the more advanced retrieval strategies improved reliability-related behaviour.

The second configuration was a query expansion-based RAG pipeline. In this configuration, the original user query was first expanded into additional query variants. Dense retrieval was then performed for each query, and the retrieved results were merged using reciprocal rank fusion. The merged results were reranked before a fixed number of chunks was selected and passed to the generator. This configuration was included to evaluate whether query reformulation and fusion could improve evidence retrieval and downstream answer quality.

The third configuration was a dynamic retrieval RAG pipeline. In this configuration, the user query was passed to dense retrieval, the retrieved chunks were reranked, and the number of chunks used for generation was determined from the similarity score gap between ranked chunks. Rather than relying on a fixed retrieval depth, the system used this gap to identify a cut-off point where the relevance of subsequent chunks decreased. It was included to evaluate whether adaptive context selection could improve reliability by selecting an amount of evidence that better matched the needs of each query. Table 5.1 summarises the three evaluated pipeline configurations and their retrieval-depth settings.

Table 5.1: Summary of evaluated RAG pipeline configurations

| **Configuration** | **Pipeline structure** | **Retrieval depth** |
|---|---|---|
| Baseline RAG | User query → dense search → fixed top-$k$ → generation | 5, 10, 15 |
| Query expansion RAG | User query → query expansion → dense search per query → RRF merge → rerank → fixed top-$k$ → generation | 5, 10, 15 |
| Dynamic retrieval RAG | User query → dense search → rerank → dynamic top-$k$ selection → generation | Dynamic |

Each configuration was evaluated using three embedding models: `BAAI/bge-large-en-v1.5` [64], `Qwen/Qwen3-Embedding-0.6B` [65], and `sentence-transformers/all-mpnet-base-v2` [34]. For each embedding model, seven experimental settings were tested: baseline RAG with top-$k$ values of 5, 10, and 15; query expansion RAG with top-$k$ values of 5, 10, and 15; and dynamic retrieval RAG. As shown in Table 5.2, this resulted in 21 experimental settings, each evaluated on 200 queries.

Table 5.2: Number of evaluated experimental settings

| **Experimental factor** | **Number** |
|---|---|
| Embedding models | 3 |
| Settings per embedding model | 7 |
| Total experimental settings | 21 |
| Queries per setting | 200 |

This experimental structure made it possible to compare the effect of

pipeline design, retrieval depth, and embedding model choice on the reliability of the generated answers. Since the same query dataset and knowledge base were used across all configurations, the results provide a controlled basis for analysing how each configuration influenced retrieval effectiveness, answer quality, faithfulness, and hallucination-aware behaviour.

## 5.2 Experimental Results

This section reports the quantitative results of the 21 experimental settings described in Section 5.1. The results are organised by pipeline configuration in order to first establish the behaviour of each retrieval strategy before comparing them in Section 5.3. The reported tables focus on Recall@$k$, Hit@$k$, MRR, answer correctness, faithfulness, and the mean number of retrieved chunks.

Precision@$k$ was computed during evaluation but is not included as a main result metric in the following tables. This is because many queries in the evaluation dataset were associated with only one or a small number of annotated supporting chunks. As a result, the maximum attainable Precision@$k$ is structurally limited when multiple chunks are retrieved. For example, if a query has one annotated relevant chunk, the maximum possible Precision@5 is 0.20, while the maximum possible Precision@10 and Precision@15 are 0.10 and 0.067, respectively. Therefore, Recall@$k$, Hit@$k$, and MRR are more informative for evaluating whether the system retrieved relevant evidence and ranked it sufficiently high.

### 5.2.1 Baseline Results

Table 5.3 reports the results for the baseline dense retrieval configuration. Since this configuration does not apply query expansion, reranking, or dynamic context selection, it serves as the reference point for the remaining experiments.

The baseline results show clear differences between the three embedding models. `BAAI/bge-large-en-v1.5` achieves the strongest retrieval performance in the baseline setting. At top-$k = 15$, it obtains the highest Recall@$k$ of 0.509, Hit@$k$ of 0.633, and MRR of 0.281. `Qwen/Qwen3-Embedding-0.6B` performs competitively but remains below BGE on the retrieval-side metrics, reaching Recall@$k$ of 0.416, Hit@$k$ of 0.520, and MRR of 0.261 at top-$k = 15$. In contrast, `sentence-transformers/all-mpnet-base-v2` obtains substantially lower retrieval scores across

Table 5.3: Baseline retrieval and generation results

| Embedding model | Top-$k$ | Count | Recall@$k$ | Hit@$k$ | MRR | Answer correctness | Faithfulness |
|---|---|---|---|---|---|---|---|
| Qwen/Qwen3-Embedding-0.6B | 5 | 200 | 0.270 | 0.373 | 0.243 | 0.626 | 0.906 |
| Qwen/Qwen3-Embedding-0.6B | 10 | 200 | 0.375 | 0.480 | 0.258 | 0.645 | 0.891 |
| Qwen/Qwen3-Embedding-0.6B | 15 | 200 | 0.416 | 0.520 | 0.261 | 0.651 | 0.920 |
| BAAI/bge-large-en-v1.5 | 5 | 200 | 0.283 | 0.387 | 0.254 | 0.645 | 0.890 |
| BAAI/bge-large-en-v1.5 | 10 | 200 | 0.434 | 0.547 | 0.274 | 0.669 | 0.895 |
| BAAI/bge-large-en-v1.5 | 15 | 200 | 0.509 | 0.633 | 0.281 | 0.647 | 0.889 |
| sentence-transformers/all-mpnet-base-v2 | 5 | 200 | 0.104 | 0.127 | 0.079 | 0.563 | 0.875 |
| sentence-transformers/all-mpnet-base-v2 | 10 | 200 | 0.205 | 0.260 | 0.098 | 0.614 | 0.882 |
| sentence-transformers/all-mpnet-base-v2 | 15 | 200 | 0.242 | 0.327 | 0.103 | 0.623 | 0.872 |

all retrieval depths, with Recall@$k$ of 0.242, Hit@$k$ of 0.327, and MRR of 0.103 at top-$k = 15$.

The results also show that increasing the retrieval depth generally improves Recall@$k$ and Hit@$k$. This trend is visible for all three embedding models and is expected because retrieving more chunks increases the probability that at least one annotated supporting chunk is included in the context. However, the effect on answer correctness is less direct. For BGE, answer correctness is highest at top-$k = 10$ with a score of 0.669, even though top-$k = 15$ produces the strongest retrieval coverage. For Qwen and MPNet, answer correctness increases as top-$k$ increases, although the improvements are relatively modest.

Overall, the baseline experiments establish two important reference patterns. First, embedding model choice has a substantial effect on retrieval effectiveness, with BGE producing the strongest baseline retrieval results and MPNet producing the weakest. Second, increasing retrieval depth improves evidence coverage, but the relationship between retrieval coverage and answer quality is not strictly monotonic. These patterns are examined further in the comparative analysis in Section 5.3.

### 5.2.2 Query Expansion Results

Table 5.4 reports the results for the query expansion configuration. This configuration extends the original query into additional query variants, merges the retrieved candidates using reciprocal rank fusion, and applies reranking before selecting the final top-$k$ chunks. The results are therefore compared against the baseline configuration to assess whether query reformulation and reranking improve evidence retrieval and answer quality.

The query expansion results show an overall improvement in retrieval-side performance compared with the baseline configuration. The improvement is particularly visible in MRR, which indicates that relevant chunks are not only retrieved more often but also ranked higher in the final retrieved context. For

Table 5.4: Query expansion retrieval and generation results

| Embedding model | Top-$k$ | Count | Recall@$k$ | Hit@$k$ | MRR | Answer correctness | Faithfulness |
|---|---|---|---|---|---|---|---|
| Qwen/Qwen3-Embedding-0.6B | 5 | 200 | 0.384 | 0.500 | 0.370 | 0.641 | 0.897 |
| Qwen/Qwen3-Embedding-0.6B | 10 | 200 | 0.429 | 0.540 | 0.377 | 0.667 | 0.878 |
| Qwen/Qwen3-Embedding-0.6B | 15 | 200 | 0.426 | 0.533 | 0.368 | 0.662 | 0.894 |
| BAAI/bge-large-en-v1.5 | 5 | 200 | 0.450 | 0.580 | 0.398 | 0.662 | 0.890 |
| BAAI/bge-large-en-v1.5 | 10 | 200 | 0.517 | 0.647 | 0.418 | 0.653 | 0.890 |
| BAAI/bge-large-en-v1.5 | 15 | 200 | 0.549 | 0.667 | 0.406 | 0.665 | 0.898 |
| sentence-transformers/all-mpnet-base-v2 | 5 | 200 | 0.247 | 0.340 | 0.250 | 0.613 | 0.873 |
| sentence-transformers/all-mpnet-base-v2 | 10 | 200 | 0.304 | 0.413 | 0.284 | 0.628 | 0.884 |
| sentence-transformers/all-mpnet-base-v2 | 15 | 200 | 0.313 | 0.413 | 0.280 | 0.637 | 0.857 |

Qwen, MRR increases from 0.243 in the baseline top-$k = 5$ setting to 0.370 with query expansion at top-$k = 5$. For BGE, MRR increases from 0.254 to 0.398 under the same retrieval depth. MPNet also benefits from query expansion, with MRR increasing from 0.079 to 0.250 at top-$k = 5$.

BGE remains the strongest embedding model under the query expansion configuration. At top-$k = 15$, it achieves the highest Recall@$k$ of 0.549 and Hit@$k$ of 0.667. However, its highest MRR is obtained at top-$k = 10$ with a score of 0.418, suggesting that increasing the number of selected chunks improves evidence coverage but does not necessarily improve the rank of the first relevant chunk. Qwen also performs strongly, particularly at top-$k = 10$, where it reaches the highest answer correctness score among the Qwen query expansion runs. MPNet improves substantially compared with its baseline results but remains below Qwen and BGE on all retrieval-side metrics.

The generation-side improvements are more modest than the retrieval-side improvements. Although query expansion generally improves Recall@$k$, Hit@$k$, and MRR, answer correctness does not increase proportionally in every configuration. This indicates that query expansion improves the availability and ranking of relevant evidence, but the final answer quality still depends on how effectively the generator uses the retrieved context.

### 5.2.3 Dynamic Retrieval Results

Table 5.5 reports the results for the dynamic retrieval configuration. In this configuration, the system does not select a fixed number of chunks for every query. Instead, it applies a reranker-based score-gap strategy to dynamically determine how many chunks should be passed to the generator. The dynamic retrieval range was set to a minimum of 3 chunks and a maximum of 15 chunks.

The dynamic retrieval results show that the system selected relatively few chunks on average. Qwen retrieved 4.24 chunks on average, BGE retrieved 5.39, and MPNet retrieved 3.98. This means that dynamic retrieval is most

Table 5.5: Dynamic retrieval results

| Embedding model | Dynamic range | Count | Recall@$k$ | Hit@$k$ | MRR | Answer correctness | Faithfulness | Mean retrieved |
|---|---|---|---|---|---|---|---|---|
| Qwen/Qwen3-Embedding-0.6B | 3–15 | 200 | 0.404 | 0.507 | 0.366 | 0.655 | 0.890 | 4.240 |
| BAAI/bge-large-en-v1.5 | 3–15 | 200 | 0.475 | 0.593 | 0.418 | 0.666 | 0.905 | 5.390 |
| sentence-transformers/all-mpnet-base-v2 | 3–15 | 200 | 0.267 | 0.347 | 0.246 | 0.609 | 0.910 | 3.975 |

directly comparable to the fixed top-$k = 5$ experiments rather than to the top-$k = 10$ or top-$k = 15$ settings.

Despite using a small average context size, dynamic retrieval achieves strong retrieval performance. For BGE, dynamic retrieval obtains Recall@$k$ of 0.475, Hit@$k$ of 0.593, and MRR of 0.418 while retrieving 5.39 chunks on average. This is substantially higher than the BGE baseline top-$k = 5$ setting, where Recall@$k$ is 0.283, Hit@$k$ is 0.387, and MRR is 0.254. Qwen and MPNet show similar improvements relative to their fixed top-$k = 5$ baselines.

BGE again achieves the strongest retrieval and answer correctness results among the three embedding models. It obtains the highest Recall@$k$, Hit@$k$, MRR, and answer correctness in the dynamic retrieval setting. MPNet has the lowest retrieval scores, but its faithfulness score is the highest among the three dynamic retrieval runs. This suggests that faithfulness and retrieval coverage capture different aspects of system behaviour: a configuration can remain faithful to the retrieved context while still retrieving less complete evidence.

### 5.2.4 Performance Across Query Categories

The overall results reported in the previous subsections aggregate three query categories: answerable, ambiguous, and unanswerable queries. This aggregation is useful for comparing complete pipeline configurations, but it can obscure important differences between query types. In particular, answerable queries are associated with identifiable supporting chunks in the corpus, whereas ambiguous and unanswerable queries represent weaker evidence conditions. Ambiguous queries may have partially relevant or multi-interpretable evidence, while unanswerable queries do not have annotated gold supporting chunks. Consequently, retrieval metrics such as Recall@$k$, Hit@$k$, and MRR are not defined for the unanswerable category.

Table 5.6 illustrates this effect using three representative BGE-based configurations: the strongest baseline configuration in terms of answer correctness, the strongest query expansion configuration in terms of retrieval coverage, and the dynamic retrieval configuration. These examples show that answerable queries generally achieve stronger retrieval and faithfulness scores than ambiguous and unanswerable queries within the same experimental run.

Table 5.6: Performance across query categories for selected BGE configurations

| Configuration | Category | Count | Recall@$k$ | Hit@$k$ | MRR | Answer correctness | Faithfulness |
|---|---|---|---|---|---|---|---|
| Baseline top-$k = 10$ | Answerable | 100 | 0.455 | 0.570 | 0.285 | 0.654 | 0.936 |
| | Ambiguous | 50 | 0.394 | 0.500 | 0.252 | 0.616 | 0.893 |
| | Unanswerable | 50 | – | – | – | 0.751 | 0.816 |
| Query expansion top-$k = 15$ | Answerable | 100 | 0.593 | 0.690 | 0.425 | 0.676 | 0.955 |
| | Ambiguous | 50 | 0.461 | 0.620 | 0.367 | 0.633 | 0.866 |
| | Unanswerable | 50 | – | – | – | 0.676 | 0.816 |
| Dynamic retrieval | Answerable | 100 | 0.512 | 0.640 | 0.454 | 0.682 | 0.937 |
| | Ambiguous | 50 | 0.401 | 0.500 | 0.347 | 0.583 | 0.914 |
| | Unanswerable | 50 | – | – | – | 0.717 | 0.831 |

The selected results show a consistent pattern for retrieval-side performance. In all three configurations, answerable queries obtain higher Recall@$k$, Hit@$k$, and MRR than ambiguous queries. For example, in the query expansion configuration with top-$k = 15$, answerable queries reach Recall@$k$ of 0.593, Hit@$k$ of 0.690, and MRR of 0.425, while ambiguous queries reach Recall@$k$ of 0.461, Hit@$k$ of 0.620, and MRR of 0.367. A similar pattern is visible in the dynamic retrieval configuration, where answerable queries achieve Recall@$k$ of 0.512 and MRR of 0.454, compared with Recall@$k$ of 0.401 and MRR of 0.347 for ambiguous queries.

The same distinction is also visible in the faithfulness scores. Answerable queries consistently produce higher faithfulness values than unanswerable queries in the selected configurations. For example, in the query expansion setting, faithfulness is 0.955 for answerable queries but 0.816 for unanswerable queries. In the dynamic retrieval setting, faithfulness is 0.937 for answerable queries and 0.831 for unanswerable queries. This indicates that the generated answers are more reliably grounded when the query has identifiable supporting evidence in the corpus.

The answer correctness scores require more careful interpretation. For unanswerable queries, a high correctness score may reflect a correct refusal or an answer that appropriately indicates insufficient evidence, rather than a factually rich answer grounded in retrieved documents. Therefore, unanswerable queries should not be interpreted in the same way as answerable queries. For answerable queries, answer correctness reflects whether the system retrieved and used the relevant evidence to produce the expected answer. For unanswerable queries, it instead reflects whether the system avoided unsupported answering.

These category-level results are important for interpreting the overall scores. The overall metrics are lower than the answerable-only results because

they combine standard evidence-supported questions with ambiguous and unanswerable cases. This is not a weakness of the evaluation design; rather, it reflects the reliability-oriented goal of the thesis. A hallucination-aware RAG system should not only perform well when relevant evidence is available, but should also behave appropriately when the evidence is incomplete, ambiguous, or absent.

### 5.2.5 Faithfulness and Hallucination Behaviour

The previous subsections primarily focused on retrieval effectiveness and answer correctness. Since the goal of this thesis is to build a more reliable and hallucination-aware RAG system, it is also necessary to examine how faithfully the generated answers were grounded in the retrieved context and where hallucination behaviour remained present.

Across the experiments, faithfulness scores were generally high compared with the retrieval-side metrics. Most configurations produced overall faithfulness values between approximately 0.87 and 0.92. This indicates that, when the system generated an answer, the answer was often judged to be supported by the retrieved context. However, faithfulness did not always follow the same pattern as retrieval performance. For example, in the baseline experiments, BGE achieved the strongest retrieval metrics at top-$k = 15$, but Qwen achieved the highest baseline faithfulness score at the same retrieval depth. Similarly, MPNet had the weakest retrieval results overall, but in the dynamic retrieval setting it achieved the highest faithfulness score among the three embedding models.

This distinction shows that retrieval quality and faithfulness measure related but different aspects of RAG behaviour. Retrieval metrics evaluate whether relevant evidence was retrieved and how highly it was ranked. Faithfulness evaluates whether the generated answer remains supported by the retrieved context. A system may retrieve a more complete set of evidence but still generate an answer that is less faithful if the generator selectively ignores, misinterprets, or combines context incorrectly. Conversely, a system may retrieve fewer relevant chunks but still generate a cautious answer that remains faithful to the limited context it received.

The hallucination results further show that retrieval augmentation does not eliminate unsupported generation. Hallucination rates were generally lower for answerable queries and higher for ambiguous or unanswerable queries. This pattern is consistent with the category-level results in Section 5.2.4. When relevant supporting evidence is available, the generator has a clearer

basis for producing a grounded response. When the evidence is incomplete, ambiguous, or absent, the risk of unsupported generation increases.

The unanswerable category is particularly important for evaluating hallucination-aware behaviour. In these cases, the system should ideally avoid producing a direct factual answer unless the retrieved context provides sufficient support. Therefore, hallucination behaviour in unanswerable queries reflects not only generation quality, but also the system's ability to recognise insufficient evidence. This makes the unanswerable subset central to the reliability evaluation, since a RAG system that performs well only on answerable questions may still be unsafe if it answers confidently when the corpus does not contain enough evidence.

Overall, the results indicate that high retrieval performance improves the conditions for grounded generation, but it is not sufficient on its own. Faithfulness and hallucination behaviour depend on both the quality of the retrieved evidence and the generator's ability to use that evidence appropriately. This motivates the comparative analysis in the next section, where the effects of embedding model choice, query expansion, retrieval depth, and dynamic retrieval are examined in more detail.

## 5.3 Comparative Analysis

The previous section reported the results for each experimental configuration. This section compares the configurations across the main experimental factors: embedding model choice, query expansion, retrieval depth, and dynamic retrieval. The purpose is to analyse how these factors influenced retrieval relevance, answer correctness, faithfulness, and hallucination-aware behaviour. This analysis also connects the empirical results to the sub-research questions concerning evidence relevance, insufficient retrieval support, and uncertainty-aware response behaviour.

### 5.3.1 Effect of Embedding Model Choice

The choice of embedding model had a substantial effect on retrieval effectiveness across all three pipeline configurations. Since the generator, corpus, query dataset, and evaluation procedure were kept constant, the differences between embedding models can primarily be attributed to how well each model represented the semantic relationship between user queries and document chunks.

Across the baseline experiments, `BAAI/bge-large-en-v1.5` achieved the strongest retrieval performance. At top-$k = 15$, BGE reached Recall@$k$ of 0.509, Hit@$k$ of 0.633, and MRR of 0.281. In comparison, `Qwen/Qwen3-Embedding-0.6B` reached Recall@$k$ of 0.416, Hit@$k$ of 0.520, and MRR of 0.261 at the same retrieval depth, while `sentence-transformers/all-mpnet-base-v2` reached substantially lower values, with Recall@$k$ of 0.242, Hit@$k$ of 0.327, and MRR of 0.103. Figure 5.1 visualises these differences and shows that BGE produced the strongest retrieval profile in the baseline setting, while MPNet consistently retrieved fewer relevant chunks and ranked them lower.

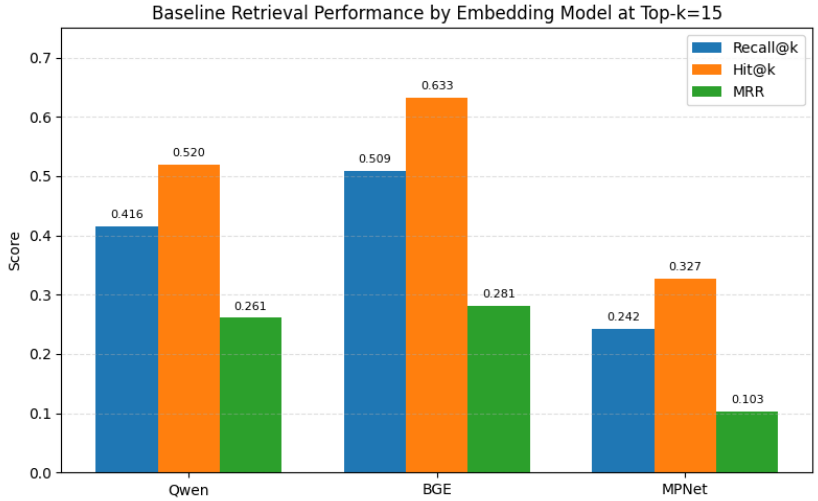


Figure 5.1: Baseline retrieval performance by embedding model at top-$k = 15$.

The same trend was also visible in the query expansion and dynamic retrieval experiments. Under query expansion, BGE achieved the highest retrieval coverage, reaching Recall@$k$ of 0.549 and Hit@$k$ of 0.667 at top-$k = 15$. Qwen reached Recall@$k$ of 0.426 and Hit@$k$ of 0.533 at the same retrieval depth, while MPNet reached Recall@$k$ of 0.313 and Hit@$k$ of 0.413. In the dynamic retrieval configuration, BGE again produced the strongest retrieval results, with Recall@$k$ of 0.475, Hit@$k$ of 0.593, and MRR of 0.418. Qwen followed with Recall@$k$ of 0.404, Hit@$k$ of 0.507, and MRR of 0.366, while MPNet remained lower with Recall@$k$ of 0.267, Hit@$k$ of 0.347, and MRR of 0.246.

These results indicate that embedding model choice is not merely an implementation detail, but a central factor in RAG reliability. A stronger

embedding model improves the probability that relevant evidence is available to the generator, which creates better conditions for grounded answer generation. This is especially important in a hallucination-aware RAG system, because the generator can only ground its response in evidence that has actually been retrieved.

However, the results also show that stronger retrieval metrics do not always lead to proportionally higher answer correctness or faithfulness. For example, in the baseline experiments, BGE achieved the highest retrieval coverage at top-$k = 15$, but its answer correctness was highest at top-$k = 10$. Similarly, Qwen sometimes achieved higher faithfulness despite lower retrieval coverage than BGE. This suggests that retrieval quality is a necessary but not sufficient condition for reliable generation. The retrieved evidence must also be sufficiently focused, non-conflicting, and effectively used by the generator.

The weaker performance of `sentence-transformers/all-mpnet-base-v2` further illustrates the importance of embedding-query alignment. MPNet consistently produced lower Recall@$k$, Hit@$k$, and MRR values than BGE and Qwen, making it less likely that the generator received the most relevant supporting chunks. Although MPNet occasionally achieved competitive faithfulness scores, this should not be interpreted as stronger retrieval behaviour. Rather, it suggests that the system could remain faithful to the retrieved context even when that context was incomplete or less relevant.

A plausible explanation is that the models differ in how strongly they are optimised for retrieval-oriented use cases. `sentence-transformers/all-mpnet-base-v2` is a broadly applicable sentence embedding model for tasks such as semantic search, clustering, and sentence similarity. In contrast, `BAAI/bge-large-en-v1.5` is part of the BGE embedding family, whose v1.5 release is described as improving retrieval ability, while Qwen3-Embedding is explicitly designed for text embedding and ranking tasks. This may explain why BGE and Qwen were better aligned with the retrieval-heavy evaluation setting, although this remains a plausible interpretation rather than a definitive causal claim.

Overall, the experiments show that embedding model choice has a direct effect on the relevance and grounding potential of retrieved information. BGE provided the strongest retrieval foundation in this evaluation, Qwen provided competitive but generally lower retrieval performance, and MPNet was the weakest of the three embedding models. This finding is central to the main evaluation goal of the thesis, because it shows that retrieval reliability depends strongly on the semantic representation used to connect user queries with

evidence chunks.

### 5.3.2 Effect of Query Expansion

Query expansion had a clear positive effect on retrieval-side performance across all three embedding models. Compared with the baseline configuration, the query expansion pipeline generally improved Recall@$k$, Hit@$k$, and MRR. This indicates that generating additional query variants, combining retrieved candidates through reciprocal rank fusion, and applying reranking helped the system retrieve relevant chunks more often and place them higher in the final ranking.

The strongest improvements are visible at top-$k = 5$, where the retrieval setting is most constrained. For Qwen, MRR increased from 0.243 in the baseline configuration to 0.370 with query expansion. For BGE, MRR increased from 0.254 to 0.398, while MPNet improved from 0.079 to 0.250. Figure 5.2 visualises this improvement. The increase in MRR is particularly important because it shows that query expansion did not only increase the probability of retrieving relevant evidence, but also improved the position of the first relevant chunk in the retrieved context.

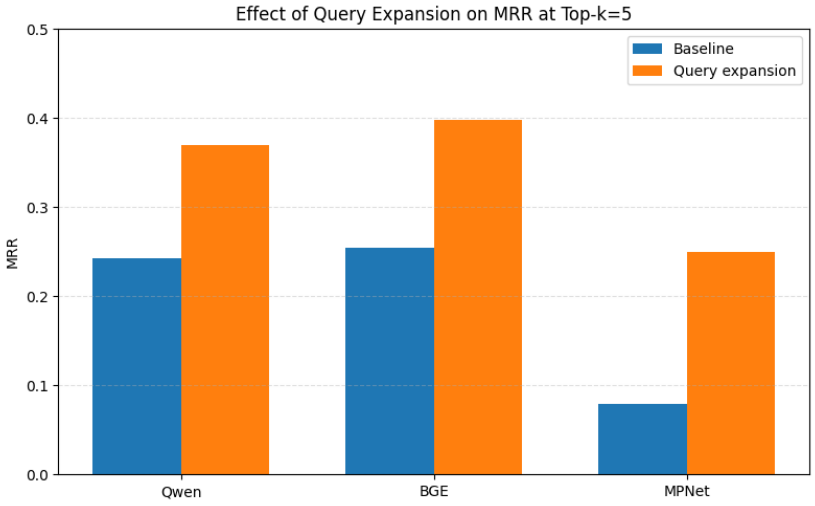


Figure 5.2: Effect of query expansion on MRR at top-$k = 5$.

The effect of query expansion can be explained by the fact that a single user query may not fully match the wording or terminology used in the indexed corpus. By generating additional query variants, the system increases the chance that at least one formulation is semantically closer to the relevant

chunks. Reciprocal rank fusion then combines evidence from the different retrieval runs, while reranking helps prioritise the most relevant candidates before the final top-$k$ chunks are selected. This is especially useful for ambiguous or underspecified queries, where the original query alone may not provide enough retrieval signal.

Overall, query expansion improved the availability and ranking of relevant evidence, especially under smaller retrieval depths. Its strongest contribution was improving MRR, which indicates better placement of relevant chunks near the top of the retrieved context. However, the relatively smaller changes in answer correctness show that reliable RAG behaviour depends not only on retrieving better evidence, but also on how effectively the generator uses that evidence.

### 5.3.3 Effect of Retrieval Depth

Retrieval depth had a clear effect on retrieval coverage. Across the fixed-depth experiments, increasing top-$k$ generally improved Recall@$k$ and Hit@$k$ because the system was allowed to retrieve more chunks. This increased the probability that at least one annotated supporting chunk was included in the retrieved context. However, the results also show that deeper retrieval did not always improve answer correctness or faithfulness.

The clearest example is the BGE baseline configuration. As shown in Figure 5.3, increasing top-$k$ from 5 to 15 improved Recall@$k$ from 0.283 to 0.509 and Hit@$k$ from 0.387 to 0.633. MRR also increased slightly, from 0.254 at top-$k = 5$ to 0.281 at top-$k = 15$. However, answer correctness did not follow the same pattern. It was highest at top-$k = 10$ with a score of 0.669, while top-$k = 15$ produced a lower answer correctness score of 0.647 despite having the strongest retrieval coverage.

This pattern suggests that retrieval depth introduces a trade-off between evidence coverage and context focus. A larger top-$k$ increases the chance of retrieving relevant information, but it can also introduce additional irrelevant, weakly relevant, or redundant chunks. These additional chunks may make the context less focused and can complicate the generation step. Therefore, the best retrieval depth is not necessarily the largest one; it depends on whether the additional retrieved evidence improves the generator's ability to answer or instead adds noise to the context.

The same trade-off is visible in the query expansion results. For BGE, Recall@$k$ increased from 0.450 at top-$k = 5$ to 0.549 at top-$k = 15$, while Hit@$k$ increased from 0.580 to 0.667. However, MRR was highest at top-$k =$

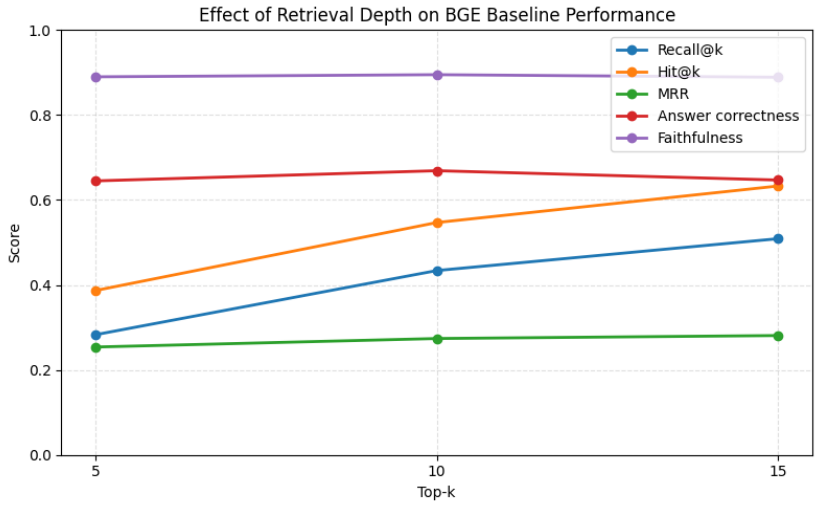


Figure 5.3: Effect of retrieval depth on BGE baseline performance.

10 rather than top-$k = 15$, and answer correctness remained relatively close across the three retrieval depths. This indicates that deeper retrieval improved evidence coverage, but the most relevant evidence was not necessarily placed more effectively in the final context.

For Qwen and MPNet, increasing top-$k$ also tended to improve retrieval coverage, but the gains in answer correctness were modest. This reinforces the observation that retrieval depth mainly affects the availability of evidence, while answer quality depends on how well the retrieved evidence is ranked, selected, and used by the generator. In other words, increasing top-$k$ can reduce the risk of missing relevant chunks, but it does not by itself guarantee more reliable or more faithful answers.

Overall, retrieval depth should be treated as a reliability trade-off rather than a simple performance parameter. A small top-$k$ may fail to retrieve enough evidence, while a large top-$k$ may introduce unnecessary context and reduce focus. The results suggest that intermediate retrieval depths can sometimes provide a better balance between coverage and answer quality, as seen in the BGE baseline setting where top-$k = 10$ produced the highest answer correctness despite top-$k = 15$ producing the highest retrieval coverage.

### 5.3.4 Effect of Dynamic Retrieval

Dynamic retrieval was introduced to avoid using the same retrieval depth for every query. Instead of always passing a fixed number of chunks to the generator, the system used a reranker-based score-gap strategy to determine how many chunks should be included. This made the retrieval depth dependent on the apparent strength and separation of the reranked evidence.

The dynamic retrieval results are most directly comparable to the fixed top-$k = 5$ experiments, since the mean number of retrieved chunks was close to five for all three embedding models. Qwen retrieved 4.24 chunks on average, BGE retrieved 5.39, and MPNet retrieved 3.98. Despite using a small average context size, dynamic retrieval substantially improved retrieval-side performance compared with the corresponding baseline top-$k = 5$ runs.

For BGE, dynamic retrieval improved Recall@$k$ from 0.283 in the baseline top-$k = 5$ setting to 0.475, Hit@$k$ from 0.387 to 0.593, and MRR from 0.254 to 0.418. Similar improvements were observed for Qwen and MPNet. For Qwen, Recall@$k$ increased from 0.270 to 0.404 and MRR increased from 0.243 to 0.366. For MPNet, Recall@$k$ increased from 0.104 to 0.267 and MRR increased from 0.079 to 0.246. Figure 5.4 visualises this comparison.

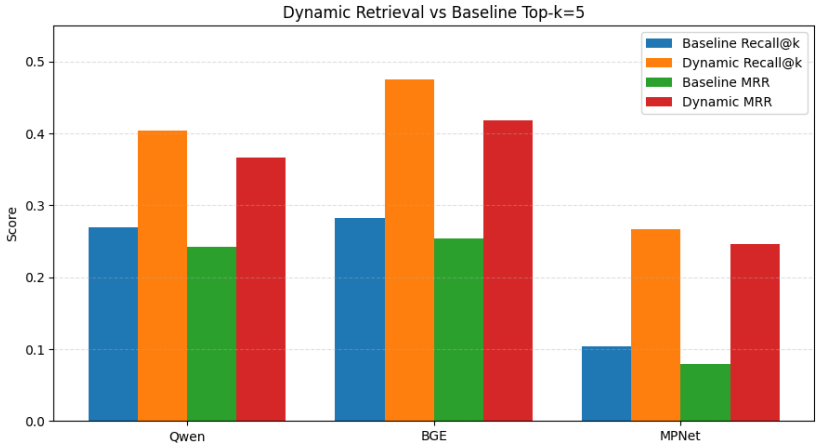


Figure 5.4: Dynamic retrieval compared with fixed top-$k = 5$ baseline retrieval.

These results suggest that dynamic retrieval provides a better balance between context size and evidence quality than simple fixed-depth retrieval. Rather than increasing the number of retrieved chunks for every query, dynamic retrieval uses the reranker scores to select a smaller but more relevant set of chunks. This is particularly useful for reliability-oriented RAG because

irrelevant or weakly relevant chunks can distract the generator or reduce the clarity of the retrieved context.

The results also show that dynamic retrieval can provide an evidence-confidence signal. If the reranker produces a clear separation between highly relevant and weakly relevant chunks, the system can select fewer chunks. If the evidence is less clearly separated, the system can include more chunks up to the maximum retrieval depth. This makes dynamic retrieval useful not only for controlling context size, but also for identifying cases where the retrieved evidence may be less certain.

Overall, dynamic retrieval was one of the most effective strategies in the evaluation. It achieved retrieval performance closer to deeper retrieval settings while using a mean context size close to top-$k = 5$. This makes it valuable for systems where reliability depends not only on retrieving relevant evidence, but also on avoiding unnecessary or noisy context.

## 5.4 Error Analysis

The quantitative results show how different retrieval configurations affected retrieval quality, answer correctness, faithfulness, and hallucination behaviour. However, aggregate metrics do not fully explain why individual failures occurred. This section therefore examines representative qualitative failure cases from the experiment logs. The goal is to identify recurring failure modes that affected the reliability of the RAG system, especially in cases where the system either refused to answer despite available evidence or generated an answer despite insufficient evidence. Table 5.7 summarises the main qualitative failure types observed during this analysis.

### 5.4.1 False Insufficiency

One recurring failure mode was false insufficiency. In these cases, the system stated that there was insufficient evidence even though the query was labelled as answerable and the expected answer was present in the knowledge base. This behaviour is reliability-relevant because hallucination awareness should not only prevent unsupported answers; it should also avoid unnecessary refusal when sufficient evidence exists.

Representative examples of this failure mode were extracted from the BGE query expansion top-$k = 15$ run. This configuration was selected because it achieved the strongest overall retrieval coverage among the evaluated configurations, with Recall@$k$ of 0.549 and Hit@$k$ of 0.667. Therefore, false

Table 5.7: Observed qualitative failure types

| **Failure type** | **Description** | **Reliability implication** |
|---|---|---|
| False insufficiency | The system states that there is insufficient evidence even though the query is answerable and relevant information exists in the knowledge base. | The system becomes overly conservative and fails to use available evidence. |
| False certainty | The system gives a direct answer to an unanswerable query instead of indicating insufficient support. | The system risks presenting unsupported or weakly supported information as grounded. |
| Partial grounding | The answer uses some retrieved evidence but also includes unsupported or over-specific claims. | The response appears grounded but may still contain hallucinated or overextended content. |
| Context misinterpretation | Relevant context is retrieved, but the generator fails to use it correctly. | Retrieval succeeds, but generation does not convert the evidence into a reliable answer. |
| Context dilution | Relevant evidence is present but surrounded by irrelevant or redundant chunks. | The generator may overlook the most useful evidence or produce a less focused answer. |

insufficiency cases in this setting are especially informative: they suggest that refusal can occur even when the retrieval configuration is comparatively strong. The full examples are provided in Appendix B, Table B.1.

In each case, the system generated the same refusal response: "There is insufficient evidence in the provided context to answer this question." However, each query had an expected answer in the knowledge base. This indicates that the system did not simply fail by hallucinating; instead, it failed by abstaining when an answer should have been produced.

These cases suggest that the system sometimes applied its insufficiency criterion too conservatively. This may occur when the relevant information is not phrased in a directly extractable form, when the evidence is distributed

across multiple chunks, or when the prompt encourages cautious behaviour under uncertainty. In such cases, the system avoids hallucination by refusing to answer, but the refusal is still incorrect because the query is answerable from the knowledge base.

False insufficiency therefore illustrates an important limitation of generic abstention behaviour. A refusal is desirable when evidence is genuinely absent, but it becomes a utility failure when sufficient evidence exists. A reliable RAG system must therefore balance caution with evidence sensitivity: it should abstain when support is insufficient, but answer when the retrieved evidence is strong enough.

### 5.4.2 Ambiguous Query Behaviour

Ambiguous queries were useful for analysing how the system behaved when the retrieved evidence supported a qualified answer rather than a single definitive response. Unlike answerable queries, ambiguous queries often required the system to express uncertainty, compare alternatives, or explain that the optimal answer depends on the task, dataset, or evaluation criterion. This makes them particularly relevant for evaluating hallucination-aware behaviour, because the desired response is not always a refusal or a direct factual answer.

Representative ambiguous-query cases were inspected from the BGE query expansion top-$k = 15$ run. The full examples are provided in Appendix B, Table B.2.

The inspected examples showed three recurring behaviours. First, the system sometimes produced appropriately qualified answers, acknowledging that the evidence did not support a universally dominant option. Second, it sometimes produced over-specific answers by selecting one option as best even though the expected answer required a more conditional interpretation. Third, it sometimes over-refused by stating that there was insufficient evidence, even when the available context supported a qualified answer.

These behaviours show that ambiguity is not the same as unanswerability. In ambiguous cases, a reliable RAG system should not necessarily abstain; instead, it should communicate the limits of the evidence and provide a conditional answer when appropriate. The results therefore suggest that hallucination-aware behaviour requires response calibration, not only binary answer-versus-refusal decisions.

### 5.4.3 Behaviour on Unanswerable Queries

Unanswerable queries were included to evaluate whether the system could recognise when the available evidence was insufficient to support a grounded answer. However, qualitative inspection showed that this category requires careful interpretation. Representative examples from the BGE query expansion top-$k = 15$ run are provided in Appendix B, Table B.3.

The inspected examples showed two main behaviours. First, the system abstained correctly by returning the expected insufficient-evidence response. Second, it sometimes produced a qualified answer that partially addressed the query while still acknowledging that the exact requested detail was not available. These behaviours show that unanswerable-query handling depends on the granularity of the requested information: related evidence may be present, but it may not support the exact detail required by the query. A reliable system should therefore either abstain or provide a clearly qualified response when the retrieved context does not fully support the requested information.

## 5.5 Validity Analysis

This section discusses the main validity considerations that affect the interpretation of the experimental results. The purpose is not to claim that the findings generalise to all RAG systems, but to clarify the strengths and limitations of the evaluation setup used in this thesis.

### 5.5.1 Internal Validity

Internal validity concerns whether the observed differences between configurations can reasonably be attributed to the experimental factors being tested. The evaluation was designed to support controlled comparison by keeping the corpus, query dataset, generator, evaluator, and evaluation procedure constant across all experiments. The main variables changed between runs were the embedding model, retrieval strategy, and retrieval depth.

This controlled setup strengthens the comparison between configurations. For example, differences between embedding models can be interpreted primarily as retrieval-side differences because the generation model and evaluation procedure were unchanged. Similarly, comparisons between baseline retrieval, query expansion, and dynamic retrieval were performed using the same query set and knowledge base.

However, some internal validity limitations remain. LLM-based generation and evaluation may introduce variability, and the evaluator may not always judge correctness, faithfulness, or hallucination behaviour in the same way as a human expert. Therefore, the quantitative results should be interpreted as comparative indicators rather than exact measurements of system reliability.

### 5.5.2 Construct Validity

Construct validity concerns whether the selected metrics capture the reliability-related properties that the thesis aims to evaluate. Retrieval quality was measured using Recall@$k$, Hit@$k$, and MRR, which together indicate whether relevant evidence was retrieved and how highly it was ranked. Generation quality was evaluated using answer correctness and faithfulness, which capture whether the generated answer was correct and whether it was grounded in the retrieved context.

A limitation is that no single metric fully captures RAG reliability. High retrieval scores do not guarantee that the generator will use the evidence correctly, and high faithfulness does not necessarily imply that the answer is complete or correct. Similarly, answer correctness may be affected by whether the evaluator judges a refusal, partial answer, or qualified answer as acceptable.

For this reason, the results were interpreted using multiple metrics rather than relying on one score. The quantitative evaluation was also complemented with qualitative error analysis to better understand failure cases such as false insufficiency, over-refusal, and uncertainty handling in ambiguous queries.

### 5.5.3 Dataset and Annotation Validity

Dataset and annotation validity concerns whether the query labels, expected answers, and supporting evidence accurately reflect the intended evaluation task. The query dataset was designed to include answerable, ambiguous, and unanswerable questions, which made it possible to evaluate different reliability-related behaviours rather than only standard answer generation.

A limitation is that these categories depend on annotation decisions. In particular, ambiguous and unanswerable queries require judgement about what counts as sufficient evidence. If the expected answer or supporting-evidence annotation does not fully capture the available context, this can affect both retrieval metrics and generation-side evaluation.

For this reason, the results for ambiguous and unanswerable queries

were interpreted cautiously. The qualitative error analysis was used to complement the aggregate metrics and to distinguish between unsupported answers, qualified answers, and appropriate refusals.

### 5.5.4 External Validity

External validity concerns the extent to which the findings generalise beyond the specific prototype and evaluation setting used in this thesis. The experiments were conducted on a fixed research-paper corpus and a query dataset focused on RAG, hallucination awareness, and related evaluation methods. As a result, the exact metric values may not transfer directly to other domains, larger knowledge bases, or production environments.

The results may also differ with other generators, embedding models, rerankers, chunking strategies, or prompting approaches. For example, a different corpus may favour different embedding models, and a different generator may respond differently to the same retrieved context.

Nevertheless, the observed patterns are relevant for similar RAG system-building contexts. In particular, the experiments show that embedding model choice, retrieval depth, query expansion, and dynamic retrieval can substantially affect retrieval quality and grounded answer generation. These findings provide useful evidence about pipeline-level trade-offs even if the exact scores are specific to this implementation.

# Chapter 6

# Discussion

This chapter discusses the implications of the experimental results presented in Chapter 5. While the previous chapter reported and compared the evaluated configurations in detail, this chapter focuses on what the findings mean for the design of reliable and hallucination-aware RAG systems.

## 6.1 Main Interpretation of the Results

The results show that RAG reliability is strongly shaped by retrieval-side design choices, but not determined by retrieval quality alone. Embedding model choice, retrieval depth, query expansion, and dynamic retrieval all affected the relevance and ranking of retrieved evidence. In particular, BGE generally produced the strongest retrieval results, while MPNet performed weakest. This indicates that the semantic representation used for retrieval is a central design decision in a RAG system.

At the same time, the experiments showed that stronger retrieval metrics did not always lead to proportional improvements in answer correctness or faithfulness. Query expansion improved retrieval coverage and ranking, but generation-side improvements were more limited. Similarly, increasing top-$k$ improved the probability of retrieving relevant evidence, but did not always improve answer quality. These findings suggest that retrieval is necessary for grounded generation, but it is not sufficient. The generator must still identify and use the relevant evidence correctly.

Dynamic retrieval was the most promising strategy from a reliability perspective. It achieved strong retrieval performance while using a relatively small average number of retrieved chunks. This suggests that adaptive context selection can help balance evidence coverage and context focus, instead of

assuming that every query requires the same fixed retrieval depth.

## 6.2 Implications for Reliable and Hallucination-Aware RAG

The findings suggest that hallucination-aware RAG should be treated as a pipeline-level problem. Unsupported or unreliable answers can result from retrieval failures, noisy context, insufficient evidence, or poor response calibration. Therefore, improving reliability requires more than adding retrieval to a language model.

A key implication is that evidence relevance and evidence sufficiency are different. Retrieved chunks may be topically relevant without being sufficient to answer the exact question. This was especially visible in ambiguous and unanswerable-query behaviour, where the appropriate response was often a qualified answer or abstention rather than a direct factual answer.

The error analysis also showed that abstention must be calibrated carefully. Refusing to answer can reduce hallucination risk, but unnecessary refusal reduces usefulness. A reliable system should therefore support different response modes: direct grounded answers when evidence is strong, qualified answers when evidence is partial, and abstention when evidence is insufficient.

Overall, the prototype demonstrates that hallucination awareness depends on combining retrieval quality, evidence sufficiency assessment, and uncertainty communication. Dynamic retrieval contributes useful retrieval-side signals, but these signals must be combined with generation-side grounding checks to produce transparent and trustworthy answers.

# Chapter 7

# Conclusions and Future work

This chapter concludes the thesis by summarising the main outcomes of the work, reflecting on the limitations of the study, and outlining directions for future development.

## 7.1 Conclusions

This thesis set out to design, implement, and evaluate a RAG system with a focus on reliability and hallucination-aware behaviour. The work covered the full development cycle of a prototype RAG system: implementing a modular pipeline, preparing a fixed knowledge base, constructing an evaluation dataset, running experiments across multiple configurations, and analysing the resulting behaviour.

The main objective of the thesis was achieved. The final prototype supported controlled experimentation across different retrieval strategies, embedding models, and retrieval depths. This made it possible to study RAG reliability as a pipeline-level property rather than as the result of a single component. The evaluation also produced meaningful results that helped reveal how different design choices affected retrieval quality, grounded answer generation, and uncertainty-related behaviour.

A key outcome of the work is that the implemented system provides a practical basis for experimenting with hallucination-aware RAG design. The modular structure made it possible to isolate and compare the effects of different pipeline components, while the evaluation dataset enabled the system to be tested on answerable, ambiguous, and unanswerable queries. This combination of implementation and evaluation supports the broader aim of understanding how RAG systems can be made more transparent and

trustworthy.

The thesis also showed that reliability in RAG is not only a matter of retrieving more information. The system must retrieve useful evidence, determine whether that evidence is sufficient, and communicate uncertainty when the evidence does not support a confident answer. This insight is one of the central conclusions of the project and provides a foundation for the future development of more robust hallucination-aware RAG systems.

Overall, the project demonstrates both the potential and the complexity of building reliable RAG systems. The prototype successfully enabled systematic experimentation, but the results also show that reliability remains an open engineering and evaluation challenge. The work can therefore be seen as an entry point into a broader design space, where future systems can build on the prototype and extend it with stronger retrieval, better uncertainty handling, larger corpora, and more production-oriented evaluation.

## 7.2 Limitations

Several limitations should be considered when interpreting the results of this thesis. First, the work was limited by the available computational and financial resources. Since RAG systems can involve expensive embedding, reranking, generation, and evaluation components, the prototype was designed around models and services that were feasible within the scope of a Master's thesis. More powerful models may have produced different results, but they would also require greater computational resources and higher API costs.

Second, the evaluation was conducted on a fixed and relatively small research-paper corpus focused on RAG, hallucination awareness, and related evaluation methods. This made the experimental setting controlled and relevant to the thesis topic, but it also limits generalisation. A larger and more diverse corpus could produce different retrieval behaviour, more varied answer types, and different failure patterns.

Third, the experiments focused mainly on retrieval quality, answer correctness, faithfulness, and hallucination-related behaviour. Response latency and cost were not treated as primary evaluation metrics. This is a limitation because practical RAG systems must also be evaluated in terms of efficiency, scalability, and operational feasibility.

Finally, only a limited number of pipeline configurations could be tested. RAG is a rapidly developing field, and many other retrieval strategies, reranking methods, indexing approaches, prompting techniques, and agentic

architectures could be explored. The prototype should therefore be understood as an experimental foundation rather than a complete production-ready system.

## 7.3 Future Work

The prototype developed in this thesis provides a foundation for several directions of future work. One natural extension is to evaluate the system on a larger and more diverse document corpus. The current corpus was focused on RAG and hallucination-awareness research, which made the evaluation controlled but domain-specific. Future work could include documents from multiple domains and formats to test how well the pipeline generalises to more varied knowledge bases.

Another direction is to test additional retrieval and indexing strategies. The current work focused on dense retrieval, query expansion, reranking, and dynamic retrieval. Future systems could explore graph-based retrieval, hierarchical indexing, hybrid sparse-dense retrieval, or document-structure-aware retrieval. These approaches may improve the system's ability to retrieve evidence from larger and more complex corpora.

Future work should also improve evidence sufficiency and uncertainty handling. The results showed that reliable behaviour requires more than retrieving relevant chunks; the system must also determine whether the retrieved evidence is sufficient for the specific query. This could be addressed through explicit answerability classifiers, confidence scoring, stronger abstention mechanisms, or post-generation verification.

Agentic RAG is another promising direction. Instead of using a fixed retrieval-generation pipeline, an agentic system could iteratively decide when to retrieve more evidence, reformulate queries, verify intermediate answers, or consult additional sources such as web search. This could make the system more flexible, especially for complex or ambiguous queries.

Finally, future work should include a more detailed cost and latency analysis. The current thesis focused primarily on reliability-related metrics, but a production-ready RAG system must also be evaluated in terms of response time, API cost, scalability, and operational feasibility. Such an analysis would help determine which pipeline configurations are practical outside an experimental setting.

## 7.4 Reflections

This project showed that building a reliable RAG system is both an engineering and evaluation challenge. The work required developing the prototype, preparing the knowledge base, constructing the query dataset, running experiments, and interpreting the results together. This made the thesis broader than a standard model comparison and provided practical insight into how many design decisions affect system behaviour.

From a technical perspective, the project highlighted the importance of modularity. Since the RAG pipeline was implemented in a modular way, it was possible to replace components and compare different configurations systematically. This was essential for understanding how embedding models, retrieval strategies, and context selection affected reliability.

There are also important ethical and practical implications. RAG systems are often used because they appear to provide grounded and trustworthy answers, but the results show that retrieval does not automatically guarantee reliability. A system may still answer with insufficient support or refuse despite available evidence. For this reason, systems that generate answers from retrieved documents should communicate uncertainty clearly and avoid presenting weakly supported claims as certain.

Finally, the work showed the strong future potential of RAG systems. The field is developing quickly, and new models, retrieval techniques, and agentic methods continue to appear. The prototype developed in this thesis represents one step in this larger development space. It demonstrates that meaningful reliability analysis is possible, while also showing that much remains to be explored before RAG systems can be considered fully trustworthy in high-stakes applications.

# References

[1] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian, "A comprehensive overview of large language models," 2024. [Online]. Available: https://arxiv.org/abs/2307.06435

[2] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2312.10997

[3] W. Zhang and J. Zhang, "Hallucination mitigation for retrieval-augmented large language models: A review," *Mathematics*, vol. 13, p. 856, 03 2025. doi: 10.3390/math13050856

[4] Y. Zhou, Y. Liu, X. Li, J. Jin, H. Qian, Z. Liu, C. Li, Z. Dou, T.-Y. Ho, and P. S. Yu, "Trustworthiness in retrieval-augmented generation systems: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2409.10102

[5] P. Manakul, A. Liusie, and M. J. F. Gales, "Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models," 2023. [Online]. Available: https://arxiv.org/abs/2303.08896

[6] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," *CoRR*, vol. abs/2005.11401, 2020. [Online]. Available: https://arxiv.org/abs/2005.11401

[7] S. Barnett, S. Kurniawan, S. Thudumu, Z. Brannelly, and M. Abdelrazek, "Seven failure points when engineering a retrieval augmented generation system," 2024. [Online]. Available: https://arxiv.org/abs/2401.05856

[8] X. Peng, P. K. Choubey, C. Xiong, and C.-S. Wu, "Unanswerability evaluation for retrieval augmented generation," 2025. [Online]. Available: https://arxiv.org/abs/2412.12300

[9] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "Ragas: Automated evaluation of retrieval augmented generation," 2025. [Online]. Available: https://arxiv.org/abs/2309.15217

[10] N. Kandpal, H. Deng, A. Roberts, E. Wallace, and C. Raffel, "Large language models struggle to learn long-tail knowledge," 2023. [Online]. Available: https://arxiv.org/abs/2211.08411

[11] M. Klesel and H. F. Wittmann, "Retrieval-augmented generation (rag)," *Business & Information Systems Engineering*, vol. 67, pp. 551–561, 2025. doi: 10.1007/s12599-025-00945-3. [Online]. Available: https://doi.org/10.1007/s12599-025-00945-3

[12] Y. Gao, Y. Xiong, M. Wang, and H. Wang, "Modular rag: Transforming rag systems into lego-like reconfigurable frameworks," 2024. [Online]. Available: https://arxiv.org/abs/2407.21059

[13] Z. Chu, J. Chen, Q. Chen, H. Wang, K. Zhu, X. Du, W. Yu, M. Liu, and B. Qin, "Beamaggr: Beam aggregation reasoning over multi-source knowledge for multi-hop question answering," 2024. [Online]. Available: https://arxiv.org/abs/2406.19820

[14] C. Niu, Y. Wu, J. Zhu, S. Xu, K. Shum, R. Zhong, J. Song, and T. Zhang, "Ragtruth: A hallucination corpus for developing trustworthy retrieval-augmented language models," 2024. [Online]. Available: https://arxiv.org/abs/2401.00396

[15] Q. Zhang, Z. Xiang, Y. Xiao, L. Wang, J. Li, X. Wang, and J. Su, "Faithfulrag: Fact-level conflict modeling for context-faithful retrieval-augmented generation," 2025. [Online]. Available: https://arxiv.org/abs/2506.08938

[16] Á. Kovács and G. Recski, "Lettucedetect: A hallucination detection framework for rag applications," 2025. [Online]. Available: https://arxiv.org/abs/2502.17125

[17] Y. Li, X. Fu, G. Verma, P. Buitelaar, and M. Liu, "Mitigating hallucination in large language models (llms): An application-oriented

survey on rag, reasoning, and agentic systems," 2025. [Online]. Available: https://arxiv.org/abs/2510.24476

[18] O. Ayala and P. Bechard, "Reducing hallucination in structured outputs via retrieval-augmented generation," in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, Y. Yang, A. Davani, A. Sil, and A. Kumar, Eds. Mexico City, Mexico: Association for Computational Linguistics, Jun. 2024. doi: 10.18653/v1/2024.naacl-industry.19 pp. 228–238. [Online]. Available: https://aclanthology.org/2024.naacl-industry.19/

[19] A. Azaria and T. Mitchell, "The internal state of an llm knows when it's lying," 2023. [Online]. Available: https://arxiv.org/abs/2304.13734

[20] Z. Wang, H. Yuan, W. Dong, G. Cong, and F. Li, "Carrot: A learned cost-constrained retrieval optimization system for rag," 2026. [Online]. Available: https://arxiv.org/abs/2411.00744

[21] Z. Jiang, X. Ma, and W. Chen, "Longrag: Enhancing retrieval-augmented generation with long-context llms," 2024. [Online]. Available: https://arxiv.org/abs/2406.15319

[22] X. Wang, Z. Wang, X. Gao, F. Zhang, Y. Wu, Z. Xu, T. Shi, Z. Wang, S. Li, Q. Qian, R. Yin, C. Lv, X. Zheng, and X. Huang, "Searching for best practices in retrieval-augmented generation," 2024. [Online]. Available: https://arxiv.org/abs/2407.01219

[23] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro, and Y. Zhang, "Sparks of artificial general intelligence: Early experiments with gpt-4," 2023. [Online]. Available: https://arxiv.org/abs/2303.12712

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[25] F. Yu, H. Zhang, P. Tiwari, and B. Wang, "Natural language reasoning, a survey," 2023. [Online]. Available: https://arxiv.org/abs/2303.14725

[26] T. Zhang, F. Ladhak, E. Durmus, P. Liang, K. McKeown, and T. B. Hashimoto, "Benchmarking large language models for news

summarization," *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 39–57, Jan. 2024. doi: 10.1162/tacl_a_00632. [Online]. Available: https://doi.org/10.1162/tacl_a_00632

[27] OpenAI, "Gpt-4 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

[28] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "Llama: Open and efficient foundation language models," 2023. [Online]. Available: https://arxiv.org/abs/2302.13971

[29] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mistral 7b," 2023. [Online]. Available: https://arxiv.org/abs/2310.06825

[30] S.-Q. Yan, J.-C. Gu, Y. Zhu, and Z.-H. Ling, "Corrective retrieval augmented generation," 2024. [Online]. Available: https://arxiv.org/abs/2401.15884

[31] P. Zhao, H. Zhang, Q. Yu, Z. Wang, Y. Geng, F. Fu, L. Yang, W. Zhang, J. Jiang, and B. Cui, "Retrieval-augmented generation for ai-generated content: A survey," *Data Science and Engineering*, vol. 11, pp. 1–29, 01 2026. doi: 10.1007/s41019-025-00335-5

[32] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, "The faiss library," 2025. [Online]. Available: https://arxiv.org/abs/2401.08281

[33] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. tau Yih, "Dense passage retrieval for open-domain question answering," 2020. [Online]. Available: https://arxiv.org/abs/2004.04906

[34] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," 2019. [Online]. Available: https://arxiv.org/abs/1908.10084

[35] Z. Yang, S. Chen, C. Gao, Z. Li, X. Hu, K. Liu, and X. Xia, "An empirical study of retrieval-augmented code generation: Challenges and

opportunities," 2025. [Online]. Available: https://arxiv.org/abs/2501.13742

[36] W. Yu, H. Zhang, X. Pan, K. Ma, H. Wang, and D. Yu, "Chain-of-note: Enhancing robustness in retrieval-augmented language models," 2024. [Online]. Available: https://arxiv.org/abs/2311.09210

[37] Y. Tang and Y. Yang, "Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries," 2024. [Online]. Available: https://arxiv.org/abs/2401.15391

[38] Z. Rackauckas, "Rag-fusion: A new take on retrieval augmented generation," *International Journal on Natural Language Computing*, vol. 13, no. 1, p. 37–47, Feb. 2024. doi: 10.5121/ijnlc.2024.13103. [Online]. Available: http://dx.doi.org/10.5121/ijnlc.2024.13103

[39] D. Kim, B. Kim, D. Han, and M. Eibich, "Autorag: Automated framework for optimization of retrieval augmented generation pipeline," 2024. [Online]. Available: https://arxiv.org/abs/2410.20878

[40] J. Jin, X. Wang, C. Fan, L. Cheng, Y. Wang, Y. Ruan, F. Zhai, X. Fang, Z. Wang, Y. Zhao, F. Xue, C. Wang, F. Zhang, F. Wang, Y. Song, X. Wang, X. Song, W. Wang, X. Wang, Z. Bai, S. Yu, Z. Li, J. Liu, J. Xue, Z. Fei, Z. Du, Y. Ma, J. Li, T. Zhang, P. Li, M. Sun, Y. Ji, W. Jin, S. Tang, Y. Zhang, X. Zheng, and J. Li, "Flashrag: A modular toolkit for efficient retrieval-augmented generation," 2024. [Online]. Available: https://arxiv.org/abs/2405.13576

[41] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-rag: Learning to retrieve, generate, and critique through self-reflection," 2023. [Online]. Available: https://arxiv.org/abs/2310.11511

[42] X. Zhao, S. Liu, S.-Y. Yang, and C. Miao, "Medrag: Enhancing retrieval-augmented generation with knowledge graph-elicited reasoning for healthcare copilot," 2025. [Online]. Available: https://arxiv.org/abs/2502.04413

[43] G. Xiong, Q. Jin, Z. Lu, and A. Zhang, "Benchmarking retrieval-augmented generation for medicine," 2024. [Online]. Available: https://arxiv.org/abs/2402.13178

[44] J. Miao, C. Thongprayoon, S. Suppadungsuk, O. A. Garcia Valencia, and W. Cheungpasitporn, "Integrating retrieval-augmented generation with

large language models in nephrology: Advancing practical applications," *Medicina*, vol. 60, no. 3, 2024. doi: 10.3390/medicina60030445. [Online]. Available: https://www.mdpi.com/1648-9144/60/3/445

[45] Z. Li, Z. Wang, W. Wang, K. Hung, H. Xie, and F. L. Wang, "Retrieval-augmented generation for educational application: A systematic survey," *Computers and Education: Artificial Intelligence*, vol. 8, p. 100417, 2025. doi: https://doi.org/10.1016/j.caeai.2025.100417. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666920X25000578

[46] N. Mayat, C. Wachter, S. Spatzenegger, M. P. Hinrichs, T. Weißer, and R. H. Schmitt, "Performance of rag-based systems in industrial organizations: a case study in the automotive industry," in *2025 IEEE 8th International Conference on Industrial Cyber-Physical Systems (ICPS)*, 2025. doi: 10.1109/ICPS65515.2025.11087842 pp. 1–6.

[47] Y. Hu, Z. Lei, Z. Zhang, B. Pan, C. Ling, and L. Zhao, "Grag: Graph retrieval-augmented generation," 2025. [Online]. Available: https://arxiv.org/abs/2405.16506

[48] X. He, Y. Tian, Y. Sun, N. V. Chawla, T. Laurent, Y. LeCun, X. Bresson, and B. Hooi, "G-retriever: Retrieval-augmented generation for textual graph understanding and question answering," 2024. [Online]. Available: https://arxiv.org/abs/2402.07630

[49] R. Zhao, H. Chen, W. Wang, F. Jiao, X. L. Do, C. Qin, B. Ding, X. Guo, M. Li, X. Li, and S. Joty, "Retrieving multimodal information for augmented generation: A survey," 2023. [Online]. Available: https://arxiv.org/abs/2303.10868

[50] A. L. Smith, F. Greaves, and T. Panch, "Hallucination or confabulation? neuroanatomy as metaphor in large language models," *PLOS Digital Health*, vol. 2, no. 11, p. e0000388, 2023. doi: 10.1371/journal.pdig.0000388. [Online]. Available: https://doi.org/10.1371/journal.pdig.0000388

[51] W. Hu, W. Zhang, Y. Jiang, C. J. Zhang, X. Wei, and L. Qing, "Removal of hallucination on hallucination: Debate-augmented RAG," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, Eds. Vienna,

Austria: Association for Computational Linguistics, Jul. 2025. doi: 10.18653/v1/2025.acl-long.770. ISBN 979-8-89176-251-0 pp. 15 839–15 853. [Online]. Available: https://aclanthology.org/2025.acl-long.770/

[52] J. Chen, H. Lin, X. Han, and L. Sun, "Benchmarking large language models in retrieval-augmented generation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 16, pp. 17 754–17 762, Mar. 2024. doi: 10.1609/aaai.v38i16.29728. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/29728

[53] L. Siagian, "Benchmarking hallucination evaluation for rag under an abstention policy: A controlled 30-query study with ragas, deepeval, and llm-as-judge," 01 2026.

[54] Z. Sun, X. Zang, K. Zheng, Y. Song, J. Xu, X. Zhang, W. Yu, Y. Song, and H. Li, "Redeep: Detecting hallucination in retrieval-augmented generation via mechanistic interpretability," 2025. [Online]. Available: https://arxiv.org/abs/2410.11414

[55] D. Ru, L. Qiu, X. Hu, T. Zhang, P. Shi, S. Chang, C. Jiayang, C. Wang, S. Sun, H. Li, Z. Zhang, B. Wang, J. Jiang, T. He, Z. Wang, P. Liu, Y. Zhang, and Z. Zhang, "Ragchecker: A fine-grained framework for diagnosing retrieval-augmented generation," 2024. [Online]. Available: https://arxiv.org/abs/2408.08067

[56] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, "Active retrieval augmented generation," 2023. [Online]. Available: https://arxiv.org/abs/2305.06983

[57] Z. Guo, L. Xia, Y. Yu, T. Ao, and C. Huang, "Lightrag: Simple and fast retrieval-augmented generation," 2025. [Online]. Available: https://arxiv.org/abs/2410.05779

[58] J. Song, X. Wang, J. Zhu, Y. Wu, X. Cheng, R. Zhong, and C. Niu, "RAG-HAT: A hallucination-aware tuning pipeline for LLM in retrieval-augmented generation," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, F. Dernoncourt, D. Preoţiuc-Pietro, and A. Shimorina, Eds. Miami, Florida, US: Association for Computational Linguistics, Nov. 2024. doi: 10.18653/v1/2024.emnlp-industry.113 pp. 1548–1558. [Online]. Available: https://aclanthology.org/2024.emnlp-industry.113/

[59] Z. Wei, W.-L. Chen, and Y. Meng, "Instructrag: Instructing retrieval-augmented generation via self-synthesized rationales," 2025. [Online]. Available: https://arxiv.org/abs/2406.13629

[60] O. Ovadia, M. Brief, M. Mishaeli, and O. Elisha, "Fine-tuning or retrieval? comparing knowledge injection in llms," 2024. [Online]. Available: https://arxiv.org/abs/2312.05934

[61] H. Soudani, E. Kanoulas, and F. Hasibi, "Fine tuning vs. retrieval augmented generation for less popular knowledge," in *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, ser. SIGIR-AP 2024. New York, NY, USA: Association for Computing Machinery, 2024. doi: 10.1145/3673791.3698415. ISBN 9798400707247 p. 12–22. [Online]. Available: https://doi.org/10.1145/3673791.3698415

[62] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[63] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007.

[64] BAAI, "Bge embedding: General embedding models for retrieval," 2023. [Online]. Available: https://huggingface.co/BAAI/bge-large-en-v1.5

[65] Q. Team, "Qwen technical report," 2023. [Online]. Available: https://arxiv.org/abs/2309.16609

[66] Microsoft, "Azure blob storage documentation," https://learn.microsoft.com/en-us/azure/storage/blobs/, accessed: 2026-05-04.

[67] Unstructured, "Partitioning," https://docs.unstructured.io/open-source/core-functionality/partitioning, accessed: 2026-05-04.

[68] ——, "Table extraction from pdf," https://unstructured.readthedocs.io/en/main/best_practices/table_extraction_pdf.html, accessed: 2026-05-04.

[69] Qdrant, "What is qdrant?" https://qdrant.tech/documentation/overview/what-is-qdrant/, accessed: 2026-05-04.

[70] Docker, "What is docker?" https://docs.docker.com/get-started/docker-overview/, accessed: 2026-05-04.

[71] Qdrant, "Similarity search," https://qdrant.tech/documentation/search/search/, accessed: 2026-05-04.

[72] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 3, pp. 333–389, 09 2009. doi: 10.1561/1500000019

[73] Sentence Transformers, "Cross-encoders," https://sbert.net/examples/cross_encoder/applications/README.html, accessed: 2026-05-04.

[74] Beijing Academy of Artificial Intelligence, "Baai/bge-reranker-base," https://huggingface.co/BAAI/bge-reranker-base, accessed: 2026-05-04.

[75] A. Williams, N. Nangia, and S. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, M. Walker, H. Ji, and A. Stent, Eds. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018. doi: 10.18653/v1/N18-1101 pp. 1112–1122. [Online]. Available: https://aclanthology.org/N18-1101/

# Appendix A

# Examples from the Evaluation Query Dataset

Table A.1 provides representative examples from the evaluation query dataset. The examples illustrate how answerable, ambiguous, and unanswerable queries were distinguished with respect to the bounded knowledge base. For answerable and ambiguous queries, chunk identifiers indicate the indexed chunks used as supporting references where applicable. For unanswerable queries, no chunk identifiers are assigned because the required evidence is not present in the indexed collection.

Table A.1: Representative Examples from the Evaluation Query Dataset

| Category | ID | Query | Source Document | Chunk IDs | Expected Answer or Rationale |
|---|---|---|---|---|---|
| Answerable | Q17 | How do different chunk sizes affect retrieval performance in a RAG system? | `searching_for_best_practices_RAG` | 27 | Chunk size significantly impacts performance: larger chunks provide more context but increase processing time, while smaller chunks improve retrieval recall and efficiency but may lack sufficient context. |
| Answerable | Q19 | Which four abilities are evaluated in the RGB benchmark for retrieval-augmented generation? | `benchmarking_LLMs_in_RAG` | 3, 49 | The four abilities evaluated are noise robustness, negative rejection, information integration, and counterfactual robustness. |

| | | | | | |
|---|---|---|---|---|---|
| Answerable | Q43 | What limitation in earlier RAG evaluation frameworks motivates unanswerability evaluation? | `unanswerability_evaluation_rag` | 2 | Existing evaluation frameworks for retrievalaugmented generation (RAG) systems focus on answerable queries, but they overlook the importance of appropriately rejecting unanswerable requests. |
| Ambiguous | A9 | Should a RAG system use top-$k$ retrieval or single retrieval for better overall quality? | `rag_for_ai_generated_content` | 181 | Using top-$k$ retrieval improves attribution (grounding) but harms fluency compared to single retrieval. The best choice depends on whether attribution or fluency is more important for the specific task. The provided context does not prescribe a universal winner. |

| | | | | | |
|---|---|---|---|---|---|
| Ambiguous | A17 | What chunk-size choice is usually best when trading off context completeness against retrieval speed? | `searching_for_best_practices_RAG` | 27 | Larger chunks provide more context but increase processing time, while smaller chunks improve retrieval recall and reduce time but may lack sufficient context. There is no single best size that works universally; the optimal choice depends on the specific balance between context completeness and speed required for the task. |
| Ambiguous | A7 | In retrieval-augmented LLM pipelines, which stage tends to dominate hallucination risk: retrieval or generation? | `hallucination_mitigation` | 1 | Both retrieval and generation phases can cause hallucinations; it does not claim one stage universally dominates across systems. |
| Unanswerable | U9 | In Modular RAG, what JSON schema is used to serialize interfaces between functional modules? | `modular_RAG_LEGO` | – | There is insufficient evidence in the provided context to answer this question. |

| | | | | |
|---|---|---|---|---|
| Unanswerable | U17 | In the six-dimension trustworthiness framework for RAG, what numeric weight is assigned to each dimension? | `trustworthiness_in_RAG_systems` | – | There is insufficient evidence in the provided context to answer this question. |
| Unanswerable | U31 | In FaithfulRAG, what tie-break rule is used when self-facts and retrieved facts conflict equally? | `faithful_rag` | – | There is insufficient evidence in the provided context to answer this question. |

# Appendix B

# Qualitative Error Examples

This appendix provides representative qualitative failure cases from the experiment logs. The examples support the error analysis in Chapter 5.

Table B.1: Representative false insufficiency cases from the BGE query expansion top-$k = 15$ run

| **Configuration** | **Query** | **Generated answer** | **Expected answer, shortened** |
|---|---|---|---|
| BGE query expansion top-$k = 15$ | What sources are used to build the RGB benchmark instances? | There is insufficient evidence in the provided context to answer this question. | The RGB benchmark instances are generated from latest news articles and external documents obtained from search engines. |

| Configuration | Query | Generated answer | Expected answer, shortened |
|---|---|---|---|
| BGE query expansion top-$k = 15$ | How does Debate-Augmented RAG try to mitigate hallucinations? | There is insufficient evidence in the provided context to answer this question. | DRAG mitigates hallucinations using multi-agent debate mechanisms, including retrieval debate and response debate. |
| BGE query expansion top-$k = 15$ | What limitation of RAG systems motivates the creation of RAGTruth? | There is insufficient evidence in the provided context to answer this question. | RAG systems may still present unsupported or contradictory claims to the retrieved contents. |
| BGE query expansion top-$k = 15$ | What are the two main components of a RAG system considered by RAGAs? | There is insufficient evidence in the provided context to answer this question. | A RAG system consists of a retrieval module and an LLM-based generation module. |

Table B.2: Representative ambiguous-query behaviours from the BGE query expansion top-$k = 15$ run

| **Behaviour** | **Query** | **Generated answer, shortened** | **Expected answer, shortened** |
|---|---|---|---|
| Appropriate uncertainty | In Debate-Augmented RAG, which stage is likely to contribute more to hallucination reduction: retrieval debate or response debate? | Both retrieval debate and response debate contribute to hallucination reduction, but the context does not state which contributes more overall. | The contribution varies by dataset; no single stage is universally dominant. |
| Appropriate uncertainty | How many multi-agent debate rounds are usually sufficient before extra rounds produce diminishing returns in DRAG-like workflows? | Usually 1–3 rounds are sufficient, with diminishing or negative returns after additional rounds depending on task complexity. | One or two rounds are often sufficient; additional rounds add cost without clear benefit, depending on the task. |
| Over-refusal | In multimodal RAG, which is more important: attribution to sources or fluency of generated text? | There is insufficient evidence in the provided context to answer this question. | The context describes an attribution–fluency trade-off; the optimal balance depends on application requirements. |

| **Behaviour** | **Query** | **Generated answer, shortened** | **Expected answer, shortened** |
|---|---|---|---|
| Over-refusal | Which pipeline stage is most likely to be optimal for modality fusion in multimodal retrieval-augmented generation? | There is insufficient evidence in the provided context to answer this question. | There is no unified perception of where or how to incorporate modalities; the optimal stage depends on architecture, modalities, and task. |
| Over-specific answering | Which category of unanswerable request is most challenging for RAG systems to handle appropriately? | The system answered that the underspecified category is the most challenging. | Six categories are defined, but no universal difficulty ranking is provided; difficulty depends on architecture, knowledge base, and evaluation criteria. |

| Behaviour | Query | Generated answer, shortened | Expected answer, shortened |
|---|---|---|---|
| Over-specific answering | Which document type is most challenging for text-only retrieval in RAG? | The system answered that visually heavy documents such as infographics are most challenging. | Difficulty varies across datasets and retrieval methods; the context does not identify one document type as universally most challenging. |

Table B.3: Representative unanswerable-query behaviours from the BGE query expansion top-$k = 15$ run

| Behaviour | Query | Generated answer, shortened | Expected answer |
|---|---|---|---|
| Correct abstention | In the enterprise RAG-Fusion chatbot pilot, what was the internal repository size in number of documents? | There is insufficient evidence in the provided context to answer this question. | There is insufficient evidence in the provided context to answer this question. |
| Correct abstention | In FlashRAG, what Docker base image digest is used for the official reproducible environment? | There is insufficient evidence in the provided context to answer this question. | There is insufficient evidence in the provided context to answer this question. |

| Behaviour | Query | Generated answer, shortened | Expected answer |
|---|---|---|---|
| Qualified answer with caveat | In CRAG, what confidence threshold triggers corrective retrieval? | The system explained that corrective retrieval is triggered below a lower confidence threshold, but stated that the exact numerical value is not specified. | There is insufficient evidence in the provided context to answer this question. |
| Qualified answer with caveat | In the controlled 30-query abstention benchmarking study, what is the exact list of the 30 evaluation queries? | The system stated that the dataset is referenced as containing 30 queries, but the exact list is not included in the provided context. | There is insufficient evidence in the provided context to answer this question. |

TRITA – XXX-EX 2025:0000
Stockholm, Sweden 2026

www.kth.se