Master Thesis

Optimizing Retrieval Augmented Generation Chatbots: A Comparative Analysis

for the Degree of

Master of Science

submitted to the Department of Mathematics, Natural Sciences, and Computer Science
at the Technische Hochschule Mittelhessen (University of Applied Sciences)

by

Thu Hang Nguyen

June 15, 2025

Referee: Sebastian Enns

Co-Referee: Prof. Dr. Peter Kneisel

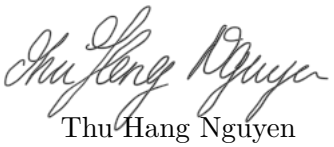
Declaration of the use of Generative AI

In accordance with the recommendation of the German Research Foundation (DFG - Deutsche Forschungsgemeinschaft)[1] and that of the journal Theoretical Computer Science[2] I (the author) herby declare the use of generative AI.

During the preparation of this work I used ChatGPT 4 in order to improve readability and language, only. After using ChatGPT 4, I reviewed and edited the content as needed and take full responsibility for the content of this thesis.

Declaration of Independence

I hereby declare that I have composed the present work independently and have not used any sources or aids other than those cited, and that all quotations have been clearly indicated.



Gießen, on June 15, 2025 Thu Hang Nguyen

1 DFG Formulates Guidelines for Dealing with Generative Models for Text and Image Creation: https://www.dfg.de/en/news/news-topics/announcements-proposals/2023/info-wissenschaft-23-72

2 Declaration of generative AI in scientific writing: https://www.sciencedirect.com/journal/theoretical-computer-science/publish/guide-for-authors

This thesis investigates the optimization of Retrieval Augmented Generation (RAG) chatbots with a focus on improving information preparation within the retrieval process. As part of an adapted Systematic Literature Review (SLR), the current state of research on retrieval methods and optimization approaches in the RAG context is systematically collected and analyzed. Based on this analysis, suitable methods are identified and classified. For the subsequent comparative evaluation of the selected retrieval approaches, an evaluation framework based on an adapted version of the Software Architecture Comparison Method (SACAM) is developed, which includes criteria such as document relevance, answer precision, response latency, and hallucination resistance. The implementation focuses on vector-based retrieval methods (e.g., dense retrieval and hierarchical variants) as well as supplementary optimization strategies such as multi-query rewriting and parent-child embedding to improve retrieval quality. The evaluation shows that the use of selected advanced retrieval strategies can lead to moderate improvements in answer quality and robustness depending on the application scenario, while limitations and challenges remain. Finally, the contributions of the thesis, existing limitations, and directions for future research are discussed.

Contents

1 Introduction

Chatbots are text- or voice-based communication interfaces that enable automated interactions between humans and machines. Their applications range from simple information bots and citizen services to internal enterprise tools designed to streamline processes and maintain productivity. In helpdesks and citizen services, they can reduce delays that often result from staff shortages. Early chatbots were statically programmed, identifying specific terms after text normalization and responding with predefined answers. In contrast, modern chatbots leverage Artificial Intelligence (AI) to handle complex queries. Trained on extensive datasets, they are capable of retaining conversational context and correctly handling time-independent queries [Koh20].

The integration of Large Language Models (LLMs) like ChatGPT marks a major milestone in chatbot technology. ChatGPT enables intuitive, conversational interactions, answering follow-up questions, identifying errors, correcting assumptions, and rejecting inappropriate requests [Tae23, Flo20, Dal21, Ope22]. LLMs have revolutionized natural language understanding and generation with human-like text generation, contextual awareness, and problem-solving capabilities. Widely used in search engines, customer support, and translation [Yao24], they enhance chatbots by improving response quality, handling complex queries, automating routine tasks and enabling domain-specific knowledge retrieval to boost efficiency.

However, LLMs have limitations. These models are trained on pre-existing datasets, which are often broad and generalized, rather than tailored to specific domains. Consequently, LLMs struggle to deliver precise and contextually appropriate responses to domain-specific inquiries, such as those encountered in fields like medicine, law or internal resources. A further critical challenge is the issue of outdated training data. Since LLMs are trained on static datasets collected at a fixed point in time, they are prone to generating responses based on obsolete or inaccurate information. This can result in outputs that are incomplete, incorrect or misleading, which is commonly referred to as *hallucinations* [Bar24].

One approach to overcoming these limitations is Retrieval Augmented Generation (RAG), which was introduced by Patrick Lewis et al. [Lew21] and has been shown to enhance LLM performance in Question Answering (QA) tasks [Roy24]. Unlike purely pre-trained LLMs, RAG combines a Language Model (LM) with an external knowledge

base, providing domain-specific and up-to-date information from fields like medicine, law or internal resources. This integration improves accuracy, enables precise answers to complex queries, and reduces *hallucinations* [Bar24]. Additionally, RAG avoids full retraining of LLMs, minimizing computational costs and ensuring sensitive data remains securely stored in the knowledge base instead of the model itself [Cuc24].

The use of RAG to enhance the response quality of LLMs has facilitated their integration into chatbot systems. The continuous advancement of these technologies has motivated many organizations to adopt RAG-based chatbots. However, practical implementation has revealed several challenges, including the complexity of system integration and instances of inaccurate responses. These challenges underscore the need for further refinement and adaptation. A more detailed discussion of these issues will be provided in the *Problem Definition* 1.1 section.

1.1 Problem Definition

As mentioned briefly in the introduction, RAG approach is associated with several challenges. Primary challenges have been outlined in '*Seven Failure Points When Engineering a RAG System*' [Bar24]. According to Scott Barnett et al., the following critical failure points (**FP**) have been highlighted:

FP1 Missing Content:
These refer to missing documents that cannot be utilized to respond to a query. Typically, when an LLM generates responses, it is instructed to formulate fallback replies such as 'Unfortunately, I cannot answer this question' when no relevant information is available. However, this mechanism does not always function reliably, and the system may still generate a response even when the necessary information is absent.

FP2 Missed the Top Ranked Documents:
Generated answers may be incorrect or incomplete even when the necessary information exists in the corpus. This typically occurs when relevant documents are not ranked among the Top-K results during retrieval, as only a limited number of documents are selected based on performance parameters.

For example, given the query 'In which city is the Eiffel Tower located?', the document 'The Eiffel Tower is located in Paris.' may not appear among the Top-K results. Instead, less relevant documents such as 'The Cologne Cathedral is located in Cologne.' or 'The Statue of Liberty is located in New York.' may be retrieved, leading the model to produce an incorrect response.

FP3 Not in Context - Consolidation strategy Limitations:
Another potential issue arises when relevant documents are retrieved from the knowledge base but are excluded from the final context used for response generation. This typically occurs when a large number of documents are retrieved, but some are discarded during the consolidation or filtering phase due to lower relevance scores. As a result, the final context provided to the model may lack critical information, leading to inadequate answers.

For example, consider the query 'In which city is the Eiffel Tower located?'. While the document 'The Eiffel Tower is located in Paris.' is initially retrieved, it may be filtered out during context consolidation in favor of seemingly more relevant but unrelated documents, such as 'The Cologne Cathedral is located in Cologne.' or 'The Statue of Liberty is located in New York.'. As a consequence, the model lacks the necessary information to answer correctly.

FP4 Not Extracted:
In this case, the answer is contained in the context, but the LLM is unable to extract the correct answer. This may occur when the context is overwhelmed by excessive noise, such as irrelevant documents that lack any connection to the query or even contain contradictory information [Cuc24].

For example, consider again the query 'In which city is the Eiffel Tower located?'. Although, the context includes the relevant document 'The Eiffel Tower is located in Paris.', it is mixed with numerous unrelated or misleading documents such as 'The Sydney Opera House is located in ...' and '... located ...'. Due to this overwhelming noise, the model may fail to extract the correct answer, even though the information is present.

FP5 Wrong Format:
For instance, when a question requires extracting information in a specific format, such as a table or list, LLMs sometimes fail to follow these instructions. For example, if the user requests 'List the cities where the Eiffel Tower and the Statue of Liberty are located', the model may generate a free-text paragraph instead of producing the expected list format.

FP6 Incorrect Specificity:
Typically, RAG systems are designed for specific applications, such as educational scenarios in a teacher-student setting, where responses should be appropriately tailored for learning purposes. However, the LLM may occasionally generate responses that are either overly general or excessively detailed, thereby failing to adequately address the user's information need. Issues with specificity may also arise when the query itself is too broad or insufficiently formulated by the user.

FP7 Incomplete:

In this case, the response is not incorrect but incomplete, even though the necessary information is present in the context and could, in principle, be extracted. This issue frequently occurs when answering questions that require the aggregation of information from multiple sources distributed across different documents. For example, queries such as 'What are the key points covered in documents A, B, and C?' are typically answered more reliably when the system is able to process and consolidate information from each document individually.

Solutions have already been developed for some of these issues. Yunfan Gao et al. provide a comprehensive overview of the current state of research in their survey '*Retrieval-Augmented Generation for Large Language Models: A Survey*' [Gao24]. The specific approaches will be examined in detail later.

RAG systems rely on two fundamental core components: a *retriever* and a *generator*. The *retriever* interacts with an external information retrieval (IR) system to identify and retrieve relevant passages or documents, then passes the selected results to the generator component [Cuc24]. The retriever process can be divided into four stages: indexing, pre-retrieval, retrieval, and post-retrieval [Gao24]. The *generator*, often based on LLMs, uses the retrieved information along with the query to generate an answer [Cuc24]. Consequently, FPs can arise in either the retriever or the generator components. For further research, it is essential to categorize the aforementioned FPs into two groups:

C1: Challenges in the preparation of information (Indexing, Pre-Retrieval, Retrieval and Post-Retrieval) (FP2, FP3, FP7)

C2: LLM Failures (FP1, FP4, FP5, FP6)

It is important to acknowledge that **FP4** could be classified into both categories. While it is described as a failure associated with the LLM, it may also result from issues in the retriever process, such as errors in retrieving relevant documents or problems with ranking the retrieved information. The potential FPs in the retriever and generator components affect the quality of the responses generated by the RAG-based chatbot.

1.2 Research Objectives and Questions

This thesis is conducted in collaboration with a leading company and focuses on optimizing a RAG-based chatbot. Here is a brief summary of the key points discussed in the section *Problem Definition* 1.1:

- **Seven critical failure points (FPs)** have been identified by Scott Barnett et al. when using RAG systems
- Based on these findings, the FPs have been categorized into **two primary groups**
- First category (**C1**): **Challenges in Information Preparation**, relates to the retrieval process and includes four stages: **indexing, pre-retrieval, core-retrieval, and post-retrieval**
- Second category (**C2**): **LLM Failures**, pertains to the generator component

The primary focus of this thesis is on **C1**, specifically the optimization of the information preparation process for enhanced response generation. **Category C1** is a critical factor in the overall performance of RAG models, directly influencing the quality of the chatbot's responses. Yunfan Gao et al. provide a comprehensive overview of the current state of RAG, presenting various optimization approaches [Gao24]. The objective of this thesis is to investigate how improvements in this area can enhance the performance and reliability of RAG-based chatbots in practical applications. This leads to the following main research question:

RQ1: Which of the existing RAG approaches focused on information preparation hold the greatest potential to optimize the performance of RAG chatbots while enhancing the reliability and quality of their responses in practical applications?

To address the main research question as effectively as possible, the following sub-research question should also be considered:

RQ2: Which RAG approaches with a focus on information preparation are currently known and which of them are suitable for optimizing RAG chatbots in practical applications?

RQ3: What evaluation metrics should be in place to assess and compare RAG approaches in terms of their performance and reliability?

RQ4: How do the selected RAG approaches perform against the defined evaluation criteria?

RQ5: What impact do these RAG approaches have on the resulting response quality, particularly in terms of accuracy, relevance, and coherence?

1.3 Methodology

A comparative analysis was conducted to explore optimization strategies for RAG chatbots. To address research question **RQ2**, a Systematic Literature Review (SLR) was conducted, drawing on the principles of systematic reviews as outlined by Kitchenham [Kit07]. A full implementation of a formal systematic review protocol was not pursued. Instead, the review focused on identifying recent and methodologically relevant publications related to RAG, LLMs and retrieval strategies in QA contexts.

Sources included arXiv, the ACL Anthology, Google Scholar, and major conference proceedings such as NeurIPS, EMNLP, and ICLR. The selection was guided by the following criteria:

- **Recency:** primarily publications from 2022 onward,
- **Practical Relevance:** relation to RAG-based QA systems,
- **Technical Substance:** introduction of new modules, interaction mechanisms, or evaluation techniques,
- **Visibility:** citation frequency or integration into frameworks such as LangChain or LlamaIndex.

The objective was not comprehensive coverage but rather a broad representation of key developments in RAG optimization. The selected approaches were subsequently classified and form the conceptual basis for the comparative analysis.

To evaluate the selected approaches (**RQ3**–**RQ5**), the Software Architecture Comparison Method (SACAM) by Stoermer et al. [Sto03] was adapted. As SACAM was originally developed for complete software architectures, only selected phases such as criteria derivation, metric definition, and comparative assessment were applied and specifically adapted to the context of RAG-based QA systems. This adaptation enables a structured and transparent evaluation of the retrieval and optimization strategies under investigation. The main research question (**RQ1**) is addressed through this comparative assessment. A visual summary of the methodological approach is provided in Figure 1.1.

Based on the identified approaches and derived evaluation criteria, three representative retrieval pipelines focusing on information preparation were designed and implemented. Each implementation integrates different retrieval strategies and optimization techniques to address specific aspects of the identified failure points.

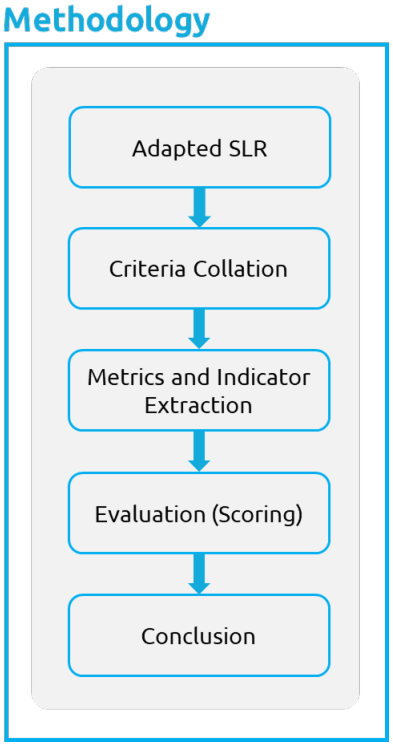


Figure 1.1: Methodology based on adapted SLR [Kit07]) and SACAM [Sto03]

The first implemented approach represents a baseline RAG pipeline (‘*Naive RAG*’), employing a standard retrieval-augmented generation framework combined with the applied chunking strategy introduced for consistent passage segmentation across all implementations. This baseline serves as a reference point for the comparative evaluation against the subsequent optimization variants. The results of these implementations are systematically evaluated using the adapted SACAM framework described above.

1.4 Scope and Delimitations

Since RAG has only been researched for a few years, the scientific literature is still limited. Due to resource and data protection constraints, this study relies solely on publicly available datasets and focuses primarily on the retriever component. More detailed limitations are discussed later in this thesis (see Section 6.3).

2 Background

After presenting the general motivation and problem definition, this chapter reviews the relevant foundations of AI, including Machine Learning (ML), Deep Learning (DL), Natural Language Processing (NLP), LLMs, Generative Artificial Intelligence (GenAI) and RAG, with a particular emphasis on chatbot systems.

2.1 Chatbots

Chatbots are commonly employed on websites, messaging platforms, and smartphone applications to handle user interactions. They are particularly useful for addressing frequently asked questions and providing basic information. Another advantage is their constant availability, allowing chatbots to handle inquiries 24/7 and thereby improve efficiency and service quality [Koh20]. The following sections provide an overview of chatbot types and the methods used to generate responses.

2.1.1 Chatbot Types

Various approaches exist to categorize chatbots based on their functional characteristics. Adamopoulou and Moussiades [Ada20] propose a classification that groups chatbots into several distinct types according to their intended use and interaction scope. The main categories are summarized in Table 2.1.

2.1.2 Response Generation Methods

Independent of their functional type, chatbots differ in how they process user input and generate responses. Three main approaches can be distinguished [Ada20]:

- **Rule-based Models:** rely on predefined rules and scripted responses. They are typically used for simple and predictable interactions.
- **Retrieval-based Models:** select appropriate responses from a predefined repository of possible answers based on input similarity.

Table 2.1: Overview of Chatbot Types

Category	Subcategory	Description
Knowledge Domain	**Open Domain Chatbots**	Capable of discussing a wide range of topics and responding flexibly to diverse user inputs.
	Closed Domain Chatbots	Focused on a specific domain and optimized for specialized tasks within that area.
Service Provided	**Interpersonal Chatbots**	Assist users in performing tasks such as booking flights or answering FAQs; often integrated into messaging platforms.
	Inter-Agent Chatbots	Enable communication and cooperation between multiple bots or agents to jointly manage tasks or exchange information.
Goal-Oriented Design	**Informative Chatbots**	Deliver information from pre-defined and static data sources.
	Conversational Chatbots	Designed to sustain natural conversations.
	Task-Based Chatbots	Focus on executing specific tasks or workflows.
Human-Aid	**Human-Aided Chatbots**	Allow human operators to intervene and improve response quality when automated systems reach their limits.

- **Generative Models:** generate responses dynamically using ML techniques, particularly neural networks and LLMs.

2.2 Artificial Intelligence

AI is a subfield of computer science with the capability to solve problems autonomously [Car83]. It can be broadly categorized into two main types [Sea80]:

Strong AI: Refers to a programmed computer that possesses cognitive states. It is not merely a tool for testing psychological theories but rather constitutes an explanation in itself, aiming to replicate and account for human cognition.

Weak AI: Serves as a tool for simulating and analyzing cognitive processes. It facilitates the formulation and testing of hypotheses in a structured and systematic manner.

Over the years, **Weak AI**, also known as **Narrow AI**, has been more precisely defined. It refers to programs that exhibit intelligence within a specific domain [Bux21, Pen07].

In recent years, research and development have increasingly focused on ML, a subset of **Weak/Narrow AI**, which enables adaptive and data-driven decision-making [Bux21].

2.2.1 Machine Learning

ML refers to a set of methods which automatically detect patterns by training an algorithm to make predictions or decisions based on data [Mur12]. There are three types of machine learning:

Supervised Learning: Aims to learn a mapping from inputs to desired outputs. A common task in this category is classification, where the algorithm is trained to approximate a function that maps input vectors to one of several classes by using *labeled* input-output examples [Cun08, Nas17, Mur12].

Unsupervised Learning: Focuses on discovering hidden structures or patterns in the data, a process often referred to as knowledge discovery. Unlike supervised learning, it operates on *unlabeled* data, meaning there are no predefined outputs. These algorithms identify similarities within the data and group them into clusters [Nas17, Mur12].

Reinforcement Learning: Involves *learning a strategy or policy* to solve a specific task through interaction with an environment. Each action taken affects the environment and yields feedback in the form of rewards or penalties, which guide the learning process over time [Nas17, Bux21].

Supervision in ML generally refers to the use of labeled data for training models but in weakly supervised or self-supervised learning, the model can learn from indirect signals, such as patterns in data or outputs from other models, rather than explicit labels.

One of the prominent methods within ML is the use of Neural Networks, which are inspired by the information processing mechanisms of the human brain. A Neural Network is a computational model composed of numerous interconnected units (or *neurons*), each performing a specific operation, typically an activation function. The connections between these neurons represent the transmission of signals, which can be interpreted as the 'memory' of the network, influencing its ability to learn from data [Bul93, Wu18]. This method serves as the foundation for numerous ML tasks, including DL and NLP.

2.2.2 Deep Learning

The technique behind DL is the use of Deep Neural Networks (DNNs), which are powerful models that have achieved remarkable performance on complex learning tasks. Until 2014, DNNs were mainly effective when large labeled datasets were available. However, they struggled with tasks involving the mapping of input sequences to output sequences. Sutskever et al. addressed this limitation by introducing the *sequence-to-sequence (seq2seq)* framework based on an Encoder-Decoder architecture, using Long Short-Term Memory (LSTM) Recurrent Neural Networks (RNNs) to enable complex sequence transformation tasks such as machine translation [Sut14].

- **Encoder:** Transforms an input sequence of variable length into a fixed length vector representation [Cho14].
- **Decoder:** Uses this vector representation to generate an output sequence of variable length [Cho14].

Extensions such as attention mechanisms [Bah14] and the Transformer model [Vas17] have significantly improved the effectiveness of this approach, making it foundational to modern NLP and GenAI applications.

2.2.3 Natural Language Processing

NLP is a field of AI concerned with enabling computers to process and interact with human language in a meaningful way [Hir15]. The field of NLP often involves the use of manually crafted regular expressions and complex logical rules. These patterns are typically employed to either search for sequences within documents and rank them based on frequency or, in the case of chatbots, to generate scripted responses (answers) to given sequences (questions). NLP encompasses two main components: the conversion of raw input text into a numerical format (such as vectors or matrix) and the development of models designed to process this numerical data [Pat23]. The conversion of raw input text into a numerical format involves several key steps:

Tokenization: The process of splitting textual data into smaller units such as words, sentences, or symbols. A *token* can be defined as a non-empty contiguous sequence of graphemes (letters) or phonemes (articulated sounds) in a text [Pul51, Mie21]. There are different methods of *tokenization*, depending on the level of granularity and the linguistic or computational goals.

- **Word-level tokenization (traditionally):** Typically relies on typographic separators such as whitespace and punctuation marks to define tokens [Mie21].

- **Subword-level tokenization:** Refers to the segmentation of text into smaller units that are not defined by traditional word boundaries. These sub-lexical units are typically not motivated by typographic or linguistic rules and are often used in modern NLP models. Such tokenization strategies are commonly employed in **modern LMs** using methods like Byte Pair Encoding (BPE) or WordPiece [Mie21, Sen16, Dev19].

2.2.4 Large Language Model

LLMs represent a class of DNNs (see Section 2.2.2) trained primarily through self-supervised learning objectives. In contrast to supervised learning (see Section 2.2.1), which requires manually labeled datasets, LLMs are pretrained using self-supervised learning tasks that are automatically derived from large corpora of unannotated text. Self-supervised learning tasks are typically designed to force the model to predict parts of its own input, thereby enabling the model to learn rich internal representations and general linguistic patterns from extensive corpora. This approach allows for highly scalable training procedures and facilitates the development of models capable of handling a broad range of downstream tasks [Bom21].

Modern LLMs are predominantly based on the Transformer architecture, which utilizes an encoder-decoder framework to model complex sequence dependencies [Vas17]. In this architecture, the encoder maps an input sequence of symbol representations into a sequence of continuous vector representations. The decoder then generates an output sequence of symbols one element at a time based on these continuous representations. Both encoder and decoder consist of multiple stacked layers incorporating self-attention mechanisms and position-wise fully connected layers. The attention mechanism can be formally described as a function that maps a query and a set of key-value pairs to an output, where queries, keys, values, and outputs are all vectors.

Prominent LLM architectures include encoder-only models such as BERT [Dev19], which are commonly applied in retrieval and representation learning tasks. Decoder-only models such as GPT-3 [Bro20] are primarily employed for generative tasks, while encoder-decoder models such as T5 [Raf20] combine both capabilities and support a wide variety of tasks, including text generation and RAG.

2.2.5 Generative Artificial Intelligence

GenAI refers to systems that are capable of generating new content based on learned patterns from data. In recent years, advances in NLP (see Section 2.2.3) have enabled these models to better understand task requirements through prompt-based learning,

often reducing the need for extensive fine-tuning. GenAI models typically accept specific raw data modalities as input, such as text or images, and generate outputs within the same modality. Most of these models are based on Transformer architectures (see Section 2.2.4). In addition to unimodal models, multimodal approaches combine multiple technologies and can process inputs from different modalities, for example, by accepting both images and text to generate textual outputs, as applied in tasks such as technical report generation. One practical application of GenAI is its integration into chatbot systems, such as ChatGPT, which employ large-scale LMs to generate conversational responses [Cao23, GB23].

2.3 Types of Databases as Knowledge Bases

In the context of AI applications, various types of databases can serve as knowledge bases depending on the nature of the data and the specific requirements of the system. This section provides a brief overview of the most common database types that are relevant in AI-driven knowledge retrieval and processing tasks.

Vector Databases. Store high-dimensional vector representations that encode raw data such as text, images, audio, or video. The dimensionality reflects the data's complexity and granularity. These representations are generated by embedding models, often based on transformer architectures or alternative feature extraction methods. Vector databases enable efficient similarity search and retrieval, particularly for complex and unstructured data, and support large-scale, real-time processing, making them essential for AI-driven systems [Han23]. Additional details on embedding generation and indexing are provided in Section 2.4.

Graph Databases. Represent data and schemas as graphs or generalized structures such as hypergraphs or hypernodes [Ang08]. Data access is performed through graph operations, including neighborhood queries, pattern matching, and connectivity analysis. Integrity constraints ensure data consistency across schema-instance relationships, identities, and functional dependencies. Graph databases are well-suited for domains where relational structures and interconnectivity are central, such as hypertext systems and geographic information systems. In AI contexts, knowledge graphs increasingly serve as structured indices to enhance retrieval and improve generative outputs [Edg24].

Relational Databases. Organize data as collections of tables, also referred to as relations. Within these tables, each row represents a fact that generally corresponds to a real-world entity or relationship, while the table and column names provide the semantic

context necessary to interpret the stored values. Relational databases are primarily employed for managing structured data. Data retrieval and manipulation in these systems are performed using the high-level declarative query language SQL [Elm10]. There are existing approaches and experimental studies that investigate the integration of relational databases as knowledge bases within AI systems [Li23].

2.4 Retrieval-Augmented Generation

RAG is a method first introduced by Patrick Lewis et al. [Lew21] to enhance the performance of LLMs in QA tasks [Roy24]. Unlike purely pre-trained LLMs, RAG integrates a LM with an external knowledge base containing specific and up-to-date information. This knowledge base may include domain-specific content from fields such as medicine, law, education or internal company resources. By incorporating external knowledge, RAG extends the capabilities of LLMs, enabling them to generate more precise and contextually relevant responses. Another advantage of RAG is its ability to *reduce hallucinations*, which refer to instances where the model generates incorrect or misleading information [Bar24]. Additionally, RAG eliminates the need for full model retraining when new data becomes available. This not only minimizes computational overhead but also ensures the confidentiality of sensitive or internal information. Instead of embedding such data directly into the model, it remains securely stored in the knowledge database and is retrieved only when necessary [Cuc24]. The following sections provide a detailed examination of the RAG pipeline, outlining its structure and key processes.

2.4.1 Fundamentals of RAG Pipeline

This subchapter begins with a fundamental explanation of the RAG Pipeline in order to establish a general understanding. The RAG Pipeline consists of three core components: *Indexing*, *Retrieval* and *Generation*. Their respective functions are described below:

Indexing: Documents such as PDF, HTML, Word, or Markdown files are transformed into a uniform plain text format after cleaning and extraction. Due to the context limitations of LMs, the text is split into smaller, more manageable chunks. These chunks are then encoded into vector representations using an embedding model and stored in a vector database for efficient similarity-based retrieval. This step is crucial for similarity searches in the subsequent *retrieval phase* [Gao24]. The indexing process is visualized in Figure 2.1. Alternative retrieval mechanisms may leverage graph databases for relation-based retrieval or relational databases for

structured fact retrieval. However, this thesis focuses on dense retrieval utilizing vector databases.

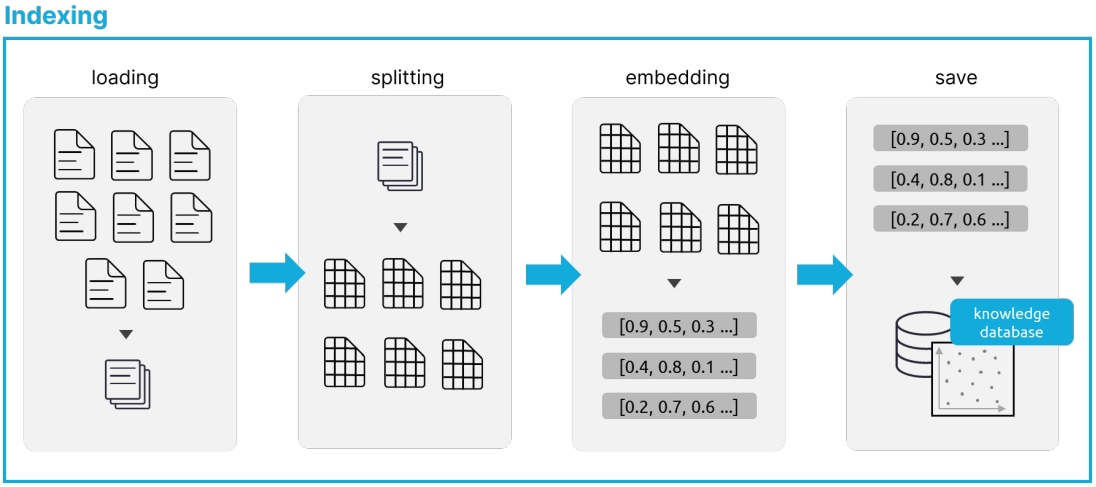


Figure 2.1: Illustration of the indexing process (based on the LangChain documentation [Lan])

Retrieval: The retrieval phase begins by encoding the user query using the same embedding model that was applied during the *indexing phase*. This ensures consistency in the vector space representation. To determine relevance, similarity scores are computed between the query vector and the indexed chunk vectors. Based on these scores, the system ranks and retrieves the top-K chunks that exhibit the highest semantic similarity to the query. These top-ranked chunks serve as an expanded context within the prompt, providing additional relevant information to enhance the response generation process [Gao24]. The retrieval process is visualized in Figure 2.2.

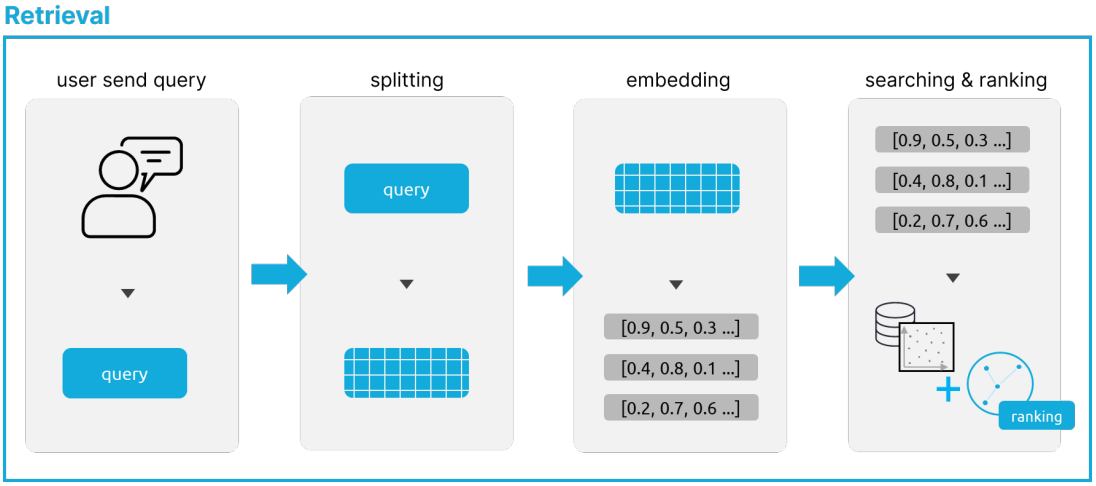


Figure 2.2: Illustration of the retrieval process

Generation: The user query and retrieved documents are synthesized into a coherent prompt and processed by the LLM to generate a response based on the available

information. The LLM's response generation is determined by the chosen approach and task-specific criteria, allowing it to either draw on its inherent parametric knowledge or rely exclusively on retrieved documents. In an ongoing dialogue, the conversational history can be integrated into the prompt, allowing the LLM to engage in multi-turn interactions by generating coherent, context-aware responses that reference previous statements [Gao24]. The generation process is visualized in Figure 2.3.

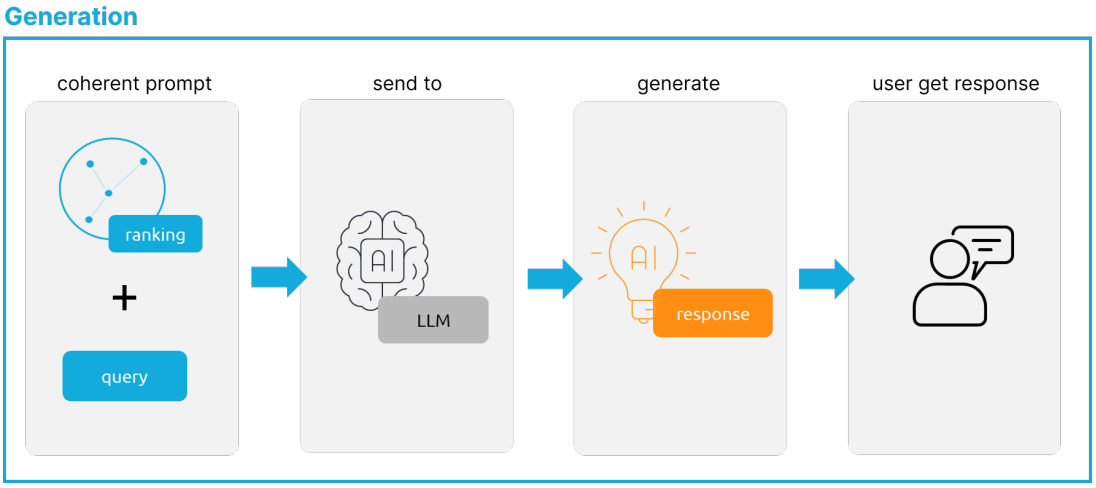


Figure 2.3: Illustration of the generation process

2.4.2 Paradigms of RAG

After describing the three core components of the RAG Pipeline, the focus now shifts to the three RAG paradigms. Understanding these paradigms is crucial for optimizing RAG applications and tailoring them to specific use cases:

Naive RAG: Follows the traditional process of RAG, which was described in subsection *Fundamentals of RAG Pipeline* (see Section 2.4.1). However, this paradigm has significant limitations [Gao24]:

- **Retrieval Challenges:** During the retrieval phase, it may struggle with precision and recall, potentially leading to the selection of misaligned or irrelevant chunks while also risking the exclusion of crucial information.
- **Augmented Generation Challenges:** Integrating retrieved information is challenging, as task-dependent variations may lead to disjointed or incoherent outputs. Redundancy from multiple sources can result in repetitive responses, while assessing relevance and maintaining consistency add complexity. A single retrieval step may lack sufficient context for complex queries, and

generation models often over-rely on augmented data, replicating retrieved content without substantive synthesis.

- **Generation Challenge:** During the response generation, the LLM may produce hallucinations if the generated content is not adequately supported by the retrieved context. Furthermore, responses may suffer from irrelevance, toxicity, or bias, which can compromise their quality and reliability.

Advanced RAG: A paradigm designed to overcome the limitations of **Naive RAG** by incorporating *pre-retrieval* and *post-retrieval* strategies to enhance retrieval quality. To address indexing challenges, it refines techniques through the use of a sliding window approach, fine-grained segmentation, and the integration of metadata. Additionally, it employs various optimization methods to streamline the retrieval process [Gao24].

Modular RAG: Enhances adaptability and flexibility by integrating diverse optimization strategies, supporting both sequential processing and end-to-end training. It improves retrieval through fine-tuning and specialized modules, such as advanced search mechanisms. By restructuring RAG modules and optimizing pipelines, it addresses specific challenges more effectively. While **Modular RAG** differs in its structure, it builds upon the foundational principles of both **Naive** and **Advanced RAG**, representing a progression and refinement within the RAG family [Gao24].

It is important to recognize that these three paradigms represent more of a progression and refinement within the RAG family. However, the choice of paradigm depends on the specific use case, as each may be better suited to different requirements.

2.4.3 Extensions of RAG Pipeline

There are two main approaches to extending the RAG pipeline: the first enhances retrieval through *pre-* and *post-retrieval* strategies (**Advanced RAG**), while the second improves adaptability and flexibility by integrating new *modules* and *patterns* (**Modular RAG**).

Pre-Retrieval: Primary focus includes [Gao24]:

- **Optimizing the indexing structure:** by improving data granularity, refining index structures, incorporating metadata, optimizing alignment and enabling mixed retrieval.

- **Enhancing the original query:** through techniques such as query rewriting, transformation, and expansion. These methods refine the user's input, making it clearer and better suited for the retrieval process.

Post-Retrieval: Focuses on effectively integrating the retrieved documents with the query to enhance the generation process. Key methods include [Gao24]:

- **Reranking Chunks:** repositioning the most relevant retrieved content to the beginning or end of the prompt for better contextual emphasis.
- **Context Compression:** extracting essential information, emphasizing critical sections and shortening the context to prevent information overload while ensuring optimal input for the generation process.

New Modules: Has specialized components to enhance retrieval and processing capabilities like [Gao24]:

- **Search Module:** enables direct searches across multiple data sources, including search engines, databases, and knowledge graphs.
- **(RAG) Fusion:** expands user queries using a multi-query strategy, leveraging parallel vector searches and intelligent re-ranking to uncover both explicit and implicit knowledge.
- **Memory Module:** creates a memory pool to better align text with data distribution, improving retrieval accuracy.
- **Routing:** directs queries through diverse data sources, selecting the optimal pathway for more relevant retrieval.
- **Predict Module:** reduces redundancy and noise by generating contextual information directly through the LLM.
- **Task Adapter Module:** adapts to various downstream tasks by automating prompt retrieval for zero-shot inputs and creating task-specific retrievers through few-shot query generation.

New Patterns: Enhance adaptability by enabling module substitution or reconfiguration to address specific challenges. These patterns go beyond the fixed *Retrieve* and *Read* structure, extending flexibility through the integration of new modules or adjustments in the interaction flow between existing components. This allows for greater customization and applicability across diverse tasks [Gao24].

3 Concept

This chapter provides the methodological and architectural basis for the evaluation and optimization of RAG systems. Key evaluation criteria are introduced, downstream tasks are classified, and architectural as well as retrieval strategies are examined. The chapter concludes with an evaluation framework specifically designed for QA, which serves as the primary use case.

3.1 Task Taxonomy and Derivation of Evaluation Criteria

To prepare the subsequent SACAM-based evaluation, this section identifies relevant goals, requirements, and evaluation criteria for the systematic assessment of RAG chatbot optimization. To derive these criteria, an initial classification of the RAG system is conducted based on the characteristics of its downstream tasks, which may be further subdivided into specific subtasks.

3.1.1 Taxonomy of RAG Applications

RAG models are applicable to various downstream tasks, each characterized by specific architectural requirements, evaluation strategies, and success criteria. These tasks can be classified into the following categories, as described by Gao et al. [Gao24]:

1. Question Answering

QA constitutes a central application domain of RAG systems. It involves generating answers to user questions based on a given query and supporting contextual information [Mav24]. The general QA task can be subdivided into several specialized subtasks, each addressing distinct question types and corresponding reasoning demands.

Single-Hop QA: Refers to QAs based on a single relevant passage, typically sufficient for fact-based or entity-specific queries [Mav24].

Multi-Hop QA: Characterized by reasoning across multiple sources or types of information such as texts, tables and knowledge graphs to answer more complex questions [Mav24, Min19].

Long-Form QA: Entails the generation of coherent, in-depth responses to open-ended questions, requiring the synthesis of information from extended textual contexts [Fan19].

Multiple-Choice QA (MCQA): Provides predefined answer options, with models evaluated based on their ability to identify the most contextually supported choice [Sha20].

Domain-Specific QA: Focuses on specialized application areas, where accurate integration of domain knowledge is required. This introduces complexity for LLMs in ensuring both accuracy and contextual relevance [Zha23].

Graph QA: Integrates LLMs with Graph Neural Networks (GNNs) to enable QA over structured knowledge graphs, supporting multi-relational reasoning via techniques such as soft prompting [He24].

Open-Domain QA: Involves retrieving and reasoning over large unstructured text corpora, and is particularly relevant for RAG systems. Four main architectural paradigms are commonly identified in the context of Open-Domain QA, particularly when implemented through RAG systems [Abd24].

- **Retriever-Reader:** This architecture integrates traditional information retrieval techniques with machine reading comprehension models. Initially, relevant documents or passages are retrieved based on the input query, these are subsequently processed by a reader model that identifies and extracts potential answer spans.

- **Generator-Retriever:** This architectural paradigm combines retrieval mechanisms with GenAI models. Retrieved passages, based on either sparse or dense vector representations, serve as input to a *seq2seq* GenAI model (e.g., transformer-based), which produces a natural language response based on the provided context.

- **Generator-Reader:** In this architecture, LLMs are employed to generate synthetic contextual documents based on the input question. The generated content serves as input to a reading comprehension model, which identifies potential answer spans. This approach separates the processes of document generation and answer extraction.

- **Retriever-Only:** This architecture approaches the QA task as a phrase retrieval problem, omitting document-level inference. Candidate answer phrases are retrieved directly, which may increase efficiency during inference, depending on the specific application context.

2. Dialogue

The dialog downstream task comprises multiple subtasks, each involving the generation of conversational responses that maintain coherence and contextual relevance.

Dialogue Generation: Refers to open-domain conversational tasks in which systems are assessed based on their capacity to produce contextually appropriate and informative responses. Evaluation is performed using both automatic metrics and human judgments [Din18].

Personal Dialogue: Denotes a dialogue modeling approach in which user-specific persona profiles are integrated to influence system responses. Outputs are conditioned on predefined traits or preferences, potentially contributing to consistency in long-term interactions [Lin19, Wan23b].

Task-Oriented Dialogue (TOD): Systems are designed to support the accomplishment of specific user goals by interpreting contextual information, planning dialog policies, and producing task-relevant responses. Recent methods make use of semi-supervised, pre-trained LMs based on large-scale corpora, which can be fine-tuned for various downstream **TOD** applications [He22].

Recommendation: Dialogue systems are designed to support users in identifying content or products aligned with their stated preferences. These systems incorporate user intent and interest in conjunction with large item collections to generate context-aware recommendations [He16].

3. Information Extraction (IE)

IE refers to the application of computational methods for identifying and extracting task-relevant information from human-readable documents. The extracted data is transformed into structured representations suitable for storage, processing, and retrieval by computer systems [Sin18]. Two common subtasks in this domain are described below:

Event Argument Extraction: Refers to the task of identifying entities (arguments) involved in specific events and classifying the roles they fulfill. This process is commonly divided into two stages: the identification of named entities (or the

use of annotated ground-truth entities) as candidate arguments, followed by the determination of their semantic roles within the event context [Zha20].

Relation Extraction: Involves detecting and categorizing predefined semantic relationships between identified entities in text. For example, in the sentence 'Jodie Foster **won** the Academy Award'. The verb *won* serves as the semantic relation connecting the entities 'Jodie Foster' and 'Academy Award' [Sin18].

4. Reasoning

Reasoning is a cognitive process involves the use of logic, evidence, and structured argumentation to derive conclusions or make decisions. It is considered a central topic in disciplines such as psychology, philosophy, and computer science [Hua22]. In the context of NLP, reasoning tasks can be divided into several subtasks:

Commonsense Reasoning: Involves the application of everyday world knowledge to disambiguate and interpret natural language, such as resolving pronoun references based on contextual plausibility [Dav15].

Chain-of-Thought (CoT) Reasoning: Characterized by the use of intermediate reasoning steps prior to producing a final answer. This approach is associated with increased interpretability and may improve prediction consistency [Tur23].

Complex Reasoning: Denotes the application of structured logic and rule-based reasoning to solve problems requiring multi-step inference. It is relevant for applications such as mathematical problem-solving, clinical decision support, or argumentative dialogue systems. Subtypes of complex reasoning include [Wan22]:

- **Analytical Reasoning:** solving problems by evaluating qualitative and quantitative constraints.
- **Logical Reasoning:** determining entailment relationships, for example in Natural Language Inference (NLI) tasks, where a model must assess whether a hypothesis logically follows from a given premise.
- **Multi-Hop Reasoning:** integrating information from multiple documents or textual segments to derive intermediate inferences and reach a final conclusion.
- **Numerical Reasoning:** performing arithmetic or symbolic operations on quantities mentioned in text.

5. Other Downstream Tasks

Beyond the major categories discussed, there exist several additional downstream tasks relevant in the context of RAG and LLM applications. These tasks are outlined briefly below. Readers are referred to the respective publications for further technical detail.

Language Understanding: Refers to tasks that require not only linguistic competence but also the application of basic reasoning and commonsense knowledge. While current models perform competitively, certain aspects of comprehensive language understanding remain challenging [Hen20].

Fact Checking/Verification: Tasks concerned with assessing the factual consistency of claims by comparing them with external sources of verified information [Kot20].

Text Generation: Refers to the task of generating coherent natural language text from structured data inputs [Leb16].

Text Summarization: Includes identifying salient information within a source document and producing a concise summary. Both extractive and abstractive techniques are commonly employed [Nar18].

Text Classification: Refers to the assignment of input texts to predefined categories (e.g., sentiment, topic), typically using LLMs through fine-tuning and in-context learning [Zha24b].

Sentiment Analysis: Addresses the classification of evaluative polarity into positive, neutral or negative categories based on lexical, contextual and syntactic features [Tab16].

Code Search: Involves retrieving relevant code snippets from large repositories based on natural language queries. This task often bridges the semantic gap between technical programming language and more abstract user queries, relying on both lexical and semantic retrieval strategies [Hus19].

Robustness Evaluation: Examines model performance in both answerable and unanswerable cases, often using binary classification to compare model outputs against human-labeled ground truth [Tha23].

Mathematical Reasoning: Evaluates models' ability to solve math problems, typically involving symbolic manipulation, arithmetic and logical reasoning [Cob21].

Machine Translation: Benefits from the availability of parallel corpora, supporting the alignment and modeling of sequences across languages [Koe05].

3.1.2 Focus on QA

Although, RAG systems can be applied to a broad range of domains, this thesis focuses on the QA task, which represents a representative and extensively studied application area. QA aligns closely with the RAG paradigm due to its reliance on retrieving external context and the need for accurate and informative responses in applied scenarios. Focusing on QA facilitates a structured evaluation of system characteristics such as retrieval quality, latency, and robustness against hallucination. Although other RAG use cases may require task-specific evaluation criteria, this study adopts a QA-oriented perspective to support domain-relevant and comparable assessment.

3.1.3 Deriving Evaluation Criteria

Based on the QA-oriented scope of this study, the following evaluation criteria are proposed to assess the effectiveness of RAG systems in practical applications:

- **Document Relevance:** Assesses whether the retrieved context is factually and semantically aligned with the user query, providing an adequate foundation for response generation.
- **Answer Precision:** Evaluates the degree to which the generated answer is correct, concise, and directly responsive to the input question.
- **Response Time/Latency:** Focuses on the time required by the system to generate an answer, based on established usability thresholds (e.g., Nielsen [Nie94]).
- **Hallucination Resistance:** Examines the extent to which generated content remains grounded in retrieved sources and avoids unsupported or fabricated information.

These criteria constitute the basis for evaluating RAG approaches with a focus on information preparation, as defined in **Category C1**. They enable a structured assessment of each approach's applicability, performance, and impact, contributing to the systematic analysis of the main research question (**RQ1**) and its sub-questions (**RQ2**–**RQ5**).

3.2 Enhancing Retrieval Performance

This chapter provides a structured overview of retrieval model types and strategies relevant to optimizing RAG systems. The presented categorization builds on the results

of the adapted SLR described in Section 1.3, which identified and classified recent retrieval approaches applicable to RAG-based QA systems. In addition to categorizing retrieval model types, the chapter analyzes retrieval granularity, discusses established retrieval methods and approaches, and introduces various optimization techniques that influence both retrieval effectiveness and generation quality.

3.2.1 Retrieval Model Types

Retrieval models can be broadly categorized according to the underlying methods used for encoding and comparing information. A systematic understanding of these model types provides a foundation for analyzing retrieval performance in various application contexts. The principal types of retrieval models are [Fan24]:

Sparse Retrieval is based on word-level representations of queries and documents. This approach is commonly applied in traditional text retrieval and in demonstration selection for in-context learning. Its main limitation lies in the absence of training, making performance sensitive to the quality of both the query and the underlying document collection. Furthermore, its reliance on term-based matching may reduce adaptability to alternative selection criteria, such as diversity, which are often relevant in LLM-based applications [Fan24].

Dense Retrieval encodes both queries and documents into a shared high-dimensional vector space, facilitating similarity comparison based on semantic features. In contrast to sparse retrieval, dense retrieval models are typically trainable, which allows adaptation to specific domains or tasks. A common architecture consists of a bi-encoder design in which separate encoders process queries and documents independently. Dense retrieval plays an important role in RAG systems, especially for dynamic context construction in in-context learning [Fan24].

Hybrid Retrieval integrates sparse and dense retrieval techniques to combine their respective strengths. While sparse retrievers emphasize lexical overlap, dense retrievers capture broader semantic relations. Implementation strategies vary: some systems apply both types jointly, others sequence them (e.g., dense retrieval followed by sparse filtering), and some extend the approach to multimodal contexts by incorporating visual embeddings. Such combinations aim to improve retrieval robustness and adaptability across heterogeneous tasks [Zha24c].

Graph Retrieval addresses the retrieval of structured information from graph-based knowledge representations in response to natural language queries. Relevant elements, such as entities, triples, paths or subgraphs, are extracted based on semantic alignment with the query. To constrain the search space and prioritize

relevant graph regions, these methods often apply semantic similarity measures between queries and graph components [Pen24].

3.2.2 Retrieval Granularity

Another important factor in improving retrieval performance is retrieval granularity. The granularity of the retrieval model determines the level at which data is retrieved, such as at the document, passage, token, or entity level. This granularity directly affects both the effectiveness of the retrieval process and the efficiency of database usage and search operations, as it influences the required storage space and computational effort [Fan24].

Chunk or Passage Retrieval is a technique commonly used in traditional Information Retrieval and is particularly effective for document retrieval tasks. Instead of retrieving entire documents, this approach focuses on presenting the most relevant sections within a document. Several types of passages can be distinguished [Liu02, Fan24]:

- **Structural passages:** These are based on the document's inherent structure, using author-defined boundaries such as paragraphs, sections, or individual sentences to demarcate passages.
- **Topical passages:** In this case, documents are divided according to subject matter or thematic coherence. Each passage corresponds to a coherent unit that reflects a specific subtopic or the main idea of the text.
- **Window passages:** These passages consist of a fixed number of words or bytes, often disregarding the logical structure of the document. The focus here is on the quantity of text rather than its organization.
- **Arbitrary passages:** Unlike the previous types, arbitrary passages can begin at any point within a document and are typically defined dynamically at query time. They can be further subdivided into:
  - **Fixed-length arbitrary passages**, which resemble overlapping windows but can start at any arbitrary point.
  - **Variable-length arbitrary passages**, which can vary in size depending on the relevance to the query.

Token Retrieval is a fine-grained retrieval method that operates at the word or subword level, rather than using larger chunks or passages. It enables faster search

processes but increases the storage burden on the database. Token Retrieval is particularly useful when rare patterns or out-of-domain data are required. Compared to chunk retrieval, token retrieval is less compact and may introduce more redundancy and irrelevance, which is why chunk retrieval remains more common, especially in RAG systems [Fan24].

Entity Retrieval focuses on knowledge rather than language and involves two main approaches: learning *entity representations* from text and external databases (e.g., Wikipedia) to build *entity memory*, or learning and retrieving mentions of entities at a more fine-grained level. This form of retrieval is particularly effective for *entity-centric tasks* and is more space-efficient than token-wise retrieval in RAG systems [Fan24].

3.2.3 Retrieval Methods and Approaches

Building upon the previously discussed retrieval models and levels of granularity, this section provides a structured overview of key retrieval techniques. These methods differ in their data representation strategies, computational requirements, and integration with downstream generation systems.

In this categorization, the core retrieval process responsible for retrieving documents based on the input query is referred to as **core-retrieval**, distinguishing it from **pre-retrieval** query preparation and **post-retrieval** context optimization. Additionally, methods extending the retrieval process itself, as well as those introduced via **new modules** and **new interaction patterns**, are included.

Pre-Retrieval

Pre-retrieval focuses on optimizing queries and indexing structures prior to the actual retrieval step. These techniques aim to enhance retrieval effectiveness by improving input formulation or search space organization before document selection begins (see Section 2.4.3).

Hypothetical Document Embeddings (HyDE) is a two-stage retrieval approach that begins by using a zero-shot, instruction-following LM to generate a hypothetical document based on a given query. Although, this synthetic document may contain hallucinated or incorrect information, it is designed to capture the underlying relevance signals of potential answers. In the second stage, a dense encoder trained with contrastive learning encodes the document (e.g. Contriever, see Section 3.2.3) into an embedding vector that is used to identify semantically

similar documents from a corpus via dense retrieval. This embedding acts as a bottleneck, helping to filter out irrelevant or inaccurate details from the initial generation. **HyDE** has demonstrated strong performance across multiple information retrieval tasks and languages, despite being fully unsupervised and not fine-tuned on task-specific data [Gao23].

Parent Child Embedding is a hierarchical retrieval technique commonly used in **Parent Document Retrieval** settings to improve retrieval from long or structured documents. Documents are divided into smaller segments (children), such as paragraphs, which are independently embedded. These child embeddings capture fine-grained semantics and are linked to their corresponding parent document. During retrieval, relevance is assessed at the child level but results are returned at the parent level. This approach enables more accurate retrieval by combining local semantic signals with document-level context, preserving both internal structure and semantic variation [Mis24].

Multi-Aspect Retrieval addresses the challenge of combining multiple, distinct aspects into a single query for more accurate results. To retrieve diverse documents, their embeddings may be far apart in the embedding space. A key solution is to use activations from the multi-head attention layers of a transformer decoder as embeddings. Transformer architecture consists of blocks with attention and feed-forward modules, where multi-headed attention captures various relationships by learning different weight matrices. These multi-aspect embeddings are then directly used for both query and document representations, improving retrieval performance [Bes24].

Multi-Query Rewriting (MQR) addresses the challenge of noisy or underspecified user queries that can lead to intent deviation and suboptimal retrieval. By generating multiple diverse reformulations of the original query, **MQR** enhances both document retrieval and final response generation. The process typically involves three stages: (1) a query rewriter generates alternative phrasings of the input query, (2) a retriever searches for relevant documents for each rewritten query, and (3) a re-ranking mechanism consolidates and ranks the retrieved results to select the top-K most relevant documents, which are then passed to the LLM for generation. This approach improves retrieval coverage and robustness against poorly formulated inputs [Li24].

Core-Retrieval

Core-retrieval constitutes the central retrieval mechanism within the RAG pipeline. It is responsible for identifying and selecting relevant documents from the external knowledge base based on the processed input query.

Maximum Marginal Relevance (MMR) operates by computing a linear combination of the query-document relevance and the maximum distance of a candidate document to the set of already selected documents. Specifically, it measures relevance and novelty independently, and combines these scores into a single metric, referred to as *marginal relevance*. A document is assigned high marginal relevance if it is both highly relevant to the query and sufficiently dissimilar to previously selected documents. This method helps to mitigate redundancy in the retrieved contexts and improves the overall quality and coverage of the information [Gol98, Xia15].

Similarity Search traditionally relies on embedding models to map sentences into a vector space, enabling the retrieval of similar sentences based on distance metrics such as Jaccard similarity, cosine similarity, or Euclidean norm. In standard approaches, similarity is measured directly by computing vector distances. However, in the context of RAG, alternative strategies have emerged, such as constructing conversational chains that evaluate sentence-pair similarity more dynamically [Ber24, Che24a].

Multi-Hop Query is essential for improving the accuracy and reliability of systems that rely on multiple sources to answer complex, multi-faceted queries [Tan24b]. It refers to a retrieval process that involves querying and reasoning over multiple pieces of supporting evidence to provide a comprehensive answer. A multi-hop query requires retrieving several chunks of information from a retrieval set, which together address the question. There are four main types of multi-hop queries:

- **Inference query:** The answer to the query is deduced through reasoning based on the retrieved evidence.
- **Comparison query:** The answer requires comparing different pieces of evidence within the retrieval set.
- **Temporal query:** The answer necessitates analyzing the temporal aspects of the retrieved information.
- **Null query:** This query type assesses the generation quality. The answer cannot be derived from the retrieved set, and the system should produce a

null response instead of generating incorrect information or *hallucinating* an answer.

Term Frequency Inverse Document Frequency (TF-IDF) is a widely used statistical measure in traditional information retrieval. It evaluates the importance of a term within a specific document relative to a corpus of documents. The method combines two components:

- **Term Frequency (TF):** Quantifies how often a term appears in a document. The assumption is that terms occurring more frequently in a document are more important within that context.
- **Inverse Document Frequency (IDF):** Reflects how rare or informative a term is across the entire corpus. It is typically calculated as $\log(N/n)$, where N is the total number of documents and n is the number of documents that contain the term. Terms that appear in many documents receive lower **IDF** scores.

The **TF-IDF** value is the product of these two measures. It assigns higher weights to terms that are frequent in a specific document but rare across the corpus. This helps to suppress common words (e.g., 'the', 'and' and emphasize more distinctive, context-specific terms. **TF-IDF** remains a foundational technique in text mining, keyword extraction, and classical document retrieval systems [SJ72, Ram03].

BM25 (Best Match 25) is a ranking function commonly used in traditional information retrieval systems. It is grounded in the probabilistic retrieval framework and often referred to as Okapi BM25. The model estimates the relevance of a document to a search query by combining **TF**, **IDF**, and document length normalization. Rather than being a single fixed formula, **BM25** represents a family of scoring functions with tunable parameters. Due to its effectiveness, interpretability, and simplicity, **BM25** remains a strong baseline in modern retrieval research, including neural and hybrid approaches [Rob95, Rob09, Ama09, Lin22].

Dense Passage Retrieval (DPR) is a retrieval method designed to efficiently retrieve the top-K passages most relevant to an input query by indexing all passages in a low-dimensional continuous space. It uses a dense encoder to map each text passage into a d-dimensional real-valued vector and builds an index over all M passages to enable efficient retrieval. During inference, a separate query encoder maps the input question into the same vector space, allowing retrieval of K passages whose vectors are closest to the query vector. **DPR** adopts a bi-encoder architecture, where the document and query encoders independently produce dense representations. The task of finding the top-K most relevant

documents corresponds to a Maximum Inner Product Search (MIPS) problem, which can be approximately solved in sub-linear time to ensure efficiency at scale [Kar20, Lew20].

Dense Hierarchical Retrieval (DHR) is a hierarchical dense retrieval framework proposed to improve passage selection in open-domain QA [Liu21]. It extends standard dense retrieval by introducing a two-level retrieval process: a **document-level retriever** first selects relevant documents, followed by a **passage-level retriever** that identifies pertinent passages within those documents. To further refine the final ranking, passage-level scores are calibrated using the relevance of their parent documents. This hierarchical design enables more efficient and accurate retrieval by narrowing the search space and maintaining contextual consistency across levels.

Hybrid Hierarchical Retrieval (HHR) is a retrieval framework that integrates both sparse and dense retrievers at two hierarchical levels: **document-level** and **passage-level** [Ari23]. Instead of selecting a single retrieval paradigm per stage, **HHR** explores hybrid strategies, including a simple yet effective heuristic that interleaves the top-$K/2$ results from both retrievers to form a top-K candidate set. Additionally, it performs passage reranking using pre-computed dense representations to reduce latency. This setup enables a systematic evaluation of the tradeoffs between accuracy, storage requirements, and retrieval latency, providing more realistic insights into deployment in large-scale systems.

Graph-Based Retriever enhances embedding-based retrieval by incorporating structural information from a knowledge graph to address information imbalance in large corpora, such as in medical domains [Del24]. Given a query, entities are extracted and connected via the shortest path in the graph. Text chunks linked to these entities and their neighbors are retrieved and ranked using a combined scoring metric that integrates graph distance, recency, and contextual impact. This rebalancing mechanism improves retrieval fairness and relevance beyond standard embedding similarity methods.

Post-Retrieval

Post-retrieval complements core retrieval by refining the retrieved results before passing them to the generator. Typical techniques include reranking, filtering, and context compression to optimize the final context window provided to the LM (see Section 2.4.3).

Reciprocal Rank Fusion (RRF) is an algorithm used in information retrieval systems to combine rankings from multiple sources. The goal is to improve

the accuracy and relevance of search results by assigning a unified score to each document based on its rank in different retrieval lists. The score for each document is calculated by using the reciprocal rank from each list and then combining them. The **RRF** score is computed as follows:

$$\text{Reciprocal Rank Fusion Score} = \frac{1}{\text{rank} + k}$$

where *rank* is the document's position in a retrieval list, and k is a constant that adjusts the influence of the original ranking.

After calculating these scores, they are combined, and the documents are reranked based on the final score. The results are then sent to a LM to generate the final output. This process enhances retrieval systems by leveraging multiple rankings, improving both precision and recall [Rac24].

Relevance Estimator (RE) measures the relevance between a question and context. It takes as input the same question-context pair as the generator but generates a classification token (*true* or *false*) to indicate the context's relevance to the question. The *true* token independently signals whether the context is relevant. By comparing multiple contexts, the probability of the *true* token is converted into a logit, which serves as the relevance score for ranking the retrieved context [Kim24].

Long Context Re-Order addresses the limitations of LLMs in processing long input sequences, where relevant information in the middle of a context window is often neglected due to primacy and recency biases [Liu23a]. Studies show that model performance significantly degrades when key information is placed in non-salient positions or when many documents (e.g., 10+) are included. To mitigate this, RAG frameworks such as **LangChain** and **LlamaIndex** implement reordering strategies that prioritize and reposition the most relevant retrieved documents, ensuring critical content is presented in more impactful positions within the context window [Lan24, Lla24].

Contextual Compression is a retrieval-enhancement technique designed to reduce the size of the retrieved context while preserving or improving its relevance. It typically follows a two-stage process: a *retriever* first selects a set of candidate documents, followed by a *compressor* that filters or condenses these documents, often at the passage level, to retain only the most salient information. This approach enables LMs to work with more focused and informative inputs, reducing unnecessary tokens and improving generation quality [Ver24, Hwa24].

Joint Passage Retrieval (JPR) is a retrieval model designed to address the challenge of multi-answer retrieval, where multiple passages must be selected to collectively cover diverse answer aspects. **JPR** employs a two-stage pipeline: a first-stage retriever (e.g., **DPR** combined with **REALM**) retrieves candidate passages, followed by an encoder-decoder reranker that autoregressively generates a sequence of passage references. This autoregressive reranking models the joint probability of the selected passages, allowing each step to condition on previously selected content. To guide the model in the absence of a ground-truth passage order, a dynamic supervision strategy is used, encouraging the selection of passages that introduce novel answer information. Furthermore, a tree decoding algorithm enables flexible control over passage diversity during generation. This approach improves both relevance and answer coverage in multi-passage retrieval scenarios [Sar24, Min21].

Self-Reflective Retrieval involves training a LM in an end-to-end manner, where the model learns to reflect on its own generation process. It generates both the task output and special reflection tokens. These tokens are divided into retrieval tokens, which signal when additional retrieval is needed, and critique tokens, which assess the quality of the output. The model evaluates the relevance of multiple retrieved passages and generates the corresponding task output. Using critique tokens, it then self-assesses and selects the output with the highest factual accuracy and overall quality [Asa23].

Chain-of-Note (CoN) is a framework designed to enhance the robustness of *retrieval-augmented* language models (**RALMs**) in the presence of noisy, irrelevant, or out-of-domain documents. The core idea is to generate sequential reading notes for each retrieved document, allowing the model to explicitly evaluate its relevance and extract reliable information before generating an answer. This process enables more systematic filtering of irrelevant content and improves the factual accuracy and contextual alignment of the final response [Yu23].

New Modules

New modules extend the RAG pipeline by introducing additional components that enhance retrieval or generation capabilities. These modules enable more advanced retrieval logic, adaptive memory integration, or more sophisticated interaction between retrieval and generation stages (see Section 2.4.3).

Retrieval Transformer (RETRO) is a retrieval-enhanced autoregressive LM that integrates external context via chunk-wise nearest-neighbor retrieval and cross-chunked attention [Bor22]. It retrieves contiguous token chunks (rather than individual tokens), significantly reducing storage and computational requirements.

A specialized chunked cross-attention mechanism incorporates the retrieved information with time complexity linear in the retrieval size. **RETRO** achieves consistent improvements in text generation quality, factual accuracy, and performance on downstream tasks such as open-domain QA. Gains scale across model sizes (150M–7B) and can be further improved at evaluation time by increasing database size and retrieval depth [Bor22, Wan23a].

Memory Retrieval (MemoRAG) is a RAG framework enhanced by global memory-augmented retrieval. It introduces a dual-system architecture in which a lightweight, long-range component first constructs a global memory representation of the extended input context. This memory is then leveraged by a second, more expressive and computationally intensive model that generates the final output. **MemoRAG** demonstrates strong performance across a variety of long-context evaluation tasks. It outperforms traditional RAG methods both in complex scenarios, where these methods often encounter difficulties, and in simpler settings that represent typical application domains [Qia25]. Building on the general concept of memory-augmented retrieval introduced by **MemoRAG**, recent approaches such as **SynapticRAG** [Hou24] and Chen et al. [Che24a] propose refinements in how memory is constructed, dynamically updated, and selectively queried during generation.

Atlas is a RAG model that fine-tunes an encoder-decoder LM jointly with a dense retriever. It builds on *Contriever*, a dual-encoder architecture trained in an unsupervised fashion to encode queries and documents independently [Iza21]. During training, Atlas applies techniques such as attention distillation, leave-one-out perplexity, and end-to-end multi-document training (EMDR^2) to align retrieval with generation objectives. The model supports various fine-tuning strategies including re-ranking, query-side adaptation, and full index updates, allowing for efficient and scalable retrieval without labeled data [Iza23].

REALM is a retrieval-based framework designed for open-domain QA, combining masked language modeling with latent knowledge retrieval. It follows a retrieve-then-predict paradigm, consisting of two main components: a *neural knowledge retriever* and a *knowledge-augmented encoder*. The retriever uses a dense inner product model, where two separate embedding functions map input queries and documents to d-dimensional vectors. Relevance scores are computed via inner product, and retrieval is implemented as a softmax over these scores. The encoder then concatenates the query with the retrieved documents and processes them using a separate Transformer, enabling cross-attention before predicting the output. The retriever is trained in a latent fashion; its parameters are updated indirectly via gradients from the encoder's prediction loss. The pretraining process incorporates

techniques such as salient span masking, the use of null documents, the prevention of trivial retrievals, and careful initialization to enhance retrieval quality [Guu20].

Adaptive Retrieval (Adaptive-RAG) is a RAG framework that dynamically determines whether and how to retrieve external information based on the complexity of the input query [Jeo24]. Central to this approach is a query classifier that assesses query difficulty and selects one of three strategies:

- **Non-Retrieval QA:** Direct LLM response for simple queries.
- **Single-Step Retrieval:** One retrieval round for moderately complex queries.
- **Multi-Step Retrieval:** Iterative retrieval for complex, multi-hop questions.

This adaptive selection is trained using supervised or reinforcement learning to balance response accuracy and retrieval cost. By integrating retrieval decisions into the model pipeline, **Adaptive-RAG** improves both efficiency and performance compared to static RAG architectures.

GraphRAG sensemaking is a graph-based RAG approach designed for sensemaking queries that require a global understanding of a document corpus. It constructs an entity knowledge graph from source documents, partitions it into hierarchical communities, and generates bottom-up summaries for each community using a LM. At query time, **GraphRAG** applies a map-reduce strategy: community summaries are used to generate partial answers (map), which are then aggregated into a final response (reduce). This method outperforms standard vector-based RAG approaches in comprehensiveness and diversity, especially for complex analytical questions [Edg24].

In-Context Adaptive Retrieval (OpenRAG) is a RAG framework that optimizes the retriever in an end-to-end manner to capture in-context relevance and adapt to evolving task requirements [Zho25]. It introduces a two-phase architecture consisting of an offline retriever optimization and an online inference stage. In the offline phase, **OpenRAG** uses a *RAG label* (a binary indicator) and a *RAG score* (a joint generation likelihood) to train the retriever based on how well documents contribute to the LMs output. In the online phase, the retriever adapts dynamically to the input query and context, enabling in-context adaptive retrieval. This design leads to consistent performance improvements, outperforming prior retrievers by up to 4.0% in knowledge-intensive generation tasks.

New Patterns

New Patterns (see Section 2.4.3) further generalize the structure of RAG pipelines. They introduce flexible architectural designs and interaction flows, allowing dynamic control over retrieval strategies and increasing modularity to address diverse task requirements.

Generator-Retriever-Generator (GRG) Approach is an architecture for open-domain QA that enhances the integration of generation and retrieval processes. Unlike traditional RAG setups, where retrieval precedes generation, **GRG** first uses a LM to generate synthetic context based on the input question. This generated context is then used to guide a dense retrieval system in selecting the most relevant external documents. A second generator subsequently produces the final answer based on both the retrieved documents and the original question. This two-stage generative framework improves contextual alignment and the informativeness of answers by combining the strengths of generative reasoning and document retrieval [Abd24].

Retrieval-Augmented Passage Tree Organization for Reasoning (RAPTOR) is a hierarchical retrieval pattern designed to efficiently navigate long or structured documents. It organizes passages into a recursive tree based on semantic similarity, enabling both broad and fine-grained retrieval. Nodes are embedded and clustered using dense retrieval techniques and traditional clustering algorithms. During retrieval, the tree is traversed recursively to locate the most relevant subpassages while preserving contextual structure [Sar24].

Fusion-in-Decoder (FiD) with Knowledge Distillation trains a retriever without strong supervision by using a reader model to guide it. The retriever first selects relevant documents, which are then processed by the reader to solve the task. The reader's attention over the input documents is treated as a supervision signal, allowing the retriever to learn document relevance via distillation from the reader's behavior (i.e., a teacher-student setup). This method avoids the need for manually labeled data and can be combined with retrieval baselines like **BM25** or **DPR** [Iza22].

Table 3.1 provides a structured overview of the retrieval methods and approaches discussed within this chapter. The methods are organized according to their position within the retrieval pipeline (*pre-, core-, post-retrieval,* as well as *new modules* or *patterns*) and are further categorized by their retrieval model type (dense, sparse, hybrid, or graph) and the level of retrieval granularity (chunk, token, or entity). This structure emphasizes that, depending on the specific stage and goals of the retrieval process, different methods can be selectively applied to enhance retrieval performance.

Table 3.1: Categorization of retrieval approaches based on their position within the retrieval pipeline, retrieval model type (dense, sparse, hybrid, graph), and retrieval granularity level (chunk, token, entity).

Methods	Retrieval Model Type				Retrieval Granularity			Sources
	Dense	Sparse	Hybrid	Graph	Chunk	Token	Entity	
Pre-Retrieval								
Parent-Child Embedding	x				x			[Mis24]
Multi-Aspect Retrieval	x				x			[Bes24]
HyDE	x				x			[Gao23]
MQR	x	x			x			[Li24, Kos24]
Core-Retrieval								
Similarity Search	x	x	x	x	x	x	x	[Wan23a, Saw24, Lua21, Wen17, Ger22, Pen24]
Multi-Hop Query	x		x		x			[Tan24b, Sid21]
TF-IDF		x			x			[SJ72]
BM25		x			x			[Rob95]
MMR	x	x	x		x			[Gol98, Tra24, Pru24]
DPR	x				x			[Kar20]
DHR	x				x			[Liu21]
HHR			x		x			[Ari23]
Graph-Based Retriever	x			x	x			[Del24]
Post-Retrieval								
Long Context Re-Order	x				x			[Liu23a]
Self-Reflective Retrieval	x	x			x	x		[Asa23, Tan24a]
Contextual Compression	x				x			[Hwa24]
CoN	x					x		[Yu23]
JPR	x				x			[Min21]
RRF	x	x	x		x			[Bru23, Qu21, Yu21]
RE	x	x			x			[Kim24, Nog19]
New Modules								
RETRO	x				x			[Bor22, Wan23a]
Memory Retrieval			x			x		[Qia25]
Atlas	x				x			[Iza23, Iza21]
REALM	x					x		[Guu20]
Adaptive Retrieval	x	x			x			[Jeo24, Xie24]
GraphRAG sensemaking				x			x	[Edg24]
In-Context Adaptive Retrieval			x		x			[Zho25]
New Patterns								
GRG	x				x			[Abd24]
RAPTOR			x		x			[Sar24]
FiD	x				x			[Iza22]

3.3 Evaluation Framework

Following the systematic analysis of retrieval strategies and the previously derived evaluation criteria, the following section introduces the evaluation framework. Based on an adapted SACAM methodology, it defines evaluation targets, quality dimensions, required abilities, and corresponding metrics for the systematic assessment of the implemented RAG approaches.

3.3.1 Evaluation Target

In the evaluation of RAG systems, two main targets can be identified that significantly influence the overall response quality:

ET1: Retrieval Quality refers to the quality of the retrieved context. It plays a critical role in determining whether relevant and useful information has been extracted from the context source. A reliable retrieval component serves as the foundation for effective response generation. The assessment of this component is based on specific criteria, which will be discussed in subsequent sections.

ET2: Generation Quality focuses on the model's ability to generate coherent, meaningful, and contextually relevant answers based on the provided context. This can be further distinguished into two content types:

- **Unlabeled Content:** Refers to instances where no reference answers (ground truth) are available. Evaluation in this case is based on qualitative characteristics of the response.
- **Labeled Content:** Involves scenarios where ground truth answers are available, allowing for a more objective evaluation through comparison.

Depending on the focus of the evaluation, one may choose to concentrate on either of these two targets: **Retrieval Quality** or **Generation Quality**. Alternatively, both targets can be evaluated simultaneously to provide a comprehensive assessment of the RAG system's performance. The specific evaluation aspects and metrics for each target will be defined in the following sections.

3.3.2 Quality Aspects and Required Abilities

This section introduces a set of key quality aspects to support a structured evaluation of retrieval and generation quality in RAG systems. Rather than presenting an exhaustive list, it offers a focused selection that captures the most critical factors for assessing performance and robustness. These aspects are grouped into two main categories:

Primary Quality Scores: These scores reflect the effectiveness of the information retrieval and generation processes from different perspectives, allowing for a comprehensive evaluation. The three primary quality dimensions are:

- **QS1 Context Relevance:** Assesses the relevance, accuracy, and specificity of the retrieved content in relation to the user query, ensuring that the provided context is useful for answering the question.
- **QS2 Answer Faithfulness:** Measures whether the generated response is grounded in the retrieved context and avoids hallucination or contradiction, thereby ensuring internal consistency.
- **QS3 Answer Relevance:** Evaluates the extent to which the generated answer directly addresses the user query and provides meaningful and appropriate information.

Essential Required Abilities: These abilities reflect the model's performance under complex or challenging conditions and highlight its adaptability and robustness in practical applications. Four essential abilities are considered:

- **RA1 Noise Robustness:** Assesses the model's ability to handle irrelevant or noisy documents. These may appear related to the query but do not provide useful or content-specific information.
- **RA2 Negative Rejection:** Evaluates whether the model can appropriately refrain from answering a question when the retrieved documents lack the necessary knowledge to generate a reliable response.
- **RA3 Information Integration:** Measures the model's ability to synthesize information from multiple documents to formulate a coherent answer to a complex query.
- **RA4 Counterfactual Robustness:** Evaluates the model's ability to identify and disregard known inaccuracies or false information within documents, even when it is explicitly exposed to potentially misleading content.

These aspects are assigned to the evaluation targets as follows: *Context Relevance* and *Noise Robustness* are linked to **Retrieval Quality**, while *Answer Faithfulness*, *Answer Relevance*, *Negative Rejection*, *Information Integration*, and *Counterfactual Robustness* correspond to **Generation Quality**. Depending on the evaluation focus, relevant aspects and abilities will be prioritized accordingly. Both targets can be evaluated together for a comprehensive assessment.

3.3.3 Types of Evaluation Metrics

Before discussing the specific evaluation metrics, it is important to distinguish between two primary categories, which determine the nature of the evaluations conducted:

Reference-based metrics assess the quality of a text by measuring the alignment between system outputs and human-written texts, referred to as references. These references serve as the gold standard for evaluation, as they represent ideal outputs. However, the use of reference-based metrics requires the preparation of reference texts, which must be carefully created and verified by humans [Ito25].

Reference-free metrics offer significant advantages in terms of scalability for Natural Language Generation (NLG) tasks, as they eliminate the need for reference texts. Recent advancements have also improved the correlation between reference-free evaluation scores and human ratings, making it an active and promising area

of research. Reference-free metrics evaluate output quality without relying on references and can be further divided into two categories [Ito25]:

- **Absolute evaluation:** Where a context and hypothesis are given, and a quality score is assigned to the output.
- **Ranking evaluation:** Where a context and a set of hypotheses are provided, and the hypotheses are ranked by quality. In particular, pairwise evaluation is a common method for comparing two hypotheses directly.

3.3.4 Evaluation Metrics

After discussing the evaluation targets, quality aspects, and required abilities of the RAG system, appropriate evaluation metrics can be derived.

The **RAG Triad**, originally introduced by TruLens, provides a structured evaluation schema for RAG systems [Tru25b]. It has since been widely adopted as the conceptual foundation for more comprehensive evaluation frameworks. The architecture comprises three key components: **Input/Query**, **Retrieved Context** (feeding into **Retrieval Quality**) and **Response** (reflecting **Generation Quality**) (see Figure 3.1) [Tru25b, luJI25].

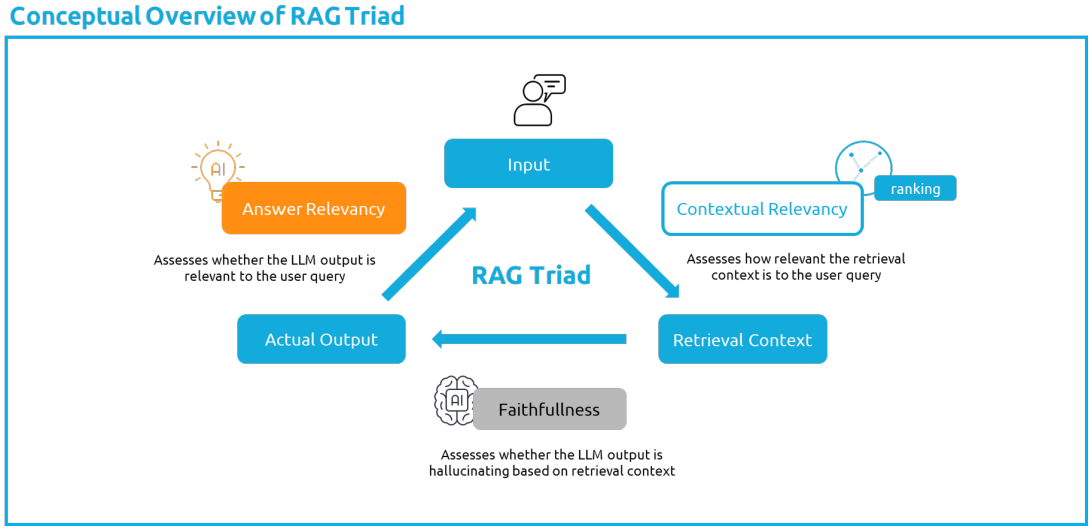


Figure 3.1: Own illustration of the RAG Triad, adapted from [luJI25].

Within this **RAG Triad**, three core quality aspects are essential for evaluating system robustness and hallucination resistance: **Context Relevance**, **Answer Faithfulness** (also referred to as *Groundedness*), and **Answer Relevance**. Various metrics are applied to quantify these aspects:

1. **Precision**
   Measures the quality (i.e. relevance) of the retrieved documents or information, specifically how many of the retrieved documents are actually relevant to the query [Sin01, Saj18, Gou05]. It is defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{Number of relevant retrieved documents}}{\text{Total number of retrieved documents}}$$

   where:

   - TP: True Positives (relevant documents correctly retrieved)
   - FP: False Positives (irrelevant documents retrieved)

2. **Recall**
   Recall measures the coverage of the relevant documents or information from the knowledge base, indicating how much of the relevant information the system successfully retrieved [Sin01, Saj18, Gou05]. It is defined as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{Number of relevant retrieved documents}}{\text{Total number of relevant documents}}$$

   where:

   - TP: True Positives (relevant documents correctly retrieved)
   - FN: False Negatives (relevant documents not retrieved)

3. **F1 Score**
   The **F1** score is defined as the harmonic mean of precision and recall, providing a balanced evaluation of both metrics. It is derived from the following formula [Chi20]:

$$F1 = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FP} + \text{FN}} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

4. **Mean Reciprocal Rank (MRR)**
   A a widely used evaluation metric in information retrieval and ranking tasks. It provides insight into how effectively a model ranks the correct output, such as a relevant sentence or document, among a set of potential candidates.

MRR is based on the concept of Reciprocal Rank (RR), which is the inverse of the rank position at which the first relevant result appears. For instance, if the correct item is ranked first, RR is 1.0. If it appears at position two, RR is 0.5. It decreases reciprocally with rank. This allows for a fine-grained understanding of how well a model prioritizes the most relevant results.

The Mean Reciprocal Rank is then computed by averaging the reciprocal ranks over a set of N queries or instances:

$$\text{MRR} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\text{rank}_i}$$

where:

- N: the total number of evaluation instances or queries
- rank_i: the rank position of the first correct result for the i-th instance

A higher **MRR** indicates that the model consistently retrieves relevant items at higher ranks, which reflects better performance in ranking and retrieval tasks [Cra09, Reh24].

5. **Mean Average Precision (MAP)**
   This evaluation metric commonly used in information retrieval to measure the performance of a system across a set of query topics. **MAP** is calculated as the arithmetic mean of the Average Precision (AP) values for each query in the evaluation set. It can be expressed as follows:

$$\text{MAP} = \frac{1}{n} \sum_{i=1}^{n} \text{AP}_i$$

where:

- n: the total number of query topics in the evaluation set
- AP_i: the Average Precision value for the i-th query topic

The Average Precision (AP) for a given query topic is computed based on the precision at each relevant document retrieved, averaged over the number of relevant

documents. A higher **MAP** value indicates better overall retrieval performance across all queries [SMB09].

6. **Normalized Discounted Cumulative Gain (NDCG)**
   A metric for assessing ranking quality in text retrieval systems. It evaluates the effectiveness of a system based on the graded relevance of retrieved passages, with scores ranging from 0 to 1, where 1 indicates an ideal ranking.

   Unlike traditional binary relevance measures, **NDCG** allows for degrees of relevance, making it suitable for evaluating systems where documents are not simply relevant or irrelevant. A key characteristic is its use of a discount function, which reduces the impact of lower-ranked documents. This is particularly important in search scenarios, as users typically pay more attention to top-ranked results. The **NDCG** at rank K is defined as [JÖ2, Wan13, Val09, Zha24a]:

   $$\text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}}$$

   where:

   - **DCG@K (Discounted Cumulative Gain):** The weighted sum of graded relevance values, where the weight decreases logarithmically with the position of the result.
   - **IDCG@K (Ideal DCG):** The maximum possible **DCG@K**, achieved when the ranking is perfectly ordered by relevance.

7. **Hit Rate**
   The **Hit Rate** measures how often it finds relevant information within the top-K retrieved documents. It calculates the fraction of queries for which the correct answer is present in the top-K results. The hit rate is defined as a binary score: $HR = 1$ if the desired relevant document is present in the retrieved results, and $HR = 0$ otherwise. Typically, the mean hit rate across all queries is reported. The formula is given as [Jos24]:

   $$\text{Hit Rate} = \frac{\left|U_{h_i t}^{K}\right|}{|U_{\text{all}}|} = \frac{\text{Number of hits}}{\text{Total number of queries}}$$

   where:

- $\left|\boldsymbol{U}_{\boldsymbol{hit}}^{\boldsymbol{K}}\right|$ is number of queries for which the correct chunk containing the answer is included in the top-K retrieved recommendation list
- $|\boldsymbol{U}_{\mathbf{all}}|$ is total number of queries in the test dataset

8. **Bilingual Evaluation Understudy (BLEU)**
   BLEU is an automatic, language-independent metric for evaluating machine translation quality [Pap02]. It approximates human judgment by computing modified N-gram precision between a candidate translation and one or more references. The score ranges from 0, denoting complete mismatch, to 1, indicating a perfect overlap between the candidate output and the reference. A brevity penalty is applied to penalize excessively short hypotheses. The **BLEU** score is formally defined as:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

   with:

   - p_n: Modified precision for N-grams of size n.
   - w_n: Weight for each N-gram, typically $w_n = \frac{1}{N}$.
   - N: Maximum N-gram length (commonly $N = 4$).

   To penalize overly short translations, BLEU incorporates a brevity penalty (BP), defined as:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-\frac{r}{c})} & \text{if } c \leq r \end{cases}$$

   where c is the length of the candidate translation and r is the effective reference length.

9. **Recall-Oriented Understudy for Gisting Evaluation (ROUGE)**
   ROUGE is an automatic metric used to evaluate the quality of summaries by comparing them to reference summaries created by humans. It measures the overlap of units such as N-grams, word sequences, and word pairs between the

candidate and reference summaries. **ROUGE** is commonly used in tasks like automatic summarization and QA. Its variants include:

- **ROUGE-N** (N-gram overlap)
- **ROUGE-L** (Longest Common Subsequence)
- **ROUGE-W** (Weighted LCS)
- **ROUGE-S**(Skip-bigram overlap)

Due to their complexity and limited applicability in the context of RAG, detailed definitions of the variants can be found in the original paper [Lin04].

10. **Metric for Evaluation of Translation with Explicit Ordering (METEOR)**

    METEOR is an automatic evaluation metric proposed as an alternative to **BLEU**, aiming to mitigate certain limitations associated with BLEU, such as its limited sensitivity to recall. In contrast to **BLEU, METEOR** explicitly incorporates recall, which allows for a more balanced assessment of how much of the reference meaning is captured. This feature is particularly relevant in evaluating the completeness of machine-generated translations. **METEOR** calculates a harmonic mean of precision and recall, typically assigning greater weight to recall and introduces a penalty to account for fragmented or misordered alignments. The final score is computed as follows:

$$\text{METEOR} = F_{\text{mean}} \cdot (1 - \text{Penalty})$$

    where F_{mean} denotes the harmonic mean of precision and recall, and the *Penalty* component penalizes disordered alignments based on the number and length of matching chunks. Detailed definitions of both components can be found in the original paper by Banerjee and Lavie [Ban05].

11. **Accuracy**
    A fundamental evaluation metric that reflects the proportion of correctly predicted instances, both positive and negative, relative to the total number of instances in a dataset. It provides an overall measure of classification performance, where a value of 1 indicates perfect prediction, and 0 indicates complete failure. The formula for accuracy is given by [Chi20]:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

where:

- TP: True Positives (instances correctly classified as positive)
- TN: True Negatives (instances correctly classified as negative)
- FP: False Positives (instances incorrectly classified as positive)
- FN: False Negatives (instances incorrectly classified as negative)

12. **Exact Match (EM)**
    EM metric represents the proportion of generated responses that exactly align with the reference answers. It quantifies the percentage of instances where the output precisely matches the corresponding target. This metric measures the proportion of predictions that exactly match any of the reference answers [Sch22, Raj16].

To provide a better overview of which metrics cover which evaluation aspects and required abilities, Table 3.2) summarizes the different metrics and highlights the **Context Relevancy, Faithfulness,** and **Answer Relevancy** aspects they evaluate, as organized by the **RAG Triad**. This facilitates the selection of the appropriate metric for specific evaluations.

Table 3.2: Mapping of Quality Scores and Required Abilities to Evaluation Metrics, organized by the RAG Triad, highlighting the Context Relevancy, Faithfulness, and Answer Relevancy.

Metrics	Quality Scores			Required Abilities				Metric Type	
	Context Relevance	Answer Faithfulness	Answer Relevance	Noise Robustness	Negative Rejection	Information Integration	Counterfactual Robustness	Ref-based	Ref-free
Context Relevancy									
Precision	x				x				x
Recall	x								x
F1 Score	x				x				x
MRR	x								x
MAP	x				x				x
NDCG	x								x
Hit Rate	x								x
Faithfulness									
BLEU			x					x	
ROUGE			x					x	
METEOR			x					x	
Answer Relevancy									
Accuracy	x	x	x	x	x	x	x		x
EM						x			x

3.3.5 Types of Evaluation for RAG

The evaluation of RAG systems can be based on two main approaches, each providing a distinct methodology for assessing the performance and capabilities of these models. These approaches can be assessed using the evaluation metrics discussed in the previous Section 3.3.3, which include both **reference-based** and **reference-free** metrics.

Benchmarks use large, standardized datasets that are often manually annotated (**reference-based**) to assess the fundamental capabilities of RAG models. Benchmarks typically focus on either the retrieval or generation component, using human-annotated data as a ground truth to measure the accuracy of retrieved documents or the correctness and coherence of generated responses [Gao24, Yu24].

Automated End-to-End Evaluation evaluates the entire RAG pipeline, using LLMs to assess the quality of generated responses. Automated evaluations can be **reference-based**, where a ground truth dataset is generated by an LLM based on a knowledge base. The quality of the generated response is then evaluated through **reference-free metrics**, which compare the generated response to the LLM-generated ground truth dataset[Siv24]. In the **reference-free** case, the evaluation is performed through automated metrics or LMs that judge quality based on coherence, relevance, and other characteristics [Es24]. This approach covers both data retrieval and response generation, ensuring the integrity of the full pipeline [Gao24, Siv24].

3.3.6 Assessment of Robustness and Mitigation Effectiveness via Mapping of Failure Points to Required Abilities

While automated metrics provide operationalized scores for *Primary Quality Aspects*, they are limited in capturing how well a system performs under adverse or realistic conditions. To address this, the evaluation framework includes a targeted mapping between empirically observed **FPs** (see Section 1.1) and the essential *Required Abilities* defined in Section 3.3.2.

This mapping allows for a qualitative assessment of robustness and helps identify whether the implemented retrieval strategies effectively mitigate known system weaknesses. In doing so, it bridges the gap between theoretical capability requirements and practical system behavior.

Table 3.3 presents the mapping for failure points from **Category C1 (Information Preparation)**, highlighting which abilities are challenged and to what extent the system was designed to address them.

Table 3.3: Mapping of FPs to Required Abilities for Category C1

Failure Point	Short Description of Issue	Mapped Required Ability
FP2 Missed the Top Ranked Documents	Relevant documents are available but not ranked high enough to be included in the context. Important evidence is missed due to cut-off at top-K.	**RA2 Negative Rejection:** Refrain from answering when sufficient evidence is missing. **RA4 Counterfactual Robustness:** Avoid generating plausible but incorrect responses based on partial or misleading evidence.
FP3 Not in Context	Retrieved relevant documents are excluded from the input context during consolidation, often due to low ranking confidence.	**RA1 Noise Robustness:** Handle and filter irrelevant or semantically weak context.
FP4 Not Extracted	The answer is present in the context but is not correctly extracted by the model, often due to noise or misleading content.	**RA1 Noise Robustness:** Extract meaningful information despite noisy or distracting content.
FP7 Incomplete	The response is incomplete even though the necessary information is present across multiple documents. Complex queries requiring synthesis are not fully resolved.	**RA3 Information Integration:** Combine information from multiple sources to provide a complete and coherent answer.

3.3.7 Evaluation Dataset

Various types of datasets can be utilized to evaluate the performance and effectiveness of a system or model. The most common categories used in the context of RAG and related evaluation frameworks are outlined below:

Benchmark Dataset Standardized, publicly available datasets frequently used in research to enable consistent and comparable evaluations across different systems (e.g., **RECALL, RGB, CRUD, RAGBench**) [Gao24, Fri24, Liu23b, Che24b, Lyu25].

Synthetic Dataset Artificially generated data, often produced through simulations or rule-based procedures. These datasets allow for fine-grained control over specific properties or constraints, making them suitable for targeted evaluations [Zhu24].

Human-Annotated Dataset Datasets in which annotations (e.g., labels, relevance judgments, correctness scores) are provided by human annotators. Such datasets

are particularly valuable for tasks involving subjective assessments or complex semantic interpretations [Niu23].

Manually Curated Dataset Datasets that are manually assembled and cleaned, typically by domain experts or developers. These datasets are often used when no appropriate benchmark exists, ensuring high data quality and domain relevance [Kat25].

4 Implementation

This chapter describes the technical implementation of the system. It outlines the technologies selected, the applied retrieval methods to address the previously identified challenges, and the integration of the evaluation pipeline. Furthermore, it details which retrieval techniques were applied to the respective RAG variants to enable subsequent evaluation according to the defined criteria and metrics.

Given the comparative nature of this study, a detailed documentation of the implemented variants, tool selection, and system configuration was necessary to ensure reproducibility and transparency.

4.1 Technology Stack

This section provides a concise overview of the core technologies employed in the implementation. A detailed rationale and technical evaluation of each component are presented in the subsequent section (see Section 4.2).

The RAG-based chatbot system and its evaluation framework are built upon the following key components:

- **Response Generation (via GenAI):** The **Mistral AI** LM is utilized to generate context-aware responses based on retrieved content.
- **Vectorization: Amazon Titan Text Embedding** is used to transform user queries and document content into dense vector representations suitable for similarity-based retrieval.
- **Cloud Infrastructure:** All embedding and model inference operations are executed on **Amazon Web Services (AWS)**, ensuring scalability and secure cloud-based deployment.
- **Vectordatabase: Chroma** was employed to store the vectorized data produced by the embedding model.

- **LLM Framework: LangChain** serves as the orchestration layer for the RAG pipeline. **AWS** integration is facilitated via the `boto3` SDK.
- **Evaluation and Dataset Generation:** The **DeepEval** framework is used for automated response evaluation and the generation of synthetic test datasets, enabling reproducible benchmarking. For this purpose, **Anthropic Claude 3.5 Sonnet** is employed as the underlying LM and **Cohere Embed 3 Multilingual** is utilized as the embedding model.

4.2 Justification of Tool Selection

This section provides a detailed rationale for the selection of the technologies introduced in Section 4.1. Each tool or framework was chosen based on specific functional requirements, architectural considerations, and compatibility with the overall system design.

4.2.1 Programming Language: Python

Python was chosen as the primary programming language for this implementation due to its dominant role in AI and ML development. It offers a mature and extensive ecosystem of libraries, frameworks, and tools that support all stages of AI workflows, from data preprocessing and model development to deployment and evaluation [Ras20].

In the context of LLMs and RAG systems, **Python** is the de facto standard. Both **LangChain** and **LlamaIndex**, the leading open-source frameworks for RAG pipelines, are primarily developed in **Python**. While limited support for **JavaScript** and **TypeScript** exists, full functionality including advanced orchestration, agent-based logic and vector store integration is currently only available in the **Python** ecosystem [Tea25d, Tea25i].

The native compatibility of these frameworks with **Python** further reinforces its suitability. Consequently, the choice of programming language directly influenced the selection of supporting libraries. The following section provides a comparative analysis of **LangChain** and **LlamaIndex**, and justifies the selection of the most appropriate framework for the RAG implementation.

4.2.2 LLM Framework Selection

In the current ecosystem, two dominant open-source frameworks are widely used for the implementation of RAG: **LangChain** and **LlamaIndex**. Table 4.1 compares the two frameworks in terms of their architectural focus and selection rationale.

Table 4.1: Comparison of LLM frameworks considered for building the RAG component

Framework	Description	Exclusion Criteria for Selection
LangChain	Modular and widely adopted framework for building LLM applications. Provides tools for prompt management, chaining, agent design, and integration with vector stores, making it highly flexible for advanced RAG systems [Tea25d].	Its broad scope and abstraction layers may introduce additional complexity and potential performance overhead for lightweight RAG use cases. However, it offers extensive modularity and flexibility for scalable architectures.
LlamaIndex	Focused on indexing and retrieval of unstructured and structured data. Designed for RAG pipelines with support for various document loaders, query engines, and vector store integrations [Tea25i].	Strong in the retrieval component but lacks modular extensibility for complex multi-stage applications. Not chosen due to limitations in advanced orchestration and agent integration.

Based on the comparison in Table 4.1, **LangChain** was selected as the framework for this implementation. Its high modularity and extensibility make it particularly suitable for advanced RAG systems that are expected to evolve over time. The ability to integrate agents, tools, and complex pipeline structures ensures that the system can scale in both functionality and architectural complexity as application requirements grow.

4.2.3 Vector Database Selection

The selection of a suitable vector database in this implementation was guided not only by immediate technical feasibility but also by strategic considerations aimed at ensuring long-term viability and extensibility. While data protection and regulatory compliance, particularly with regard to personal and sensitive business information, were key drivers [Par16, dJ25, Com24], additional factors such as scalability, system interoperability, and support for advanced retrieval features were also taken into account. To maintain full control over data access and processing, only self-hostable solutions were considered, thereby allowing the system to be deployed in local, regulated environments. This approach ensures not only legal compliance but also architectural flexibility for future integration into production-grade RAG pipelines. To operationalize these criteria, a number of vector database options were assessed. Table 4.2 presents the evaluation results and corresponding exclusion justifications.

Table 4.2: Comparison of vector databases considered for integration in a locally deployable RAG chatbot

Vector DB	Description	Exclusion Criteria for Selection
Chroma	An open-source vector database that integrates seamlessly with the LangChain framework [Tea25b]. Chroma is positioned as a lightweight, developer-friendly solution [Tea25a]. It is particularly well-suited for prototyping and can be deployed locally.	Chroma is still under active development and currently lacks advanced capabilities such as clustering, replication, or large-scale scalability. It is therefore best suited for small-scale, local deployments and prototyping use cases.
FAISS	FAISS is known for its high efficiency in similarity search and clustering of dense vectors, particularly on large datasets. This performance is largely attributed to the fact that many of its core algorithms are implemented for GPU execution, enabling highly parallelized computation on local machines [Tea25j]. FAISS is also compatible with LangChain, allowing it to be integrated into the RAG pipelines [Tea25c].	Although k-means vector quantization enables high reconstruction accuracy, both memory consumption and encoding complexity increase exponentially with the size of the codebook. As a result, this method quickly reaches practical limits in terms of memory and computation when applied to large datasets or when a high number of clusters is required [Dou24]
pgvector	Pgvector is an open-source extension for PostgreSQL that enables vector similarity search within relational databases [pc25]. It is fully self-hostable, making it particularly suitable for privacy-sensitive applications with existing structured data. Pgvector is compatible with LangChain [Tea25f]	Too complex to integrate for rapid prototyping. However, well-suited for production-grade RAG systems with existing PostgreSQL infrastructure.
Weaviate	Weaviate is an AI-native, open-source vector database that supports both vector and hybrid search. Its built-in vector index compression improves memory efficiency for large datasets [Tea25o]. Additionally, it is compatible with LangChain, enabling seamless integration into RAG pipelines [Tea25h].	High system complexity due to required deployment via Weaviate Cloud Services (WCS), Docker, or Kubernetes, even in self-hosted scenarios [Tea25n]. The resulting overhead is comparatively high and may not be suitable for small-scale or prototype RAG applications.
Qdrant	Offers support for sparse vectors, enabling hybrid search strategies. It provides a production-ready service with a convenient API for storing, searching, and managing vectors, including additional payload data and advanced filtering capabilities, which makes it particularly suitable for use with AWS-based infrastructures [Tea25l]. Qdrant is also compatible with LangChain [Tea25g].	As of May 2025, Qdrant had fewer GitHub stars (23.8k) than more established alternatives like FAISS (35.2k) or Milvus (35k), indicating a comparatively smaller community. Moreover, certain advanced features such as distributed deployment are more mature in the cloud-managed version, as seen in the more limited capabilities of the official Helm Chart [Tea25m].
Milvus	Supports a wide range of deployment scenarios, from local prototyping to large-scale Kubernetes clusters capable of managing tens of billions of vectors. It is highly scalable and offers a variety of configurable indexing and search algorithms, making it well-suited for complex vector retrieval tasks [Tea25k]. Milvus is compatible with LangChain [Tea25e].	For smaller-scale prototyping, Milvus entails considerable implementation complexity, which may render it unsuitable for limited-scope projects. A lightweight variant called Milvus Lite is available, but it only supports the flat index, which limits performance and scalability options within LangChain [Tea25e].

Based on the comparative analysis, two vector databases were shortlisted: **pgvector** and **Chroma.** One of the main advantages of **pgvector** is its seamless integration with the well-established **PostgreSQL** database system. In environments where **PostgreSQL** is already in use, **pgvector** can be easily adopted, enabling direct embedding of vector functionality into an existing relational database. This makes it particularly suitable for hybrid retrieval approaches that combine structured and unstructured data.

However, for the purposes of this implementation, **pgvector** was deemed too complex and resource-intensive. As a result, **Chroma** was selected due to its lightweight architecture and ease of integration.

Nonetheless, **pgvector** remains a strong candidate for future production-grade deployments, where its robustness, scalability, and integration capabilities could offer significant long-term advantages.

4.2.4 Infrastructure Selection: AWS vs. Local Execution

For the implementation, **AWS** was selected as the infrastructure platform to enable access to high-performance GenAI models. A local deployment was not feasible due to insufficient storage and computational resources.

AWS provides direct access to models such as **MistralAI**, which was used as a core component for response generation in the RAG-based chatbot developed in this work. The selection of **Mistral Large** and the underlying rationale are discussed in detail in Section 4.2.5.

4.2.5 GenAI-Model Selections

In the implementation, various GenAI models were employed, each selected to meet specific requirements and contextual constraints. This section presents the models used, their areas of application, and the rationale behind their selection.

1. RAG Application

The model **Mistral Large** was used for response generation within the RAG system. The decision to use **MistralAI** was based on several factors: First, it is an open-weight model with strong performance capabilities [Jia23]. Second, its open-source nature [AI] allows for full deployment in a self-hosted environment, which facilitates compliance with European data protection standards (GDPR) [fSidIBb] when used appropriately and aligns with the regulatory objectives of the EU AI Act [Par]. Third, the transparency and controllability of the model architecture enable a higher degree of control over the processing of sensitive data, which is an important factor in public sector contexts [fSidIBa].

2. Evaluation Tool

The evaluation tool is designed to assess the performance of the RAG system and is likewise based on generative AI models. It fulfills two primary functions: (1) the generation of synthetic datasets for testing purposes and (2) the automated end-to-end evaluation of answer quality through explanatory feedback. The selection of suitable models is particularly critical in this context to avoid bias and overfitting due to model homogeneity [Bub23, Stu24, Zho23]. Therefore, several models were assessed and selectively employed:

2.1 Use in Synthetic Dataset Generation. Selecting an appropriate model requires consideration of **DeepEval**'s structural expectations. Specifically, **DeepEval** requires a model capable of supporting structured chat interactions, as it differentiates between the fields `input` (question), `expected_output`, `context`, and `source`, and semantically maps them into a role-based message structure [Dee25g]. This corresponds to the commonly used `messages[]` schema, which is understood by chat-compatible LLMs. For such conversational applications, **Amazon Bedrock** provides the **Converse API**, which supports structured message exchanges in the `messages[]` schema [AWS25b]. Additionally, **AWS** maintains an official list of models that are compatible with this API [AWS25g]. It is important to note that a model must not only support the **Converse API** but must also fully process the `messages[]` schema according to the conventions used by **OpenAI** and **Anthropic** [Ser25, AWS25d, AWS25f, AWS25h]. Only such models are fully compatible with **DeepEval** through the **LangChain** framework and the `ChatBedrock` class. Furthermore, the selected model must offer multilingual capabilities, in particular support for German, since both the context documents and the generated questions are in German. Lastly, regional availability must be verified to ensure that the model is available in the designated **AWS** region for deployment [AWS25e]. Based on these criteria, the remaining candidate models were evaluated and are shown in Table 4.3.

Table 4.3: Exclusion criteria for candidate GenAI models for synthetic data generation in DeepEval

Model	Exclusion Criteria for Selection
Anthropic Claude 3.5 Sonnet	*None.* Strong contextual understanding [Len24]. Delivers consistent and robust evaluations [Jau24]. Highly suitable for evaluation purposes due to its strong contextual understanding.
Mistral Large	Already deployed in the RAG component of the application. Re-use was avoided to minimize model overlap.
Meta LLaMA	Only available in AWS Bedrock via provisioned throughput, which was not suitable for this project due to higher operational costs and lack of on-demand availability in the designated region.
Writer Palmyra X5	Not available in the designated region. Additionally, there is limited publicly available documentation or benchmarking data regarding its suitability for synthetic dataset generation.

Based on the selection criteria and the exclusion conditions summarized in Table 4.3, only **Claude 3.5 Sonnet** remained as a viable option. **Writer Palmyra X5**, which was initially considered for dataset generation to avoid model overlap, had to be excluded due to its unavailability in the target **AWS** region. Consequently, **Claude 3.5 Sonnet** was selected for both synthetic dataset generation and evaluation (see justification in the following section 4.2.5). To address the associated risk of model bias and reduced diversity, all generated outputs were manually reviewed and revised where necessary to ensure high quality and minimize model-specific artifacts.

Implementation Note: Although **DeepEval** in principle supports the automated generation of synthetic test data through integration with **LangChain** and **AWS** using the **Converse API** (accessed via `boto3`), the practical implementation turned out to involve significant complexity that exceeded the intended project scope. The current version of **DeepEval** does not account for repeated throttling behavior on the **AWS** side. Mitigating this issue would have required either batching requests or implementing an exponential backoff mechanism, both of which would have necessitated substantial modifications to **DeepEval**'s internal scripts, resulting in a disproportionate implementation effort. Consequently, automated test data generation was omitted. Instead, 50 evaluation samples were manually curated from the available PDF documents to form a manually curated dataset 3.3.7. This alternative still enabled meaningful end-to-end evaluation of answer quality, as the manually curated dataset maintained a high level of content quality. Additionally, this approach *mitigated the risk of bias and overfitting* that could have arisen from using the same model for both data generation and evaluation. Full automation remains a potential enhancement for future iterations of the system (see Section 6.3).

2.2 Use in Automated End-to-End Evaluation. **Amazon Bedrock** provides a limited set of GenAI models specifically designated for evaluation tasks in RAG applications [AWS25c]. These are categorized as supported evaluator models for (1) built-in metrics and (2) custom metrics. For this implementation, only the latter are relevant, as **DeepEval** requires evaluator models that support custom metric definitions and chat-based interfaces. In addition to technical capabilities, further aspects must be considered in the selection process. As previously discussed, regional availability plays a crucial role [AWS25e]. Based on the above requirements, the supported evaluator models for custom metrics were assessed and are summarized in Table 4.4.

Table 4.4: Exclusion criteria for candidate GenAI models for automated evaluation in DeepEval

Model	Exclusion Criteria for Selection
Mistral Large	Already deployed in the RAG component. Re-use was avoided to prevent model overlap.
Claude 3.5 Sonnet	*None.* Strong contextual understanding [Len24]. Delivers consistent and robust evaluations [Jau24].
Claude 3.7 Sonnet	Technically accessible via cross-Region inference but not natively available in the selected AWS region [AWS25e]. To minimize latency and ensure data locality, this option was excluded. Despite being a newer version of Claude 3.5 Sonnet, availability constraints were prioritized.
Claude 3 Haiku 3.5	Technically accessible via cross-Region inference but not natively available in the selected AWS region. Additionally, outperformed by Claude 3.5 Sonnet in multiple benchmarks [Tea24].
Claude 3 Haiku 3	Technically accessible via cross-Region inference but not natively available in the selected AWS region. Predecessor of Haiku 3.5.
Meta LLaMA 70B Instruct	Only available in AWS Bedrock via provisioned throughput, as shown in the AWS console at the time of implementation. Since on-demand access was not available in the designated region, and provisioned throughput incurs higher costs and setup effort, the model was excluded.
Amazon Nova Pro	Limited public documentation on evaluation performance. Inference calls can take up to 50 minutes, requiring extensive timeout configuration (60 minutes) [AWS25a] and significantly delaying response cycles. Due to these inefficiencies, the model was excluded.

Based on the criteria and the comparative analysis, **Claude 3.5 Sonnet** was selected for the evaluation process. The model was chosen due to its proven contextual reasoning capabilities and its ability to produce stable, consistent evaluations across various task domains.

4.2.6 Embedding Model Selections

As outlined in Section 4.2.5, different embedding models were utilized depending on the specific application context. This section provides an overview of which embedding model was employed for each use case. The structure follows the categorization introduced in Section 4.2.5, namely: (1) RAG application, and (2) evaluation tool.

1. RAG Application

As of May 2025, **AWS** provides only two embedding models. For the RAG-based chatbot, the **Titan Embeddings v2** model was selected. This model is capable of processing and vectorizing text in multiple languages and is well-suited for general-purpose retrieval tasks.

2. Evaluation Tool

In contrast to the RAG application, the embedding model within the evaluation tool is used exclusively for the generation of synthetic data. More specifically, it serves to vectorize the entire context corpus, enabling the system to retrieve relevant segments and generate appropriate question-answer pairs based on a given query and the vectorized dataset. For this task, the **Cohere Embed 3 Multilingual** model was selected, primarily due to its strong multilingual capabilities, including reliable support for the German language.

Implementation Note: Please note that the generation of synthetic data was ultimately replaced by a manually curated dataset 3.3.7, as discussed in Section 4.2.5.

4.2.7 Evaluation Framework Selection

As discussed in Section 3.3.5, RAG systems can be evaluated using either benchmarks or automated end-to-end evaluation. This implementation focuses on automated evaluation to gain more granular insights into response quality. This section is structured into three parts: First, an overview of available evaluation tools is provided. Then, their respective evaluation metrics are presented. Finally, the tools are compared and one is selected for implementation.

Overview of Evaluation Tools

RAGAS. The *Retrieval-Augmented Generation Assessment System* (**RAGAS**) originally designed for *reference-free evaluation*, leveraging LLMs to automate the assessment process [Es24]. With the later introduction of *non-LLM-based metrics*, the framework has been extended to support *reference-based evaluation*, enabling comparison against synthetic or human-annotated datasets. Notably, several RAGAS metrics can be applied in both *LLM-based* and *non-LLM-based* evaluation settings, depending on the specific requirements of the application context [Rag25i]. RAGAS offers a broad set of evaluation metrics.

ARES. The *Automated Retrieval-based Evaluation System* (**ARES**) introduces a framework for evaluating RAG systems, focusing on the quality of retrieved information and the relevance of the generated responses [SF23]. These aspects correspond to the evaluation targets **ET1** and **ET2** (see Section 3.3.1). The approach comprises (1) generating synthetic QA pairs from corpus passages using a LM, (2) fine-tuning three lightweight judge models to evaluate ARES's core metrics of context relevance, answer faithfulness and answer relevance, and (3) applying Prediction-Powered Inference (PPI) [Ang23] to estimate statistically reliable confidence intervals based on a small set of human-annotated data.

TruLens. Gao et al. introduced **TruLens** as another evaluation tool applicable to RAG systems [Gao24]. TruLens also contributed to formalizing the concept of the **RAG Triad**, as outlined in Section 3.3.4 of Chapter *Concept*. It is a software tool designed to assess the quality and effectiveness of LLM-based applications through the use of feedback functions. These functions are not limited to post-hoc evaluation but can also be applied at runtime, providing dynamic feedback to iteratively improve the RAG pipeline. For instance, context relevance can be used as a guardrail to filter out irrelevant information before it is passed to the LLM, thereby reducing hallucinations and enhancing efficiency.

It provides a rich set of metrics, including RAG-specific ones, and supports human-in-the-loop feedback. **DeepEval** is designed for use in production environments with integration capabilities for CI/CD pipelines and a complementary cloud platform called **Confident AI** [Dee25h].

LangSmith. Is a observability and evaluation platform from **LangChain** to monitor AI applications perfomance. It can be used with or without LangChain. **LangSmith** follows a three-stage evaluation workflow: (1) instrumenting observability features such

as metrics and alerts, (2) collecting performance data during live traffic for scoring and feedback, and (3) iteratively refining prompts with version control and collaboration tools.

Evaluation Metrics per Tool

The following section presents the evaluation metrics offered by each framework, with definitions and references. The metrics are organized per tool to ensure clarity and comparability.

Table 4.5 lists the core metrics supported by **RAGAS**, including both *LLM-based* and *non-LLM-based evaluation* options.

ARES and **TruLens** implement comparable evaluation metrics that align with similar quality objectives, although both frameworks employ different terminology. For clarity and comparability, these are presented jointly in Table 4.6.

Table 4.6: Shared metrics in ARES and TruLens

Evaluation Tool(s)	Metric	Description	Source
ARES, TruLens	**Context Relevance**	Evaluates how well the retrieved context aligns with the information need of the query. Corresponds to **QS1** (see Section 3.3.2).	[SF23], [Tru25b]
	Answer Faithfulness/Groundedness (ARES/TruLens)	Assesses whether the generated response is faithful to the retrieved context. Corresponds to **QS2** (see Section 3.3.2).	[SF23], [Tru25b]
	Answer Relevance	Measures how relevant the generated answer is to the original user query. Corresponds to **QS3** (see Section 3.3.2).	[SF23], [Tru25b]

The following Table 4.7 provides an overview of the metrics implemented in **DeepEval** for evaluating RAG system performance. All listed metrics apply the LLM-as-a-judge paradigm and provide justifications for the assigned scores. In addition, to the standard RAG-specific metrics, two further metrics called `G-Eval` and `RAGAS` are included due to their relevance for specialized or extended evaluation needs.

LangSmith provides a variety of evaluation metrics, including specific ones for RAG systems. A fundamental distinction is made between whether a reference output is required and which evaluation mode is applied [Lan25a]:

- **Offline evaluation:** Used when prompts require a reference output. Commonly applied in answer correctness evaluation, where ground truth answers are available.

Table 4.5: RAGAS metrics

Evaluation Tool	Metric	Description	Type	Source
RAGAS	**Context Precision**	Measures the proportion of relevant information in the retrieved context, based on the traditional Precision metric (see Section 1).	LLM-based Non-LLM-based	[Rag25b]
	Context Recall	Measures the proportion of relevant information successfully retrieved from the available sources, based on Recall (see Section 2).	LLM-based Non-LLM-based	[Rag25c]
	Context Entities Recall	This metric is an advanced version of the Recall metric, which assesses the recall of retrieved context based on the number of relevant entities.	LLM-based	[Rag25a]
	Noise Sensitivity	Measures robustness against irrelevant or noisy input. Aligns with ability **RA1** (see Section 3.3.2).	LLM-based	[Rag25h]
	Response Relevancy	Assesses how well the generated response addresses the query. Maps to quality score **QS3** (see Section 3.3.2).	LLM-based	[Rag25j]
	Faithfulness	Evaluates whether the generated response remains truthful to the context. Aligned with **QS2** (see Section 3.3.2).	LLM-based	[Rag25d]
	When using LangChain as the LLM framework, RAGAS recommends additional metrics:			
	LLMContextRecall	Assesses the alignment between retrieved context and reference answer, estimating recall without manual annotations. Computes TP and FN based on their comparison, following the principles of the Recall metric (see Section 2).	LLM-based	[Rag25g, Rag25e]
	Factual Correctness	Assesses factual accuracy by comparing the response to a reference using Precision, Recall, and F1 (see Sections 1, 2, 3). Employs claim-based evaluation and natural language inference.	LLM-based Non-LLM-based	[Rag25f, Rag25e]
	Faithfulness	Also part of the core RAGAS metrics but explicitly recommended for RAG evaluation with LangChain.	LLM-based	[Rag25e]

- **Online evaluation:** Applied for prompts that do not require a reference. Suitable for real-time or streaming applications.
- **Pairwise evaluation:** Compares answers from different RAG chains based on user-defined criteria (e.g., tone, style), rather than factual correctness.

All **LangSmith** metrics follow the *LLM-as-a-judge* paradigm. The following Table 4.8 provides an overview of relevant metrics.

Table 4.7: Overview of DeepEval metrics for evaluating RAG systems.

Evaluation Tool	Metric	Description	Source
DeepEval	**Answer Relevancy**	Assesses how relevant the *actual output* produced by the LLM is with respect to the input. This metric corresponds to **QS3** (see Section 3.3.2).	[Dee25a]
	Faithfulness	Evaluates whether the actual output is factually consistent with the retrieved context. This corresponds to **QS2** (see Section 3.3.2).	[Dee25e]
	Contextual Precision	Derived from classical precision (see Section 1). Evaluates whether the most relevant nodes in the retrieved context are ranked higher than irrelevant ones, based on semantic alignment.	[Dee25b]
	Contextual Recall	Based on classical recall (see Section 2). Assesses how well the retrieved context semantically covers the key information from the expected output, allowing flexible, meaning-based comparison.	[Dee25c]
	Contextual Relevancy	Evaluates the overall semantic relevance of the retrieved context with respect to the input. This corresponds to **QS1** (see Section 3.3.2).	[Dee25d]
	Additional Evaluation Metrics:		
	G-Eval	A flexible evaluation framework that combines the LLM-as-a-judge paradigm with CoT prompting. It enables the evaluation of LLM outputs against custom, user-defined criteria and is particularly suited for subjective or use-case-specific assessments.	[Dee25f]
	RAGAS	DeepEval integrates the RAGAS library, which includes the following metrics: `RAGASAnswerRelevancyMetric`, `RAGASFaithfulnessMetric`, `RAGASContextualPrecisionMetric`, and `RAGASContextualRecallMetric`. While functionally similar to DeepEval's native RAG metrics, minor implementation differences exist. For consistency, DeepEval's default metrics are generally preferred.	[Dee25i]

Comparison and Evaluation Tool Selection

Building on the previously presented evaluation metrics, the following comparison provides a structured overview of key evaluation tools for RAG systems. The goal is to support a transparent and well-founded tool selection process. For each tool, the supported metrics are listed, along with key advantages and exclusion criteria. The goal is to provide a transparent rationale for the final tool selection. All tools considered fulfill the general evaluation targets **ET1** and **ET2** as defined in Section 3.3.1.

Table 4.8: Overview of LangSmith metrics for evaluating RAG systems.

Evaluation Tool	Metric	Description	Source
LangSmith	**Document Relevance**	Measures how relevant the retrieved context is to the user query. This aligns with **QS1** (see Section 3.3.2).	[Lan25a]
	Answer Faithfulness	Evaluates whether the generated answer is grounded in the retrieved context, avoiding hallucination. Related to **QS2** (see Section 3.3.2).	[Lan25a]
	Answer Helpfulness	Assesses whether the answer addresses the user's query and provides useful information. This corresponds to **QS3** (see Section 3.3.2).	[Lan25a]
	Answer Correctness	Compares the generated response to a ground truth reference to assess factual correctness. This metric requires a reference output and conceptually overlaps with both **QS2** and **QS3**, as it evaluates whether the response is both grounded and appropriately addresses the query.	[Lan25a]
	Pairwise Comparison	Compares two model outputs side by side, useful for evaluating stylistic or preference-based criteria. Often applied in summarization or generation quality comparisons.	[Lan25a]

Evaluation Tool Selection. **DeepEval** was selected as the evaluation framework due to its robust set of standard RAG evaluation metrics and its extensible architecture for defining custom criteria. While **LangSmith** offers stronger native integration with **LangChain**, **DeepEval** provides greater long-term flexibility for extending the evaluation process. Although user-defined metrics via **G-Eval** were not used in this thesis, their availability makes **DeepEval** more suitable for future expansions. A detailed comparison and justification is provided in Table 4.9.

Metric Selection and Alignment. The selected evaluation framework **DeepEval** determines the applied metrics. To assess the system's retrieval and generation quality, the following core metrics were used: **Answer Relevancy**, **Faithfulness**, **Contextual Precision**, **Contextual Recall**, and **Contextual Relevancy**. These align with the defined quality aspects **QS1**–**QS3** (see Section 3.3.2) and operationalize the evaluation criteria introduced in Section 3.1.3, such as *Document Relevance*, *Answer Precision*, and *Hallucination Resistance*.

While **DeepEval** offers **G-Eval** for implementing custom metrics, this thesis prioritizes standardized evaluation to ensure comparability and reproducibility. Future extensions could integrate **G-Eval** to address domain-specific quality dimensions once a reliable baseline has been established.

Table 4.9: Overview of selected evaluation tools for RAG systems with advantages and exclusion criteria

Evaluation Tool	Aspects / Abilities	Metrics	Advantages	Exclusion Criteria	Source
RAGAS	QS1, QS2, QS3, RA1	Context Precision, Context Recall, Context Entities Recall, Noise Sensitivity, Response Relevancy, Faithfulness, LLMContextRecall, Factual Correctness	Hybrid evaluation (reference-based and reference-free). No human-annotated dataset required (if preferred). Dataset can be generated by LLM. Faster evaluation due to lack of human annotation.	**Familiarity bias:** Preference for predictable phrasing. **Skewed rating distributions:** Inconsistent scoring, limits comparability. **Anchoring effects:** Early judgments influence later ratings.	[Wan23c]
ARES	QS1, QS2, QS3	Context Relevance, Answer Faithfulness, Answer Relevance	Efficient evaluation with minimal human annotation (150–300 data points). Improved accuracy compared to the RAGAS framework (as of March 2024).	**Limited annotation scale:** More labeled data required for higher reliability. **High computational demand:** Requires 32GB GPU memory and substantial runtime. **Language restriction:** Currently supports only English (as of March 2024). **Other:** No support for generating synthetic datasets	[SF23]
TruLens	QS1, QS2, QS3	Context Relevance, Groundness, Answer Relevance	Implements the RAG Triad metrics and allows the definition of custom evaluation functions via its feedback function API, offering flexibility for use-case-specific quality assessments.	No support for generating synthetic datasets. Focuses solely on evaluation of existing outputs	[Tru25b, Tru25a]
DeepEval	QS1, QS2, QS3	Answer Relevancy, Faithfulness, Contextual Precision, Contextual Recall, Contextual Relevancy, G-Eval, RAGAS	In addition to standard RAG metrics, DeepEval supports G-Eval, enabling the definition of custom evaluation criteria for domain-specific or subjective tasks. It also offers CI/CD integration, making it well-suited for continuous evaluation in production environments. The framework additionally supports RAGAS-based metrics for broader compatibility.	While highly versatile, certain advanced use cases may still require adaptation or additional customization, as DeepEval is actively evolving.	[Dee25h]
LangSmith	QS1, QS2, QS3	Document Relevance, Answer Faithfulness, Answer Helpfulness, Answer Correctness, Pairwise Comparison	Well-suited for LangChain-based applications, as it originates from the same development environment. This facilitates seamless integration, systematic evaluation, and iterative improvement of RAG systems. LangSmith supports both online and offline evaluation modes as well as pairwise comparisons and can be combined with unit tests.	Currently, it does not offer fully user-defined evaluation metrics (e.g., as provided by DeepEval's G-Eval).	[Lan25a]

4.3 Retrieval Method Selection and Functional Categorization

In line with the advanced RAG architecture described in Section 2.4.3, the retrieval process in this thesis is structured into four functional stages: indexing, pre-retrieval, core retrieval, and post-retrieval. This staged decomposition aligns with the failure point categorization defined in **Category C1**, which addresses key challenges in information preparation. It therefore serves as a conceptual and operational framework for selecting and assigning retrieval methods to specific mitigation tasks.

Specifically, this section addresses **FP2** (Missed the Top Ranked Documents), **FP3** (Irrelevant Context), and **FP4** (Noise Overload), due to their significant influence on context quality. The embedding model used throughout the retrieval process was constrained by infrastructure limitations on AWS and is discussed separately in Section 4.2.6.

While **FP7** (Incomplete Answers) also falls under **Category C1**, it is not addressed in detail in this work due to time constraints.

4.3.1 Indexing

The indexing stage serves as the initial step in the retrieval pipeline and is responsible for preparing documents in a structured and retrievable format. Within this stage, chunking plays a central role by defining the granularity of retrievable units and thereby directly influencing retrieval effectiveness.

Chunking Strategy. Although not a retrieval method in the strict sense, the *chunking strategy* is a foundational aspect of the pre-retrieval stage. It structures retrievable units and mitigates **FP4** by carefully balancing chunk size and overlap: overly small chunks risk omitting relevant context, while excessively large ones may introduce irrelevant content. Similar chunking considerations were proposed in [Guu20, Kar20].

4.3.2 Pre-Retrieval Method Selections

In the implemented retrieval setup, the pre-retrieval stage was designed to optimize input representations before document retrieval takes place. Two key components were incorporated at this stage: a carefully defined *chunking strategy*, which determines the structure and granularity of retrievable units, and the use of *multi-query rewriting* to improve query formulation and increase document coverage.

MultiQueryRetriever. The **MQR** used in this setup aligns with the approach described in the Concept chapter and is implemented via **LangChain** [Lan25d]. While not derived from a specific paper, it follows the same core idea: generating diverse reformulations of the user query, retrieving documents for each variant, and aggregating the results. This improves coverage and reduces irrelevant context, thereby mitigating **FP3**.

4.3.3 Core-Retrieval Method Selections

This stage targets the core retrieval process, primarily addressing **FP2** and **FP3**. Depending on the specific retrieval configuration, both dense and sparse techniques are employed. Additionally, **FP4** is partially mitigated through reranking strategies.

Similarity Search. As described in Section 3.2.3, **Similarity Search** is a dense retrieval method that identifies semantically similar sentences via vector comparison in embedding space. It enhances recall and helps mitigate **FP2**.

BM25. The sparse retrieval baseline **BM25** estimates document relevance based on **TF**, **IDF**, and length normalization. It serves as a strong complement to dense methods by boosting keyword-matching performance, thereby addressing both **FP2** and **FP3**.

MMR. As a reranking strategy, **MMR** balances query relevance and content diversity across retrieved candidate documents. This helps reduce redundancy (**FP2**) and promotes coverage of complementary information, which may also contribute to mitigating irrelevant context (**FP3**) and, to some extent, noise overload (**FP4**).

4.3.4 Post-Retrieval Method Selections

At the end of the retrieval pipeline, two post-retrieval methods are employed to refine the quality and relevance of retrieved content before generation.

ParentDocumentRetriever. This **LangChain** method [Lan25e] builds on the **Parent-Child Embedding** strategy described in the *Concept* chapter. While the initial retrieval is conducted over fine-grained child chunks stored in a vector database, the final context returned to the LM includes their corresponding parent documents. This design allows for more coherent and informative inputs at generation time and is therefore categorized as a post-retrieval method. By expanding the retrieved context to the parent level, the method enhances semantic completeness and mitigates **FP3**. However, due to the

inclusion of broader textual segments, it may also introduce additional noise, potentially increasing the risk of **FP4**.

EnsembleRetriever. Implemented via **LangChain** [Lan25b], this method combines the outputs of multiple retrievers using the **RRF** algorithm. **RRF** reranks documents by aggregating their ranks across different retrieval outputs, giving preference to consistently high-ranked results. This improves robustness and recall, mitigating **FP2** and **FP3**.

ContextualCompressionRetriever. As discussed in **Contextual Compression**, this method reduces noise by retaining only the most relevant portions of retrieved documents. The implementation used in this setup is based on **LangChain** and includes two types of compression strategies [Lan25c]:

- **LLMChainFilter:** Filters entire documents based on their relevance to the query, thereby mitigating **FP3**.
- **LLMChainExtractor:** Extracts only the most relevant content spans from each document, addressing both **FP3** and **FP4**.

These post-retrieval mechanisms enhance precision and ensure that the generation model receives focused and contextually appropriate input.

4.4 Evaluation Integration and Synthetic Dataset Generation

As discussed in Section 4.2.7, **DeepEval** was selected as the evaluation framework for this study. Section 3.3.7 outlined potential strategies for constructing evaluation datasets. This chapter describes how **DeepEval** was technically integrated into the system and explains the process of generating and validating synthetic evaluation data. Implementation-specific adjustments for **AWS** compatibility and custom components are also addressed.

4.4.1 Synthetic Dataset Generation

The evaluation dataset was designed around administrative directives (*Weisungen*) to assess whether the chatbot can effectively handle legal content and assist users in locating and referencing specific regulations in public administration contexts.

To generate synthetic evaluation data, referred to as 'goldens' in **DeepEval**, a custom script was implemented based on the official documentation [Dee25h]. Since the default

configuration uses **ChatGPT**, custom embedding and response models were integrated to comply with **AWS** infrastructure constraints. The `generate_from_documents` function was used for question generation, as the source documents were well-suited for this approach.

As detailed in Section 4.2.5, fully automated dataset generation was ultimately omitted due to integration limitations and reliability issues with **AWS**. Instead, a manually curated dataset of 50 evaluation samples was used. The structure of the evaluation samples is summarized in Table 4.10.

Table 4.10: Structure of evaluation samples

Field	Description
Input	User query provided to the system
Context	Retrieved context passages
Source File	Origin of retrieved documents
Expected Output	Ground-truth answer
Actual Output	Not applicable during dataset generation

4.4.2 Automatated End-to-End Evaluation Integration

Building upon the selected evaluation metrics 4.2.7, the end-to-end evaluation process was implemented using **DeepEval**'s framework [Dee25h], with necessary adaptations for **AWS**-based models used for embedding and generation (see Section 4.1). JSON formatting issues, mainly caused by escaped characters such as `\"`, leading to invalid evaluation inputs. Since **LangChain** did not provide a sufficiently robust JSON parser for the evaluation process, a custom solution was implemented to maintain input consistency and prevent evaluation failures.

To accelerate the evaluation process, the golden sample assessments were parallelized. Additionally, custom threshold values were set for each metric to reflect the open-ended nature of QA tasks (see Table 4.11 for an overview).

Table 4.11: Evaluation Metrics and Thresholds

Metric	Threshold	Description
Contextual Precision	0.6	Tolerates minor irrelevant content if core information is present
Contextual Recall	0.7	Ensures that most relevant context is captured
Contextual Relevancy	0.7	Balances approximate matches and semantic fidelity
Answer Relevancy	0.75	Enforces alignment between question and response
Faithfulness	0.8	Prioritizes factual correctness and discourages hallucinations

The defined thresholds aim to balance tolerance for variation in natural language with the need for semantic accuracy and alignment with the retrieved context.

4.5 System Integration

This section outlines how the selected retrieval methods were integrated into the system architecture. To enable a comparative analysis, three RAG-based chatbot variants were implemented, each representing a distinct retrieval configuration.

A shared preprocessing step is the *chunking strategy*, which will be introduced first, as it serves as a common foundation for all variants. All approaches also share the same generation component: the **Mistral Large** LM with a temperature setting of 0, ensuring deterministic outputs strictly grounded in the retrieved context.

While the selected evaluation metrics (see Section 4.2.7) cover key quality criteria such as *Document Relevance, Answer Precision* and *Hallucination Resistance*, they do not include system efficiency. Therefore, a **custom timer** was implemented in all approaches to measure average response time as an additional evaluation dimension.

The document corpus stored in the **Chroma** vector database consists of 107 official directives derived from SGB I, SGB II, and selected sections of SGB III, specifically covering general provisions, unemployment insurance, and activation and reintegration policies.

In addition, the corpus includes recent administrative guidelines such as 202401006, 202401009, and 202406008, reflecting updated legal interpretations resulting from legislative changes. Their inclusion ensures comprehensive coverage of current regulatory developments.

4.5.1 Chunking Strategy Across All RAG Approaches

A uniform chunking pipeline was applied across all systems to ensure consistent and high-quality retrievable units. This process minimizes noise and preserves contextual coherence. The key preprocessing steps include:

- **Removal of formatting artifacts** (e.g., hyphens, excess whitespace, tables of contents).
- **Exclusion of low-value segments** (e.g., recurring headers and footers).
- **Selective inclusion of metadata** to enrich document context.

- **Preservation of cohesive structures** (e.g., tables, bullet lists).
- **Vector storage with timestamping and hashing** to prevent duplication.
- **Filtering of irrelevant or meaningless chunks** (e.g., placeholder pages, isolated footnotes).
- **Table extraction using `PyMuPDF` to preserve structure and semantics.**

4.5.2 Approach 1: Naive RAG-based Chatbot.

This baseline system uses a conventional RAG pipeline with dense retrieval (see Figure 4.1, left section). The chunking configuration from Section 4.5.1 is applied (chunk size: 400 tokens; overlap: 100). Retrieval is performed using a dense vector store with **Similarity Search** as the method, returning the top-7 results. This setup helps mitigate **FP4** through noise-aware chunking.

4.5.3 Approach 2: Hybrid Multi-Query RAG Chatbot.

This variant combines **Multi-Query Retrieval** and **BM25** in a hybrid setup (see Figure 4.1, middle section). The `MultiQueryRetriever` generates three query reformulations, each executed via dense retrieval. **MMR** reranks the top-10 candidates to select 5 final documents, using `lambda_mult = 0.98` emphasizes relevance over diversity in the selection process. **BM25** returns the top-3 results based on lexical scoring. The outputs are merged via **LangChain**'s `EnsembleRetriever` (weights: 0.6 **Multi-Query**, 0.4 **BM25**). `LLMChainFilter` was initially used for compression but was replaced by `LLMChainExtractor` due to over-filtering (see Section 4.3.4). This setup addresses **FP3** and **FP4** by enhancing retrieval coverage and precision.

4.5.4 Approach 3: Hybrid Parent-Document RAG-based Chatbot.

This variant incorporates the **Parent-Child Embedding** strategy (see Figure 4.1, right section). Child chunks (400 tokens with 100-token overlap) are embedded into the **Chroma** vector database, while parent chunks (2000 tokens with 200-token overlap) are stored locally. Initial retrieval is performed on the child chunks, and the corresponding parent segments are returned via the `ParentDocumentRetriever` (top-5 results). Retrieval uses a dense vector store with **Similarity Search** as the primary method. Additionally, **BM25** retrieval contributes three additional results. Both result sets are merged using the `EnsembleRetriever`, weighted at 0.8 for parent-based retrieval and 0.2 for **BM25**.

To mitigate potential redundancy from large parent chunks, a custom duplicate filtering mechanism is applied. The `ContextualCompression` module was excluded due to its latency overhead. This setup addresses **FP3** and **FP4** by expanding context and reducing noise, while also supporting **FP2** through ensemble-based retrieval.

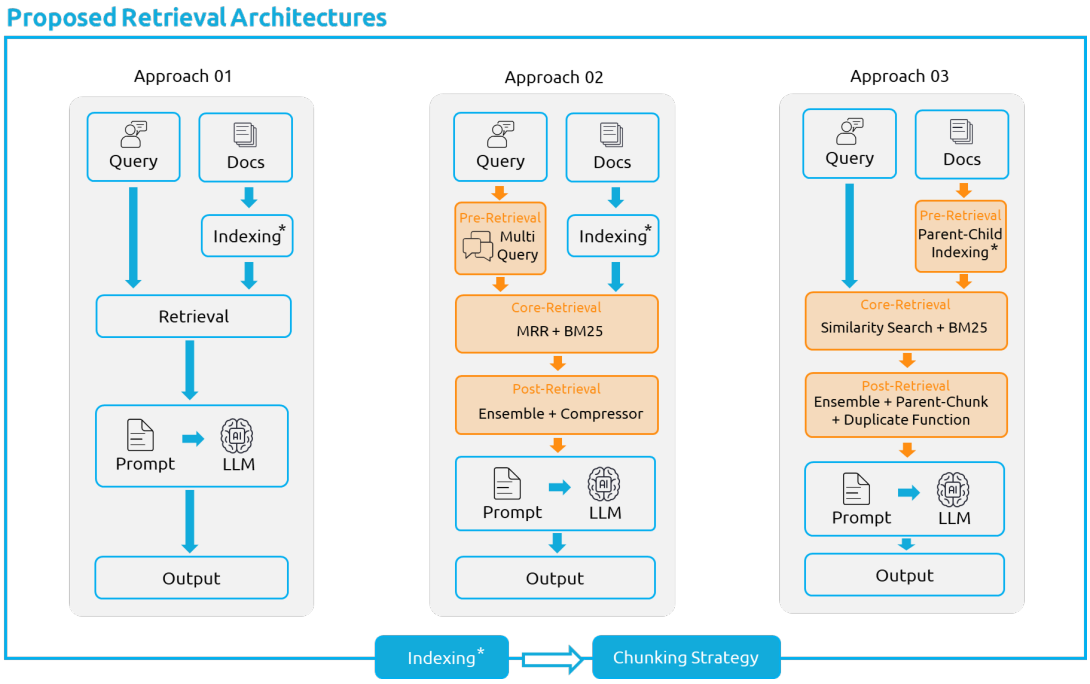


Figure 4.1: Own illustration of the architectural design of all three approaches, adapted from [Gao24].

5 Evaluation

This chapter evaluates the selected RAG approaches using the previously defined criteria and metrics. The analysis is based on the adapted SACAM methodology (see Section 1.3) and addresses research questions **RQ3** to **RQ5** (as introduced in Section 1.2).

5.1 Objectives of the Evaluation

The evaluation aims to systematically assess the selected RAG approaches with respect to their effectiveness, reliability, and suitability for QA scenarios. It follows a structured procedure to define evaluation criteria and link them to specific system functions.

In order to answer the main research question (**RQ1**), the following sub-questions guide the evaluation:

- **RQ3:** What evaluation metrics should be used to assess and compare RAG approaches in terms of performance and reliability?
- **RQ4:** How do the selected RAG approaches perform according to the defined evaluation criteria?
- **RQ5:** What impact do these RAG approaches have on the resulting response quality, particularly regarding accuracy, relevance, and coherence?

5.2 Evaluation Design and Setup

This section outlines the technical setup, data foundation and evaluation procedures used in line with the overall methodology introduced earlier.

5.2.1 Evaluation Dimensions and Criteria

The evaluation is structured along several core dimensions to ensure comparability and to cover relevant system characteristics. The following summary outlines the key focus areas of the evaluation:

Application Context: Within the scope of QA tasks (see Section 3.1), three RAG approaches are subjected to comparative evaluation, with the objective of contributing to the resolution of the main research question **RQ1**.

Evaluation Criteria: The evaluation is guided by quality dimensions tailored to QA tasks. These were derived in the context of this study and include: *Document Relevance*, *Answer Precision*, *Response Time/Latency* and *Hallucination Resistance* (see Section 3.1.3).

Retrieval Types: The evaluated RAG approaches make use of both dense and hybrid retrieval strategies (see Section 4.5).

Evaluation Targets: Based on the *Evaluation Criteria* and the focus on **Category C1**, **ET1** is of primary relevance. **ET2** is additionally included to examine whether the results from **ET1** have an observable effect on the generated responses.

Robustness Evaluation: In addition to the primary Evaluation Targets, selected *failure points* (**FPs**; see Section 5.2.2) are mapped to the defined *Required Abilities* (**RAs**; see Section 3.3.2) to assess each system's behavior under challenging retrieval conditions (see Section 5.4.2).

5.2.2 System Integration and Method Assignment

The applied technologies have been described in detail in the implementation chapter (see Section 4.1). Section 4.3 introduced the selection and categorization of retrieval methods, along with their mapping to **Category C1** and the corresponding to **FP2**, **FP3**, and **FP4**. Although **FP7** also belongs to **Category C1**, it is not included in the evaluation due to its specific requirements regarding multi-document reasoning.

Building on this, the integration of the three distinct RAG approaches for comparative evaluation is discussed in Section 4.5. The underlying corpus includes a domain-specific dataset of 107 legal directives from SGB I, II and III, ensuring coverage of recent legislative amendments and current administrative directives.

5.2.3 Evaluation Integration and Metric Assignment

The selection of evaluation metrics (see Section 4.2.7) was guided by a multi-stage derivation process. Initially, key evaluation criteria were conceptually defined in Section 3.1.3, based on quality demands in QA-focused RAG systems. These included *Document Relevance*, *Answer Precision*, *Response Time*, and *Hallucination Resistance*.

These criteria were then abstracted into three *Primary Quality Score* (QS1-QS3) as outlined in Section 3.3.2. They can be mapped to the overarching evaluation dimensions of the **RAG Triad** (see Section 3.3.4). These dimensions served as the theoretical basis for metric selection.

DeepEval was chosen as the evaluation framework due to its modular design and its ability to operationalize these dimensions using automated metrics (see Section 4.4.2). The following threshold settings for each metric were selected:

- **Contextual Precision**: 0.6
- **Contextual Recall**: 0.7
- **Contextual Relevancy:** 0.7
- **Answer Relevancy:** 0.75
- **Faithfulness:** 0.8

Since **DeepEval** does not natively support latency measurements, the evaluation criterion *Response Time/Latency* was implemented separately using **custom timers** in each RAG component (see Section 4.5). This ensured full operationalization of all originally defined evaluation criteria.

The initial idea of automated dataset generation was omitted due to technical constraints and integration overhead with **AWS**, as detailed in Section 4.4.1. Instead, 50 high-quality samples were manually curated to enable meaningful evaluation and avoid potential model bias and overfitting.

For automated evaluation tasks, **Anthropic Claude 3.5 Sonnet** was used as the underlying LM, and **Cohere Embed 3 Multilingual** served as the embedding model for context representation (see Section 4.1).

5.3 Implementation and Execution

Building on the metric configuration and tool selection described above, the evaluation was carried out using **DeepEval**'s integrated execution environment. The **Confident AI** dashboard was employed to visualize and compare results across all test cases and RAG approaches.

For each of the 50 evaluation samples, the platform provided structured outputs, including the generated answer, expected reference, retrieved context, and the corresponding metric scores with model-generated justifications. This setup allowed for a transparent assessment of system behavior under comparable conditions.

Table 5.1: Structure of Evaluation Output per Test Case

Field	**Description**
Test Case Name	Unique identifier of the evaluation sample
Success	Overall success status of evaluation execution
Input	Generated question for the test case
Expected Output	Ground-truth answer used for evaluation
Actual Output	Model-generated response
The following fields are recorded per metric:	
<Metric>_success: Boolean indicating if score passes threshold	
<Metric>_score: Computed score (numeric value)	
<Metric>_reason: Textual explanation for assigned score	
<Metric>_verboseLogs: Detailed logs with model verdicts and statement-level reasoning	
<Metric>_threshold: Threshold value applied for this metric	

To support the interpretation of metric results, each evaluation instance was manually reviewed in addition to the automated scoring. This process involved inspecting the generated output, the retrieved context, the expected answer, and the reasoning provided by the *LLM-as-a-judge* (**Claude 3.5 Sonnet**). The goal was to understand how specific scores were assigned and to ensure that the evaluation system had accurately interpreted context and semantic intent. Particular attention was paid to cases where score justifications appeared unclear or borderline, in order to critically assess the reliability of the judgment process.

The evaluation platform supported pairwise comparative analysis of the implemented RAG approaches. By comparing metric values across test cases, it was possible to identify performance differences between two systems at a time. This view enabled the detection

of recurring patterns, such as one approach consistently achieving higher **Contextual Precision**, while another demonstrated stronger **Faithfulness**. These observations informed the interpretation of evaluation outcomes and supported answering research questions **RQ4** and **RQ5**.

Additionally, test cases that exhibited unusual metric combinations or inconsistencies were documented and analyzed in detail. These qualitative observations are synthesized in Section 5.4.1 and serve to complement the quantitative evaluation results.

5.4 Evaluation Results

The following analysis addresses the evaluation criteria of *Document Relevance*, *Answer Precision* and *Hallucination Resistance* through the combined quantitative and qualitative findings presented in Subsection 5.4.1 as well as the subsequent interpretation in Subsection 5.4.2. The criterion *Response Time/Latency*, while likewise situated within Subsection 5.4.1, is examined separately in Subsubsection 5.4.1 using empirical response time measurements.

Potential threats to validity and limitations of the evaluation setup are discussed in Subsection 5.4.3, including annotation uncertainty, metric interpretation, and contextual ambiguity.

5.4.1 Quantitative and Qualitative Results

This section combines quantitative metrics with qualitative findings from selected cases to evaluate the performance of the three RAG approaches across five dimensions: **Contextual Precision, Contextual Recall, Contextual Relevancy, Answer Relevancy** and **Faithfulness**.

General Overview

An overview of the average metric scores computed by ConfidentAI for all three RAG approaches is provided in Table 5.2. Approach 03 achieves the highest values for both **Contextual Precision** and **Contextual Recall**, indicating superior retrieval accuracy and coverage. In contrast, **Contextual Relevancy** is highest in **Approach 01**, suggesting a stronger alignment between retrieved content and the ground truth in that case.

Table 5.2: Comparison of the average metric scores of the three RAG approaches

Metric	Approach 01	Approach 02	Approach 03
Contextual Precision	0.72	0.33	0.75
Contextual Recall	0.78	0.76	0.83
Contextual Relevancy	0.59	0.31	0.55
Answer Relevancy	0.98	0.99	0.98
Faithfulness	0.97	0.94	0.96

A visual comparison is provided in Figure 5.1, which illustrates the relative performance differences and highlights key trade-offs between the approaches. Notably, despite the retrieval-related advantages of **Approach 03**, its **Answer Relevancy** and **Faithfulness** scores are nearly equivalent to those of **Approach 01**. Moreover, **Approach 01** slightly exceeds **Approach 03** in **Faithfulness** (0.97 vs. 0.96). These findings indicate that improvements in retrieval metrics do not necessarily translate into higher answer quality. Possible underlying factors, such as contextual alignment, are discussed further in Section 5.4.2.

Despite relying only on a basic *chunking strategy* without additional enhancements, Approach 01 achieves a high **Faithfulness** score, suggesting that even simple retrieval setups can yield accurate and consistent answers when the context is well-structured.

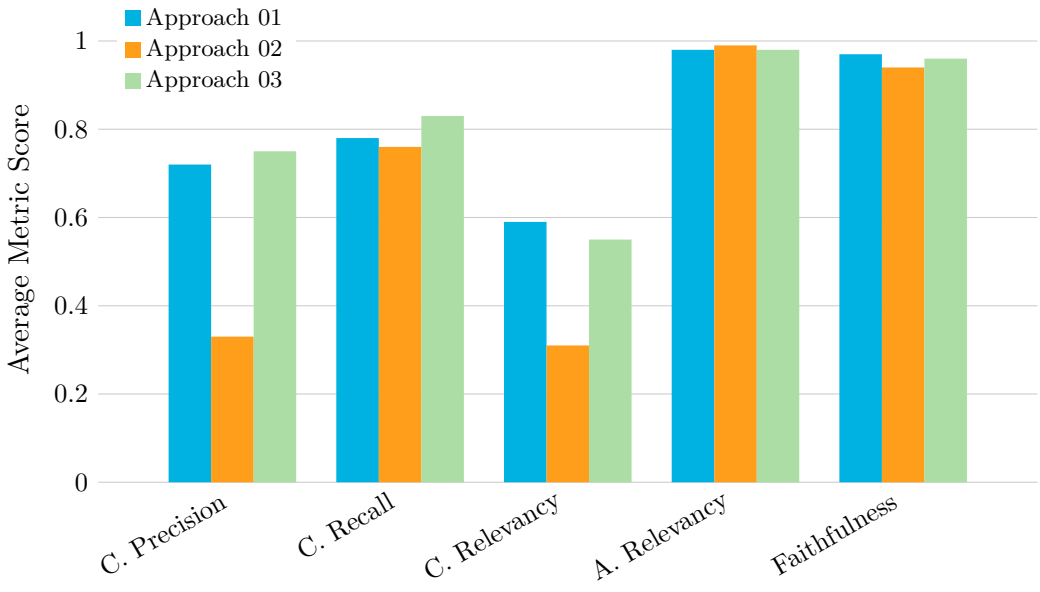


Figure 5.1: Comparison of average metric scores across RAG approaches

Pass Rate Perspective

A closer inspection of the pass rates in Table 5.3 reveals notable differences between the approaches. For example, although **Approach 02** reaches only 6% in **Contextual Precision** and 0% in **Contextual Relevancy**, it achieves a comparable score in **Answer Relevancy** (98%) and the highest result in **Faithfulness** (94%).

Table 5.3: Pass Rate of the three RAG approaches

Metric	Threshold	Approach 01	Approach 02	Approach 03
Contextual Precision	0.60	80%	6%	80%
Contextual Recall	0.70	68%	64%	72%
Contextual Relevancy	0.70	32%	0%	20%
Answer Relevancy	0.75	98%	98%	98%
Faithfulness	0.80	88%	94%	90%

These discrepancies between **ET1** and **ET2** are further illustrated in Figure 5.2. The bar chart provides a visual overview of the pass rate distribution across all metrics, highlighting how strengths and weaknesses differ across evaluation dimensions.

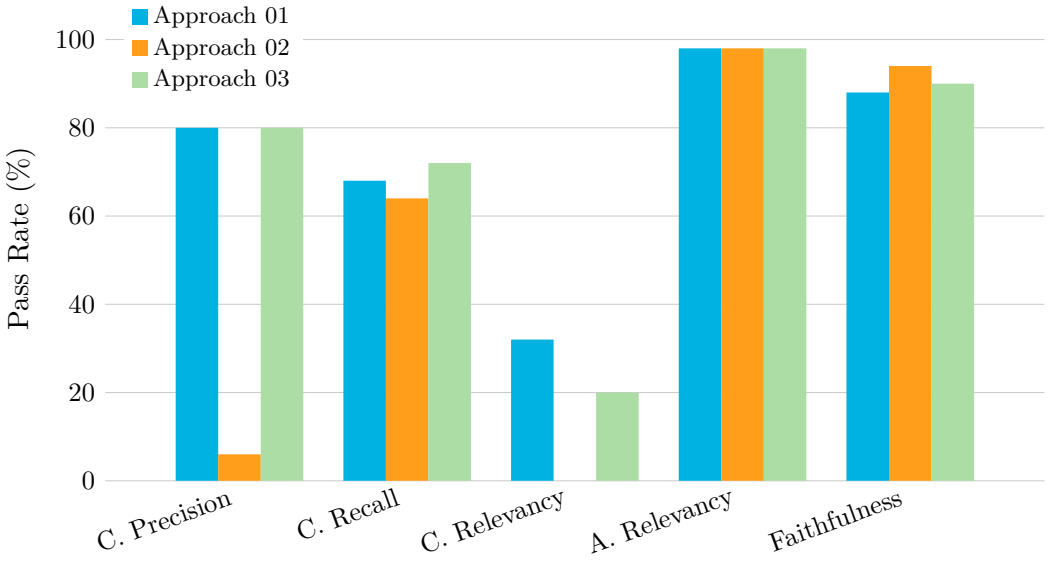


Figure 5.2: Pass rate distribution across all evaluation metrics for the three RAG approaches

Case-Based Insights

While the quantitative metrics outline general strengths and weaknesses across approaches, specific test cases provide deeper insight into how different retrieval dynamics affect answer quality. Table 5.4 highlights selected examples where unusual metric

combinations occur such as low contextual scores alongside high **Faithfulness** and helps explain the interaction between retrieval noise, model behavior, and answer correctness.

Note: The evaluation system refers to the individual context elements retrieved during information retrieval as *nodes*. In the context of this thesis, these units are consistently referred to as *chunks*. The terms are considered equivalent and interchangeable for purposes of analysis.

Table 5.4: Summary of selected test cases highlighting key retrieval-to-answer dynamics

Test Case	Approach	Key Insight	Implication
TC6	**01**	No relevant chunks retrieved, but answer is accurate and faithful	LM likely relied on prior knowledge, high faithfulness despite 0.00 in precision/recall
TC24	**01**	Retrieved context is plausible but lacks legal specificity (e.g., no supervisory complaint)	Faithful output may obscure gaps in retrieval specificity
TC42	**01**	The retrieval output lacks relevant legal provisions, and the model misinterprets the content by conflating disbursement waiver criteria with reimbursement thresholds. Despite the absence of supporting context, it generates a plausible-sounding rule based on a misreading of the 50€ clause.	Demonstrates high answer relevancy despite low contextual and faithfulness scores. Highlights the risk of semantically precise hallucinations when no evidential grounding is available.
TC10	**02**	Context includes partially relevant and misleading content	Model interprets context slightly wrong, faithfulness still above threshold
TC7	**02**	Relevant chunks present but diluted by irrelevant context	Contextual relevancy suffers despite high precision and faithfulness

Continued on next page

Test Case	Approach	Key Insight	Implication
TC26	**02**	Relevant chunks (6–9) ranked below irrelevant ones (1–5), despite containing key legal content	Retrieval ranking failure significantly lowers contextual precision, suggests reranking is essential for hybrid approaches
TC11	**03**	The retrieved context covers approximately 50% of the expected output. The generated answer aligns with the user query and remains faithful to the available context but does not include all expected elements.	The model omits unsupported details instead of hallucinating, demonstrating that high faithfulness and answer relevancy are still achievable despite limited retrieval coverage.
TC14	**03**	Output directly answers the question but simplifies legal details (only §104 cited, though §§102–105 are relevant)	High answer relevancy despite low context precision. Model robust, but loses legal nuance due to retrieval ranking and context noise
TC39	**03**	Answer is factually relevant and well-formulated, but cites §50 SGB X incorrectly instead of §76 SGB IV, despite the latter being present in the context.	Reveals a failure of norm attribution: the LLM generates legally valid structure but anchors to the wrong legal reference. Highlights the need for stricter citation grounding even when faithfulness is expected to follow from correct retrieval.

Continued on next page

Test Case	Approach	Key Insight	Implication
TC19	**01/03**	Additional reference to 'tax advisors' in the output is supported by the retrieval context	Example of model-driven context enrichment: relevant supplementary information improves specificity without compromising faithfulness
TC34	**01/03**	both approaches achieved an identical **Contextual Relevancy** score of 0.48, despite Approach 03 utilizing a more sophisticated retrieval strategy.	This suggests a limitation in retrieval effectiveness when relevant information is diluted by thematically unrelated content. The advantage of more complex pipelines diminishes if no additional context filtering is applied.
TC18	**All**	Low **Contextual Precision** ($\leq$ 0.50) but high **Answer Relevancy** ($\geq$ 0.80) across all approaches	Demonstrates model robustness: relevant answers can still emerge even when most retrieved chunks are off-topic or poorly ranked. Indicates possible overperformance due to LLM generalization.
TC31	**All**	All contextual metrics failed across all approaches. While the generated answers are relevant, the retrieved context lacks key procedural content (e.g., mediation process, suspension of sanctions).	Highlights a blind spot in the retrieval stack. Calls for improved filtering or retrieval augmentation for procedural legal norms such as §15a SGB II.

Continued on next page

Test Case	Approach	Key Insight	Implication
TC36	**All**	All three approaches achieved an identical **Contextual Recall** score of 0.50, as none were able to retrieve the full scope of information required by the expected output.	This indicates a systematic retrieval gap affecting all methods equally. While core aspects of insurance obligation during work incapacity were covered, details on unexcused absences were entirely missing from the retrieved context. This suggests the need for targeted retrieval augmentation or content-aware expansion to capture procedural nuances.
TC41	**All**	All approaches generate fully relevant answers that contradict or partially misrepresent the context content, particularly regarding whether maintenance payments are treated as income.	Reveals a key limitation of LLMs in legal settings: answers may appear legally plausible while misaligning with retrieved content. Highlights the need for stricter grounding mechanisms beyond relevance scoring. **Faithfulness** degradation can occur even with contextually covered answers.

Additional evaluations of all 50 test cases for each RAG approach are documented in Appendix 7.

Efficiency of Response Time

To evaluate the *Response Time/Latency* criterion introduced in Section 3.1.3, the average response times of each RAG approach were measured. According to Nielsen [Nie94], response times under one second are perceived as instantaneous, while delays beyond ten

seconds risk disrupting the user's cognitive flow and reducing overall interaction quality. Maintaining low latency is therefore essential for usability in real-world applications.

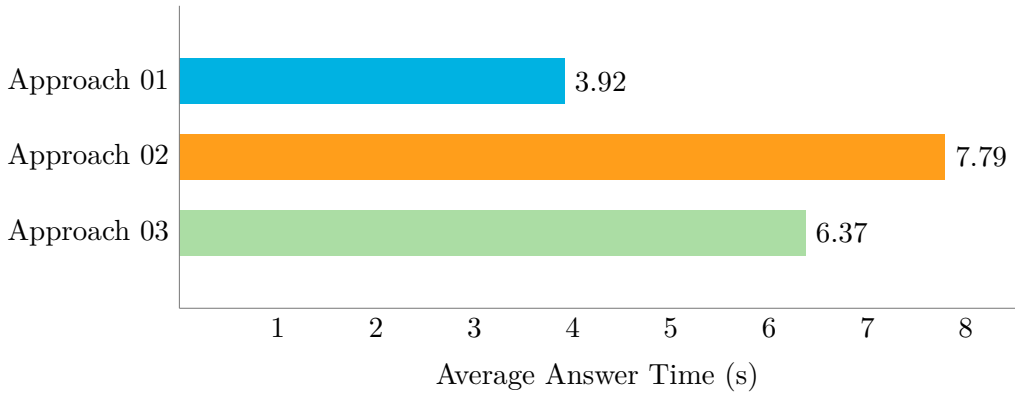


Figure 5.3: Comparison of the average response generation time (in seconds) for each RAG approach. Lower values reflect higher response efficiency.

The results show that **Approach 01** delivers the fastest average response time (3.92 seconds), while **Approach 02** exhibits the slowest (7.79 seconds). **Approach 03** lies between the two, with an average of 6.37 seconds. Although all approaches remain below the 10-second threshold for tolerable delays, only **Approach 01** comes close to the 1-second mark that preserves the impression of immediate system feedback. These differences are particularly relevant in practical applications where responsiveness directly impacts perceived usability, as discussed further in Section 5.4.2.

5.4.2 Interpretation and Discussion

This interpretation is based on the empirical findings presented in Subsection 5.4.1 and supported by selected examples summarized in Table 5.4.

The evaluation reveals that high answer quality (**ET2**) does not necessarily stem from strong retrieval quality (**ET1**). In test cases such as **TC6** and **TC18**, all models generated relevant answers despite weak contextual foundations, indicating a certain generalization capability of the underlying LLM.

Approach 01 – Simple Dense Retrieval

Despite its simple architecture and lower **ET1** performance, **Approach 01** frequently achieves solid **ET2** results. This suggests that the LLM may compensate for missing context with pretrained knowledge. However, the retrieved context often contains incomplete or irrelevant chunks and lacks proper ranking, limiting both **Contextual Precision** and **Contextual Relevancy**.

Approach 02- Hybrid Multi-Query RAG with Ensemble-Reranking

Approach 02 achieves high **Contextual Recall** through its multi-query strategy but suffers significantly from contextual noise. Marginal or redundant chunks reduce **Contextual Relevancy**. The filtering mechanisms (`LLMChainFilter/Extractor`) proved either overly restrictive or ineffective. Similarly, the ensemble reranker fails to reliably prioritize relevant content (e.g., **TC24, TC31**). The **MMR**-based selection (Top-5) offers broader coverage but struggles to balance relevance and information density.

Approach 03 – Parent-Child Hybrid Retrieval with Ensemble-Reranking

Owing to its parent-child structure, **Approach 03** achieves higher average **Contextual Precision** than the other approaches (see Figure 5.1). In **TC2**, a case of semantic redundancy was observed. While it did not significantly affect overall performance, it indicates potential for improvement in deduplication and ranking logic. In **TC39**, **Faithfulness** issues arose despite complete context availability, highlighting shortcomings in the LLM's legal output. Likewise, **TC22** demonstrates that a high **Answer Relevancy** score does not necessarily equate to high **Faithfulness** if the answer includes incorrect or incomplete legal references.

General Observations

More complex retrieval strategies (**Approach 02** and **03**) do not inherently lead to better performance: **Approach 01** outperforms others in terms of **Contextual Relevancy** in some cases. Test cases such as **TC34** and **TC41** emphasize that even advanced pipelines are limited in effectiveness without targeted context filtering. In several instances, a single relevant chunk was sufficient to answer the question, whereas additional passages diluted the evaluation. This suggests that adaptive retrieval may be preferable to a fixed top-K strategy.

Evaluation of Failure Points and Mapping to Required Abilities

As outlined in the evaluation framework (see Section 3.3.6), the effectiveness of the system's retrieval strategies under realistic conditions is assessed by mapping empirically observed **FPs** to the defined *Required Abilities* (**RAs**). While the *Primary Quality Scores* (Section 3.3.2) are operationalized via automatic evaluation metrics (Section 4.2.7), they alone do not sufficiently capture system robustness. The **FPs-RAs** mapping enables a deeper understanding of how well each approach handles challenges such as noisy input, incomplete evidence, or misleading context.

- **RA1** (Noise Robustness) is addressed through **FP4** (Noise Overload) and **FP3** (Irrelevant Context), which evaluates whether models can generate accurate an-

swers despite irrelevant or semantically diluted context. Both **Approaches 02** and **03** were designed to mitigate this via filtering (e.g., duplicate reduction). However their practical implementation proved insufficient. `LLMChainFilter` was overly aggressive, while `LLMChainExtractor` was ineffective (e.g., **TC31**). **Approach 03** also showed localized failures (e.g., **TC2**), indicating unresolved contextual noise (**FP4**). **Approach 02** fails to reliably prioritize relevant information (**FP3**) despite using **MMR** and query reformulation (e.g., **TC24**).

- **RA2 (Negative Rejection)** is reflected in **FP2** (Missed the Top Ranked Documents), where models ideally should abstain from confident output when relevant information is absent. This ability was not demonstrated in **TC6** and **TC42**, where plausible answers were generated despite insufficient contextual evidence.
- **RA4 (Counterfactual Robustness)** is linked to **FP2** (**TC39**), where the model produced a legally plausible but factually incorrect reference despite the correct context being available. **Approach 03**, though generally strong due to its parent-document logic, failed to maintain output fidelity in this instance.

Note: RA3 (Information Integration) could not be evaluated in this thesis, as **FP7** was excluded from the analysis (see Section 5.2.2).

In summary, the **FP**-based analysis confirms that although mitigation strategies for the essential abilities were conceptually well-founded and integrated into the system architectures, their real-world effectiveness remained inconsistent and often insufficient.

Conclusion

Approach 03 produced the highest average scores overall but showed clear limitations in reranking and exhibited a duplicate handling failure in one isolated case. **Approach 02** was most affected by contextual noise and unreliable prioritization of relevant information. Surprisingly, **Approach 01** achieved comparatively stable results and showed superior performance in **Contextual Relevancy** compared to the more complex setups. However, this metric remains limited overall, as only 32% of test cases surpassed the defined threshold (see Figure 5.2).

With regard to efficiency, **Approach 01** is the fastest, which is unsurprising given its simpler architecture. Nonetheless, all models maintained acceptable response times (see Section 5.4.1).

Only a few test cases successfully met the full set of core evaluation metrics defined by the RAG Triad (see Section 3.3.4). While **Approach 03** showed measurable improvements in **Category C1** (**Contextual Precision** and **Contextual Recall**), it remains evident that gains in **ET1** do not automatically translate into improved **ET2** performance.

Relation to SACAM and Metrics

The applied metrics were derived from an adapted SACAM framework, primarily leveraging its structured comparison principles. In contrast to the original scenario-driven method, this study employs a quantitative, metric-based evaluation tailored to the RAG context.

Throughout the analysis, **Contextual Relevancy** and **Faithfulness** occasionally followed divergent patterns, for instance when semantically fitting answers lacked proper legal grounding. This underscores the need for application-specific weighting of quality criteria. The selected set of differentiated metrics (e.g., **Contextual Precision, Contextual Recall, Answer Relevancy**) allows for a comparative and nuanced assessment of all approaches, in line with a pragmatically adapted SACAM methodology.

Relation to Research Questions (RQ3–RQ5)

To conclude the evaluation, the following section explicitly relates the findings to the previously formulated research questions **RQ3** to **RQ5**.

RQ3: The applied metrics **Contextual Precision, Recall, Relevancy, Answer Relevancy** and **Faithfulness** proved suitable for assessing both **ET1** and **ET2**. They support a structured and reliable comparison of system performance and output validity. Their selection was derived from QA-specific evaluation criteria and implemented using the **DeepEval** framework and custom latency tracking. A detailed rationale and configuration can be found in Section 5.2.3.

RQ4: The evaluated RAG approaches differ significantly in performance:

- **Approach 03** achieves the highest average scores but exhibits weaknesses in reranking and duplicate filtering.
- **Approach 02** demonstrates high **Contextual Recall** but is hindered by contextual noise and ineffective filtering.
- **Approach 01**, while architecturally simpler, achieves notably strong **Contextual Relevancy** and in some respects matches the robustness of the two more advanced methods, although it shares similar shortcomings in **Faithfulness** and ranking.

RQ5: **ET2** does not depend strictly on **ET1**. Several cases (e.g., **TC6**, **TC18**) showed relevant answers despite weak context, highlighting the generalization capacity of the LLM. Conversely, cases like **TC39** or **TC22** reveal that even complete context cannot prevent factually incorrect or legally inaccurate answers, emphasizing the need for verification mechanisms and domain-specific grounding.

5.4.3 Threats to Validity

The following limitations outline potential threats to the validity of the conducted evaluation. They are structured according to four commonly accepted dimensions: internal validity, external validity, construct validity, and conclusion validity. The purpose of this section is to reflect critically on the design choices, data foundations, and interpretation of results.

Internal Validity:

(1) Model-Related Distortions

Model Bias and Prior Knowledge: The evaluated responses were generated by RAG systems using **Mistral Large** as the underlying LM. Due to its extensive pretraining, **Mistral** may produce high-quality answers even when the provided context is incomplete or noisy. This can mask weaknesses in retrieval performance and lead to inflated scores, particularly for metrics such as **Faithfulness** and **Contextual Relevancy**, which assume a direct link between context quality and answer accuracy.

Semantic Generalization by the Evaluation Model: For the automated assessment of metrics like **Answer Relevancy**, **Contextual Precision**, and **Faithfulness**, a pre-trained model (**Anthropic Claude 3.5**) was used. Although the evaluation is based on structured inputs such as the question, answer, reference, and retrieved context, the model's semantic generalization capabilities may lead to positive ratings even when the context is insufficient. This reduces the reliability of tracing answer quality back to retrieval effectiveness.

Retrieval–Generation Decoupling: High-quality context retrieval does not always lead to accurate answers, as the LM may misinterpret or misuse correct inputs. This undermines a clear attribution of evaluation scores to the retrieval component. **TC39** exemplifies this, with complete context but legally inaccurate output.

(2) Evaluation Uncertainty and Interpretation

Interpretative Analysis of Evaluation Outputs: All test cases were manually reviewed to assess the plausibility of metric justifications and detect potential inconsistencies, especially in borderline cases. While this supported the interpretation of results, it introduced a degree of subjectivity that may affect internal validity.

Metric Misalignment and Interpretation Risk: Isolated metric scores can occasionally be misleading, especially when surface-level relevance masks factual or legal deficiencies. This underscores the need for holistic interpretation across multiple dimensions rather than relying on a single metric. For instance, in **TC22**, high **Answer Relevancy** coincided with low **Faithfulness** due to incomplete legal references.

(3) System Complexity and Infrastructure Constraints

Varying System Complexity: The compared RAG approaches were intentionally designed with different structural complexities, ranging from a simple dense retrieval setup (**Approach 01**) to more elaborate hybrid and reranking pipelines (**Approach 02** and **03**). While this supports a realistic system-level comparison, the combination of multiple optimization steps complicates attribution of performance effects to specific components, particularly the retriever.

Technical Limitations in the Setup: Certain components (e.g., `LLMChainFilter`, `LLMChainExtractor`) proved ineffective in practice (see Sections 4.3.4, 5.4.2), leading to pragmatic adjustments such as omitting compression in **Approach 03** and excluding **FP7** from the evaluation (see Section 5.2.2). An initially planned approach for synthetic data generation was dropped due to **AWS** integration challenges. Furthermore, using the same model **Anthropic Claude 3.5** for both data generation and evaluation was considered methodologically problematic due to the risk of bias and overfitting. Instead, 50 manually curated test cases were used as a fallback to mitigate such risks (see Section 4.4.1)

External Validity:

Domain-Specific Data: The evaluation corpus is limited to 107 German legal directives (SGB I–III). The results are therefore not necessarily transferable to other domains, languages, or less-structured datasets.

Limited Test Set Size: Only 50 manually curated test cases were used. While high-quality, the dataset lacks statistical representativeness and may omit edge cases or linguistic variation.

Absence of User-Centered Validation: Although response time was measured empirically, no user-centered evaluation was conducted to assess perceived latency or overall user satisfaction. This limits the generalizability of the findings to real-world usage scenarios.

Construct Validity:

Heuristic Thresholds: The pass/fail cutoffs used in this study (e.g., 0.75 for **Answer Relevanc**y) are empirically motivated but ultimately heuristic. In safety-critical or legally sensitive applications, stricter or domain-specific thresholds may be more appropriate.

Conclusion Validity:

Overinterpretation of Individual Metrics: Strong results in individual metrics, such as high **Faithfulness** despite weak context, can create a misleading impression of system performance. For reliable conclusions, it is necessary to interpret the combined metric set holistically, as intended by the **RAG-Triad** approach.

Overgeneralization Risk: Several conclusions (e.g., the surprising performance of **Approach 01**) are based on aggregated averages and a small number of qualitative cases. These insights should be seen as indicative, not definitive, until validated at larger scale.

Overall, while the evaluation methodology is sound and transparent, the outlined factors may affect the reliability, generalizability, or interpretability of the results. These limitations should be considered when transferring the findings to other domains or applications.

6 Conclusion

This chapter summarizes the main findings of the thesis, outlines the resulting contributions, discusses relevant limitations, and identifies directions for future work.

6.1 Summary

This thesis investigated the optimization of RAG systems for domain-specific QA chatbots, with a particular focus on legal documents. Although the legal domain served as the primary context throughout the evaluation, it became evident during the analysis that such use cases require stricter standards regarding contextual accuracy and faithfulness, as discussed in Section 5.4.3. Five research questions were defined in Section 1.2 and examined using a structured approach that integrated targeted literature analysis and comparative system evaluation. The following summarizes the findings with respect to each sub-research question (**RQ2-RQ5**). Together, these insights provide the foundation for answering the main research question (**RQ1**).

Answer to RQ2:
Which RAG approaches with a focus on information preparation are currently known, and which of them are suitable for optimizing RAG chatbots in practical applications?

This research question was addressed through a SLR based on the principles proposed by Kitchenham [Kit07]. While a formal SLR was not fully implemented, the analysis focused on identifying recent retrieval methods and RAG-specific enhancements (see Section 3.2.3). These were categorized both in terms of their applicability within the RAG pipeline and their underlying retrieval model type (e.g., dense, sparse, hybrid). In addition, the approaches were classified according to their retrieval granularity (e.g., chunk-level, token-level, or entity-level) in order to better assess their suitability for domain-specific QA scenarios.

Answer to RQ3:
What evaluation metrics should be in place to assess and compare RAG approaches in terms of their performance and reliability?

The answer to this research question involved a multi-step process, which is described in detail in Section 5.2.3. Based on the derived evaluation criteria and quality dimensions, the following metrics were selected:

- **Contextual Precision**
- **Contextual Recall**
- **Contextual Relevancy**
- **Answer Relevancy**
- **Faithfulness**

To ensure complete coverage of all defined evaluation criteria, including efficiency, a **custom timer** was implemented in each RAG approach to measure *Response Time/Latency.*

Answer to RQ4:
How do the selected RAG approaches perform according to the defined evaluation criteria?

This sub-research question was addressed in the Evaluation chapter (see Chapter 5). The results show that **Approach 03** achieved the highest overall metric scores, particularly in terms of contextual precision and recall. Surprisingly, **Approach 01**, despite its simpler architecture, demonstrated notably strong contextual relevancy.

Answer to RQ5:
What impact do these RAG approaches have on the resulting response quality, particularly regarding accuracy, relevance, and coherence?

This question was also answered in the evaluation (see Section 5.4.2). The findings indicate that **ET2** does not strictly depend on **ET1**. Several test cases yielded relevant answers even with weak or incomplete context, illustrating the generalization capacity of the underlying LM. Conversely, some cases showed that even complete context does not guarantee factually correct or legally accurate responses, which underlines the relevance of verification mechanisms and domain-specific grounding.

Overall, the results emphasize the importance of interpreting evaluation metrics holistically and demonstrate that architectural complexity does not automatically lead to higher output quality.

Answer to the main research question (RQ1)
Which of the existing RAG approaches focused on information preparation hold the greatest potential to optimize the performance of RAG chatbots while enhancing the reliability and quality of their responses in practical applications?

The evaluation shows that **Approach 03** holds the greatest potential for optimizing RAG chatbots, particularly due to its strong retrieval performance in terms of contextual precision and recall. However, its architecture also revealed weaknesses, especially in reranking effectiveness and insufficient duplicate filtering, which may limit its robustness in more complex or noisy retrieval scenarios. Its structured retrieval setup ensures comprehensive and relevant context, which is crucial for reliable response generation.

Nevertheless, the results also highlight that architectural complexity does not automatically guarantee superior output quality. **Approach 01**, despite its simplicity, achieved surprisingly strong contextual relevancy and competitive faithfulness.

Ultimately, answer quality depends not only on retrieval but also on the model's ability to generalize and the domain's specific requirements. Especially in sensitive domains like legal QA, additional verification mechanisms are essential to ensure factual and contextual accuracy.

Although mitigation strategies for known failure points (**FP2**–**FP4**) were conceptually integrated into the system architectures, their practical effectiveness was inconsistent. Components such as reranking, query reformulation, and context filtering did not reliably eliminate irrelevant, noisy or insufficient input. This led to only partial improvements in robustness and output quality (see Section 5.4.2).

These findings must be interpreted in light of the evaluation's limitations (see Section 5.4.3). In particular, the influence of model bias, the small and domain-specific test set, and the use of heuristically defined thresholds may affect the generalizability of the results. While **Approach 03** showed strong technical potential, its practical advantage should be validated further in broader, user-centered or domain-independent settings.

6.2 Contributions

Several conceptual, technical, and evaluative advancements were introduced to improve the design of RAG-based QA chatbot for domain-specific applications. Drawing on literature review, system implementation, and empirical evaluation, the following contributions extend existing research and provide practical guidance for real-world use cases.

Conceptual Framework. A structured taxonomy of RAG applications is developed, including tailored evaluation criteria for QA-specific use cases. Various retrieval types and granularities are identified, accompanied by a comprehensive overview of applicable methods. A core contribution is the design of an evaluation framework that outlines evaluation targets, quality aspects, and required abilities, all aligned with the **RAG Triad**. The framework also clarifies different types of evaluation metrics and their suitability for RAG system assessment.

Practical System Implementation. The conceptual framework was applied to implement multiple RAG approaches, focusing on improving information preparation through pre-, core-, and post-retrieval techniques. Detailed justifications for the chosen technology stack are provided, offering practical guidance. Special emphasis is placed on retrieval strategy selection and the operational integration of evaluation metrics.

Failure Point Mitigation via Required Abilities. Building on established elements like **FPs** and quality metrics, this thesis extends the evaluation methodology by introducing a systematic mapping between **FPs** and **RAs**. This addition supports a more targeted robustness analysis and complements standard metric-based evaluation.

Efficiency-Oriented Evaluation. In addition to output quality, each RAG approach was instrumented with response-time measurement to assess practical usability. The observed average response times reveal trade-offs between retrieval complexity and system responsiveness.

Domain-Specific Observations for Legal QA. Although not the initial design focus, the evaluation surfaced challenges particularly relevant to legal QA scenarios. These include the importance of contextual accuracy, strict verifiability, and constrained model generalization. The findings confirm concerns previously identified in **LexGLUE** benchmarks [Cha21], neural legal judgment prediction studies [Cha19] and recent RAG-specific evaluations such as **LexRAG** [Li25] and demonstrate their practical significance in real-world RAG-based legal applications.

Together, these contributions offer both theoretical and practical value for the design, evaluation, and deployment of robust RAG systems in real-world, domain-sensitive environments.

6.3 Limitations

This section outlines overarching limitations of the thesis that extend beyond the methodological risks discussed in the *Threats to Validity* section (see Section 5.4.3). While the latter focuses on the internal soundness and external generalizability of the

evaluation setup, the following limitations reflect broader conceptual, practical, and domain-specific boundaries of the study.

Limited Focus on Retrieval Quality. The evaluation framework was primarily centered on assessing **ET1**, while **ET2** was only considered indirectly, specifically to observe whether improved retrieval context led to better answers. However, the evaluation revealed that retrieval and generation can behave independently. A more holistic evaluation would require an expanded framework that equally emphasizes both components.

Absence of User-Centered Evaluation. The system was evaluated purely on a technical level without any user feedback or usability testing. Important dimensions such as perceived latency, trust and user satisfaction were not captured, limiting conclusions about practical user experience.

Domain Generalization Limitations. While the system was tested using legal documents, the conceptual framework itself was designed for general QA applications and does not address domain-specific QA needs such as in legal, medical or technical contexts. The evaluation surfaced domain-specific issues (see Section 5.4.2) but the framework does not yet account for them explicitly.

Limited Metric Scope. The evaluation relied on metrics aligned with the **RAG Triad**, which are well-established in practice but cover only basic aspects of system performance. More advanced or task-specific evaluation dimensions (e.g., multi-hop reasoning, legal soundness, user satisfaction) were not included.

Small and Manually Curated Dataset. The evaluation corpus consisted of 107 German legal directives and 50 manually created QA pairs. These questions were derived from the content but were not generated or verified by legal experts. Consequently, the dataset does not support complex or nuanced legal inquiries, limiting the generalizability of findings. As such, the results should be interpreted as indicative rather than definitive.

No Real-World System Testing. The system was not deployed or tested under real-world usage conditions. The evaluation does not simulate realistic query traffic, diverse user behavior, or integration with external information systems.

Dependency on Automated LLM-Based Evaluation. The evaluation was conducted via automated metrics using *LLM-as-a-judge* models, which introduces additional abstraction and possible unreliability (see Section 5.4.3). These models may overgeneralize or tolerate weak context, limiting the accuracy of some judgments.

Infrastructure Constraints. Due to limitations with **AWS** infrastructure, the originally planned synthetic data generation was not implemented. This restricted the diversity of the test cases and reduced the scope of scenario coverage.

Emerging Nature of the Research Field. RAG is an emerging area with rapidly evolving tooling. Many frameworks and best practices are still under development. Consequently, some design decisions were based on currently available tools that may change in future work.

Overall, these limitations do not undermine the validity of the findings but help to contextualize them. The results should be understood as a technically grounded, exploratory study conducted under controlled conditions. They offer valuable guidance for future system design and evaluation, particularly in the area of retrieval quality. However, further research, broader validation and domain-specific adaptation are needed before the approach can be considered fully generalizable.

6.4 Future Work

Although the evaluation framework and comparative system analysis have yielded valuable insights, several open questions and avenues for further research remain. The following directions highlight potential extensions and refinements to the current work.

Domain-Specific Adaptation. Extend the evaluation framework to account for domain-specific requirements, particularly for legal, medical or technical/regulatory QA scenarios. This includes incorporating domain knowledge into the system design and evaluation logic.

Synthetic Dataset Generation. Enable automated generation of synthetic evaluation datasets via **AWS** to support larger and more diverse test collections. This would improve scalability and reduce dependency on manual test case creation.

Custom Metrics for Retrieval Evaluation. Integrate custom evaluation metrics using the **G-Eval** module of the **DeepEval** framework to more precisely assess retrieval behavior and context quality beyond generic metrics.

Improved Reranking and Chunking Strategies. Implement more effective reranking mechanisms and introduce a dynamic chunking procedure to better balance context granularity and relevance in retrieval, with the goal of reducing noise and improving contextual alignment.

Extension to Generation Quality. Expand the evaluation framework to explicitly cover generation quality aspects, such as factuality, completeness and answer formulation to support a comprehensive assessment across both retrieval and generation components.

End-to-End Evaluation Framework. While initial end-to-end evaluation combining retrieval and generation metrics was applied, future work could further develop integrated evaluation frameworks that capture a wider range of real-world tasks, domain-specific constraints, and long-context interactions in RAG systems.

User-Centered Evaluation. Future work should include user studies and UX testing to assess the practical usability and acceptance of the developed RAG approaches. This includes evaluating how end users interact with the system, perceive answer quality, and respond to latency or uncertainty in outputs.

Bibliography

[Abd24] Abdallah, Abdelrahman und Jatowt, Adam: Generator-retriever-generator approach for open-domain question answering. *arXiv preprint arXiv:2307.11278v3* (2024), URL https://doi.org/10.48550/arXiv.2307.11278

[Ada20] Adamopoulou, Eleni und Moussiades, Lefteris: An overview of chatbot technology, in: *Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, Part II 16*, Springer, S. 373–383, URL https://doi.org/10.1007/978-3-030-49186-4_31

[AI] AI, Mistral: Mistral AI, URL https://github.com/mistralai, accessed: 2025-05-25

[Ama09] Amati, Giambattista: *BM25*, Springer US, Boston, MA (2009), S. 257–260, URL https://doi.org/10.1007/978-0-387-39940-9_921

[Ang08] Angles, Renzo und Gutierrez, Claudio: Survey of graph database models. *ACM Computing Surveys (CSUR)* (2008), Bd. 40(1): S. 1–39, URL https://doi.org/10.1145/1322432.1322433

[Ang23] Angelopoulos et al.: Prediction-powered inference. *Science* (2023), Bd. 382(6671): S. 669–674, URL https://doi.org/10.48550/arXiv.2301.09633

[Ari23] Arivazhagan, Manoj Ghuhan; Liu, Lan; Qi, Peng; Chen, Xinchi; Wang, William Yang und Huang, Zhiheng: Hybrid Hierarchical Retrieval for Open-Domain Question Answering, in: Anna Rogers; Jordan Boyd-Graber und Naoaki Okazaki (Herausgeber) *Findings of the Association for Computational Linguistics: ACL 2023*, Association for Computational Linguistics, Toronto, Canada, S. 10680–10689, URL https://aclanthology.org/2023.findings-acl.679/

[Asa23] Asai, Akari; Wu, Zeqiu; Wang, Yizhong; Sil, Avirup und Hajishirzi, Hannaneh: Self-rag: Learning to retrieve, generate, and critique through self-reflection, in: *The Twelfth International Conference on Learning Representations*, URL https://doi.org/10.48550/arXiv.2310.11511

[AWS25a] AWS: Amazon Nova models (2025), URL https://docs.aws.amazon.com/bedrock/latest/userguide/model-parameters-nova.html, accessed: 2025-05-25

[AWS25b] AWS: Carry out a conversation with the Converse API operations (2025), URL https://docs.aws.amazon.com/bedrock/latest/userguide/conversation-inference.html?utm_source=chatgpt.com, accessed: 2025-05-25

[AWS25c] AWS: Evaluate the performance of RAG sources using Amazon Bedrock evaluations (2025), URL https://docs.aws.amazon.com/bedrock/latest/userguide/evaluation-kb.html, accessed: 2025-05-25

[AWS25d] AWS: Mistral AI chat completion (2025), URL https://docs.aws.amazon.com/bedrock/latest/userguide/model-parameters-mistral-chat-completion.html, accessed: 2025-05-25

[AWS25e] AWS: Model support by AWS Region in Amazon Bedrock (2025), URL https://docs.aws.amazon.com/bedrock/latest/userguide/models-regions.html, accessed: 2025-05-25

[AWS25f] AWS: Pixtral Large (25.02) parameters and inference (2025), URL https://docs.aws.amazon.com/bedrock/latest/userguide/model-parameters-mistral-pixtral-large.html, accessed: 2025-05-25

[AWS25g] AWS: Supported models and model features (2025), URL https://docs.aws.amazon.com/bedrock/latest/userguide/conversation-inference-supported-models-features.html, accessed: 2025-05-25

[AWS25h] AWS: Writer Palmyra X5 (2025), URL https://docs.aws.amazon.com/bedrock/latest/userguide/model-parameters-palmyra-x5.html, accessed: 2025-05-25

[Bah14] Bahdanau, Dzmitry; Cho, Kyunghyun und Bengio, Yoshua: Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014), URL https://doi.org/10.48550/arXiv.1409.0473

[Ban05] Banerjee, Satanjeev und Lavie, Alon: METEOR: An automatic metric for MT evaluation with improved correlation with human judgments, in: *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, S. 65–72, URL https://aclanthology.org/W05-0909/

[Bar24] Barnett, Scott; Bae, Sungbin; Wang, Ziyang; Chi, Qinyuan; Sun, Mingyang; Wang, Xingyu; Zhang, Yihong; Xu, Weizhi; Li, Yitong; Zhang, Yueyu; Chen, Xiang und Wen, Jiwei: Seven Failure Points When Engineering a Retrieval Augmented Generation System (2024), URL https://dl.acm.org/doi/10.1145/3644815.3644945

[Ber24] Bertin, Jean: Advancing Similarity Search with GenAI: A Retrieval Augmented Generation Approach. *arXiv preprint arXiv:2501.04006* (2024),

URL https://doi.org/10.48550/arXiv.2501.04006

[Bes24] Besta, Maciej; Kubicek, Ales; Niggli, Roman; Gerstenberger, Robert; Weitzendorf, Lucas; Chi, Mingyuan; Iff, Patrick; Gajda, Joanna; Nyczyk, Piotr; Müller, Jürgen et al.: Multi-head rag: Solving multi-aspect problems with LLMs. *arXiv preprint arXiv:2406.05085* (2024), URL https://doi.org/10.48550/arXiv.2406.05085

[Bom21] Bommasani, Rishi et al.: On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258* (2021), URL https://doi.org/10.48550/arXiv.2108.07258

[Bor22] Borgeaud, Sebastian et al.: Improving Language Models by Retrieving from Trillions of Tokens, in: Kamalika Chaudhuri; Stefanie Jegelka; Le Song; Csaba Szepesvari; Gang Niu und Sivan Sabato (Herausgeber) *Proceedings of the 39th International Conference on Machine Learning*, Bd. 162 von *Proceedings of Machine Learning Research*, PMLR, S. 2206–2240, URL https://proceedings.mlr.press/v162/borgeaud22a.html

[Bro20] Brown, Tom et al.: Language models are few-shot learners. *Advances in neural information processing systems* (2020), Bd. 33: S. 1877–1901

[Bru23] Bruch, Sebastian; Gai, Siyu und Ingber, Amir: An analysis of fusion functions for hybrid retrieval. *ACM Transactions on Information Systems* (2023), Bd. 42(1): S. 1–35, URL https://doi.org/10.1145/3596512

[Bub23] Bubeck, Sébastien et al: Sparks of artificial general intelligence: Early experiments with gpt-4 (2023)

[Bul93] Bulsari, A.: Some analytical solutions to the general approximation problem for feedforward neural networks. *Neural Networks* (1993), Bd. 6(7): S. 991–996, URL https://doi.org/10.1016/S0893-6080(09)80008-7

[Bux21] Buxmann, Peter und Schmidt, Holger (Herausgeber): *Künstliche Intelligenz: Mit Algorithmen zum wirtschaftlichen Erfolg*, Springer Gabler, Berlin, Heidelberg, 2 Aufl. (2021), URL https://doi.org/10.1007/978-3-662-61794-6

[Cao23] Cao, Yihan; Li, Siyu; Liu, Yixin; Yan, Zhiling; Dai, Yutong; Yu, Philip S und Sun, Lichao: A comprehensive survey of ai-generated content (aigc): A history of generative ai from gan to chatgpt. *arXiv preprint arXiv:2303.04226* (2023), URL https://doi.org/10.48550/arXiv.2303.04226

[Car83] Carbonell, Jaime G.; Michalski, Ryszard S. und Mitchell, Tom M.: *An Overview of Machine Learning*, Springer Berlin Heidelberg, Berlin, Heidelberg (1983), S. 3–23, URL https://doi.org/10.1007/978-3-662-12405-5_1

[Cha19] Chalkidis, Ilias; Androutsopoulos, Ion und Aletras, Nikolaos: Neural legal judgment prediction in English. *arXiv preprint arXiv:1906.02059*

(2019), URL https://doi.org/10.48550/arXiv.1906.02059

[Cha21] Chalkidis, Ilias; Jana, Abhik; Hartung, Dirk; Bommarito, Michael; Androutsopoulos, Ion; Katz, Daniel Martin und Aletras, Nikolaos: LexGLUE: A benchmark dataset for legal language understanding in English. *arXiv preprint arXiv:2110.00976* (2021), URL https://doi.org/10.48550/arXiv.2110.00976

[Che24a] Chen, Jiawei; Lin, Hongyu; Han, Xianpei und Sun, Le: Benchmarking large language models in retrieval-augmented generation, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Bd. 38, S. 17754–17762, URL https://doi.org/10.1609/aaai.v38i16.29728

[Che24b] Chen, Jiawei; Lin, Hongyu; Han, Xianpei und Sun, Le: Benchmarking large language models in retrieval-augmented generation, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Bd. 38, S. 17754–17762

[Chi20] Chicco, Davide und Jurman, Giuseppe: The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics* (2020), Bd. 21: S. 1–13, URL https://doi.org/10.1186/s12864-019-6413-7

[Cho14] Cho, Kyunghyun; van Merriënboer, Bart; Gulcehre, Caglar; Bahdanau, Dzmitry; Bougares, Fethi; Schwenk, Holger und Bengio, Yoshua: Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation, in: Alessandro Moschitti; Bo Pang und Walter Daelemans (Herausgeber) *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, S. 1724–1734, URL https://aclanthology.org/D14-1179/

[Cob21] Cobbe, Karl; Kosaraju, Vineet; Bavarian, Mohammad; Chen, Mark; Jun, Heewoo; Kaiser, Lukasz; Plappert, Matthias; Tworek, Jerry; Hilton, Jacob; Nakano, Reiichiro et al.: Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168* (2021), URL https://doi.org/10.48550/arXiv.2110.14168

[Com24] Commision, European: EU Data Act Explained (2024), URL https://digital-strategy.ec.europa.eu/en/factpages/data-act-explained

[Cra09] Craswell, Nick: *Mean Reciprocal Rank*, Springer US, Boston, MA (2009), S. 1703–1703, URL https://doi.org/10.1007/978-0-387-39940-9_488

[Cuc24] Cuconasu, Florin; Trappolini, Giovanni; Siciliano, Federico; Filice, Simone; Campagnano, Cesare; Maarek, Yoelle; Tonellotto, Nicola und Silvestri, Fabrizio: The Power of Noise: Redefining Retrieval for RAG Systems, in: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR

'24, Association for Computing Machinery, New York, NY, USA, S. 719–729, URL https://doi.org/10.1145/3626772.3657834

[Cun08] Cunningham, Pádraig; Cord, Matthieu und Delany, Sarah Jane: *Supervised Learning*, Springer Berlin Heidelberg, Berlin, Heidelberg (2008), S. 21–49, URL https://doi.org/10.1007/978-3-540-75171-7_2

[Dal21] Dale, Robert: GPT-3: What's it good for? *Natural Language Engineering* (2021), Bd. 27(1): S. 113–118, URL https://doi.org/10.1017/S1351324920000601

[Dav15] Davis, Ernest und Marcus, Gary: Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM* (2015), Bd. 58(9): S. 92–103, URL https://doi.org/10.1145/2701413

[Dee25a] DeepEval: Answer Relevancy (2025), URL https://deepeval.com/docs/metrics-answer-relevancy

[Dee25b] DeepEval: Contextual Precision (2025), URL https://deepeval.com/docs/metrics-contextual-precision

[Dee25c] DeepEval: Contextual Recall (2025), URL https://deepeval.com/docs/metrics-contextual-recall

[Dee25d] DeepEval: Contextual Relevancy (2025), URL https://deepeval.com/docs/metrics-contextual-relevancy

[Dee25e] DeepEval: Faithfulness (2025), URL https://deepeval.com/docs/metrics-faithfulness

[Dee25f] DeepEval: G-Eval (2025), URL https://deepeval.com/docs/metrics-llm-evals

[Dee25g] DeepEval: Model support by AWS Region in Amazon Bedrock (2025), URL https://www.deepeval.com/docs/synthesizer-generate-from-docs, accessed: 2025-05-25

[Dee25h] DeepEval: Quick Introduction (2025), URL https://deepeval.com/docs/getting-started

[Dee25i] DeepEval: RAGAS (2025), URL https://deepeval.com/docs/metrics-ragas

[Del24] Delile, Julien; Mukherjee, Srayanta; Van Pamel, Anton und Zhukov, Leonid: Graph-based retriever captures the long tail of biomedical knowledge. *arXiv preprint arXiv:2402.12352* (2024), URL https://doi.org/10.48550/arXiv.2402.12352

[Dev19] Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton und Toutanova, Kristina: Bert: Pre-training of deep bidirectional transformers for language understanding, in: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, S. 4171–4186, URL https://doi.org/10.48550/arXiv.1810.04805

[Din18] Dinan, Emily; Roller, Stephen; Shuster, Kurt; Fan, Angela; Auli, Michael und Weston, Jason: Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241* (2018), URL https://doi.org/10.48550/arXiv.1811.01241

[dJ25] der Justiz, Bundesministerium: Geschäftsgeheimnisgesetz (GeschGehG) (2025), URL https://www.gesetze-im-internet.de/geschgehg/

[Dou24] Douze, Matthijs et al.: The Faiss Library. *arXiv preprint arXiv:2401.08281* (2024), URL https://doi.org/10.48550/arXiv.2401.08281

[Edg24] Edge, Darren; Trinh, Ha; Cheng, Newman; Bradley, Joshua; Chao, Alex; Mody, Apurva; Truitt, Steven; Metropolitansky, Dasha; Ness, Robert Osazuwa und Larson, Jonathan: From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130* (2024), URL https://doi.org/10.48550/arXiv.2404.16130

[Elm10] Elmasri, Ramez und Navathe, Shamkant: *Fundamentals of Database Systems*, Addison-Wesley Publishing Company, USA, 6th Aufl. (2010)

[Es24] Es, Shahul; James, Jithin; Anke, Luis Espinosa und Schockaert, Steven: Ragas: Automated evaluation of retrieval augmented generation, in: *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, S. 150–158, URL https://doi.org/10.48550/arXiv.2309.15217

[Fan19] Fan, Angela; Jernite, Yacine; Perez, Ethan; Grangier, David; Weston, Jason und Auli, Michael: ELI5: Long form question answering. *arXiv preprint arXiv:1907.09190* (2019), URL https://doi.org/10.48550/arXiv.1907.09190

[Fan24] Fan, Wenqi; Ding, Yujuan; Ning, Liangbo; Wang, Shijie; Li, Hengyun; Yin, Dawei; Chua, Tat-Seng und Li, Qing: A survey on rag meeting llms: Towards retrieval-augmented large language models, in: *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, S. 6491–6501, URL https://doi.org/10.48550/arXiv.2405.06211

[Flo20] Floridi, Luciano und Chiriatti, Massimo: GPT-3: Its Nature, Scope, Limits, and Consequences. *Minds and Machines* (2020), Bd. 30: S. 681–694, URL https://link.springer.com/article/10.1007/s11023-020-09548-1

[Fri24] Friel, Robert; Belyi, Masha und Sanyal, Atindriyo: Ragbench: Explainable benchmark for retrieval-augmented generation systems. *arXiv preprint arXiv:2407.11005* (2024)

[fSidIBa] für Sicherheit in der Informationstechnik (BSI), Bundesamt: Generative KI-Modelle Chancen und Risiken für Industrie und Behörden, URL

https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/KI/Generative_KI-Modelle.pdf?__blob=publicationFile&v=7, accessed: 2025-05-25

[fSidIBb] für Sicherheit in der Informationstechnik (BSI), Bundesamt: Guidelines for secure AI system development, URL https://www.ncsc.gov.uk/files/Guidelines-for-secure-AI-system-development.pdf, accessed: 2025-05-25

[Gao23] Gao, Luyu; Ma, Xueguang; Lin, Jimmy und Callan, Jamie: Precise Zero-Shot Dense Retrieval without Relevance Labels, in: Anna Rogers; Jordan Boyd-Graber und Naoaki Okazaki (Herausgeber) *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Toronto, Canada, S. 1762–1777, URL https://aclanthology.org/2023.acl-long.99/

[Gao24] Gao, Yunfan; Deng, Yuxuan; Sun, Kaiwen; Hu, Zhongyu; Shi, Haoyu; Jiang, Peng; Liu, Bing; Wang, Rui; Li, Yufan; Zhan, Xiaohui; Li, Sheng und Yang, Zhilin: Retrieval-Augmented Generation for Large Language Models: A Survey (2024), URL https://arxiv.org/abs/2312.10997

[GB23] Gozalo-Brizuela, Roberto und Garrido-Merchán, Eduardo C: A survey of Generative AI Applications. *arXiv preprint arXiv:2306.02781* (2023), URL https://doi.org/10.48550/arXiv.2306.02781

[Ger22] Gerritse, Emma J; Hasibi, Faegheh und de Vries, Arjen P: Entity-aware transformers for entity search, in: *Proceedings of the 45th international acm sigir conference on research and development in information retrieval*, S. 1455–1465, URL https://doi.org/10.1145/3477495.3531971

[Gol98] Goldstein, Jade und Carbonell, Jaime: Summarization: (1) Using MMR for Diversity- Based Reranking and (2) Evaluating Summaries, in: *TIPSTER TEXT PROGRAM PHASE III: Proceedings of a Workshop held at Baltimore, Maryland, October 13-15, 1998*, Association for Computational Linguistics, Baltimore, Maryland, USA, S. 181–195, URL https://doi.org/10.3115/1119089.1119120

[Gou05] Goutte, Cyril und Gaussier, Eric: A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation, in: David E. Losada und Juan M. Fernández-Luna (Herausgeber) *Advances in Information Retrieval*, Springer Berlin Heidelberg, Berlin, Heidelberg, S. 345–359, URL https://doi.org/10.1007/978-3-540-31865-1_25

[Guu20] Guu, Kelvin; Lee, Kenton; Tung, Zora; Pasupat, Panupong und Chang, Mingwei: Retrieval Augmented Language Model Pre-Training, in: Hal Daumé III und Aarti Singh (Herausgeber) *Proceedings of the 37th International Conference on Machine Learning*, Bd. 119 von *Pro-*

ceedings of Machine Learning Research, PMLR, S. 3929–3938, URL https://proceedings.mlr.press/v119/guu20a.html

[Han23] Han, Yikun; Liu, Chunjiang und Wang, Pengfei: A comprehensive survey on vector database: Storage and retrieval technique, challenge. *arXiv preprint arXiv:2310.11703* (2023), URL https://doi.org/10.48550/arXiv.2310.11703

[He16] He, Ruining und McAuley, Julian: Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering, in: *proceedings of the 25th international conference on world wide web*, S. 507–517, URL https://doi.org/10.48550/arXiv.1602.01585

[He22] He, Wanwei; Dai, Yinpei; Yang, Min; Sun, Jian; Huang, Fei; Si, Luo und Li, Yongbin: Space-3: Unified dialog model pre-training for task-oriented dialog understanding and generation. *arXiv preprint arXiv:2209.06664* (2022), URL https://doi.org/10.48550/arXiv.2209.06664

[He24] He, Xiaoxin; Tian, Yijun; Sun, Yifei; Chawla, Nitesh; Laurent, Thomas; LeCun, Yann; Bresson, Xavier und Hooi, Bryan: G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems* (2024), Bd. 37: S. 132876–132907, URL https://doi.org/10.48550/arXiv.2402.07630

[Hen20] Hendrycks, Dan; Burns, Collin; Basart, Steven; Zou, Andy; Mazeika, Mantas; Song, Dawn und Steinhardt, Jacob: Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300* (2020), URL https://doi.org/10.48550/arXiv.2009.03300

[Hir15] Hirschberg, Julia und Manning, Christopher D: Advances in natural language processing. *Science (New York, N.Y.)* (2015), Bd. 349(6245): S. 261–266, URL https://doi.org/10.1126/science.aaa8685

[Hou24] Hou, Yuki; Tamoto, Haruki und Miyashita, Homei: Integrating Temporal Representations for Dynamic Memory Retrieval and Management in Large Language Models. *arXiv preprint arXiv:2410.13553* (2024), URL https://doi.org/10.48550/arXiv.2410.13553

[Hua22] Huang, Jie und Chang, Kevin Chen-Chuan: Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403* (2022), URL https://doi.org/10.48550/arXiv.2212.10403

[Hus19] Husain, Hamel; Wu, Ho-Hsiang; Gazit, Tiferet; Allamanis, Miltiadis und Brockschmidt, Marc: Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436* (2019), URL https://doi.org/10.48550/arXiv.1909.09436

[Hwa24] Hwang, Taeho; Cho, Sukmin; Jeong, Soyeong; Song, Hoyun; Han, SeungYoon und Park, Jong C: EXIT: Context-Aware Extractive Compression for Enhancing Retrieval-Augmented Generation. *arXiv preprint*

arXiv:2412.12559 (2024), URL https://doi.org/10.48550/arXiv.2412.12559

[Ito25] Ito, Takumi; van Deemter, Kees und Suzuki, Jun: Reference-free Evaluation Metrics for Text Generation: A Survey. *arXiv preprint arXiv:2501.12011* (2025), URL https://doi.org/10.48550/arXiv.2501.12011

[Iza21] Izacard, Gautier; Caron, Mathilde; Hosseini, Lucas; Riedel, Sebastian; Bojanowski, Piotr; Joulin, Armand und Grave, Edouard: Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118* (2021), URL https://doi.org/10.48550/arXiv.2112.09118

[Iza22] Izacard, Gautier und Grave, Edouard: Distilling knowledge from reader to retriever for question answering. *arXiv preprint arXiv:2012.04584v2* (2022), URL https://doi.org/10.48550/arXiv.2012.04584

[Iza23] Izacard, Gautier; Lewis, Patrick; Lomeli, Maria; Hosseini, Lucas; Petroni, Fabio; Schick, Timo; Dwivedi-Yu, Jane; Joulin, Armand; Riedel, Sebastian und Grave, Edouard: Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research* (2023), Bd. 24(251): S. 1–43, URL https://doi.org/10.48550/arXiv.2208.03299

[JÖ2] Järvelin, Kalervo und Kekäläinen, Jaana: Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* (2002), Bd. 20(4): S. 422–446, URL https://doi.org/10.1145/582415.582418

[Jau24] Jauhiainen, Jussi S und Guerra, AgustÃn Garagorry: Evaluating Students' Open-ended Written Responses with LLMs: Using the RAG Framework for GPT-3.5, GPT-4, Claude-3, and Mistral-Large. *arXiv preprint arXiv:2405.05444* (2024), URL https://doi.org/10.48550/arXiv.2405.05444

[Jeo24] Jeong, Soyeong; Baek, Jinheon; Cho, Sukmin; Hwang, Sung Ju und Park, Jong C: Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403* (2024), URL https://doi.org/10.48550/arXiv.2403.14403

[Jia23] Jiang, Albert Q. et al.: Mistral 7B (2023), URL https://doi.org/10.48550/arXiv.2310.06825

[Jos24] Joshi, Pankaj; Gupta, Aditya; Kumar, Pankaj und Sisodia, Manas: Robust multi model rag pipeline for documents containing text, table & images, in: *2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, IEEE, S. 993–999, URL https://doi.org/10.1109/ICAAIC60222.2024.10574972

[Kar20] Karpukhin, Vladimir; Oguz, Barlas; Min, Sewon; Lewis, Patrick SH; Wu, Ledell; Edunov, Sergey; Chen, Danqi und Yih, Wen-tau: Dense Passage Retrieval for Open-Domain Question Answering., in: *EMNLP (1)*, S. 6769–6781, URL https://doi.org/10.48550/arXiv.2004.04906

[Kat25] Katsis, Yannis et al.: MTRAG: A Multi-Turn Conversational Benchmark for Evaluating Retrieval-Augmented Generation Systems. *arXiv preprint arXiv:2501.03468* (2025)

[Kim24] Kim, Kiseung und Lee, Jay-Yoon: Re-rag: Improving open-domain qa performance and interpretability with relevance estimator in retrieval-augmented generation. *arXiv preprint arXiv:2406.05794* (2024), URL https://doi.org/10.48550/arXiv.2406.05794

[Kit07] Kitchenham, Barbara und Charters, Stuart M.: Guidelines for performing Systematic Literature Reviews in Software Engineering (2007), URL https://www.researchgate.net/publication/302924724_Guidelines_for_performing_Systematic_Literature_Reviews_in_Software_Engineering

[Koe05] Koehn, Philipp: Europarl: A Parallel Corpus for Statistical Machine Translation, in: *Proceedings of Machine Translation Summit X: Papers*, Phuket, Thailand, S. 79–86, URL https://aclanthology.org/2005.mtsummit-papers.11/

[Koh20] Kohne, Andreas; Gohring, Tobias; Heumann, Christoph und Wurster, Stefan: *Chatbots: Aufbau und Anwendungsmöglichkeiten von autonomen Sprachassistenten*, Springer Vieweg (2020), URL https://link.springer.com/content/pdf/10.1007/978-3-658-28849-5.pdf

[Kos24] Kostric, Ivica und Balog, Krisztian: A surprisingly simple yet effective multi-query rewriting method for conversational passage retrieval, in: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 2271–2275, URL https://doi.org/10.1145/3626772.3657933

[Kot20] Kotonya, Neema und Toni, Francesca: Explainable automated fact-checking for public health claims. *arXiv preprint arXiv:2010.09926* (2020), URL https://doi.org/10.48550/arXiv.2010.09926

[Lan] Langchain: Build a Retrieval Augmented Generation (RAG) App: Part 1, URL https://js.langchain.com/docs/tutorials/rag/, accessed: 2024-11-25

[Lan24] LangChain: LangChain Documentation: How to Reorder Documents in Long Contexts (2024), URL https://python.langchain.com/docs/how_to/long_context_reorder/, accessed: 2025-05-05

[Lan25a] Langchain: Evaluation concepts (2025), URL https://docs.smith.langchain.com/evaluation/concepts#evaluators

[Lan25b] LangChain: How to combine results from multiple retrievers (2025), URL https://python.langchain.com/docs/how_to/ensemble_retriever/

[Lan25c] LangChain: How to do retrieval with contextual compression (2025), URL https://python.langchain.com/docs/how_to/contextual_compression/

[Lan25d] LangChain: How to use the MultiQueryRetriever (2025), URL https://python.langchain.com/docs/how_to/MultiQueryRetriever/

[Lan25e] LangChain: How to use the Parent Document Retriever (2025), URL https://python.langchain.com/docs/how_to/parent_document_retriever/

[Leb16] Lebret, Rémi; Grangier, David und Auli, Michael: Neural text generation from structured data with application to the biography domain. *arXiv preprint arXiv:1603.07771* (2016), URL https://doi.org/10.48550/arXiv.1603.07771

[Len24] Leng, Quinn; Portes, Jacob; Havens, Sam; Zaharia, Matei und Carbin, Michael: Long context rag performance of large language models. *arXiv preprint arXiv:2411.03538* (2024), URL https://doi.org/10.48550/arXiv.2411.03538

[Lew20] Lewis, Patrick; Perez, Ethan; Piktus, Aleksandra; Petroni, Fabio; Karpukhin, Vladimir; Goyal, Naman; Küttler, Heinrich; Lewis, Mike; Yih, Wen-tau; Rocktäschel, Tim et al.: Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems* (2020), Bd. 33: S. 9459–9474, URL https://doi.org/10.48550/arXiv.2005.11401

[Lew21] Lewis, Patrick; Perez, Ethan; Piktus, Aleksandra; Petroni, Fabio; Karpukhin, Vladimir; Goyal, Naman; Küttler, Heinrich; Lewis, Mike; Yih, Wen-tau; Rocktäschel, Tim; Riedel, Sebastian und Zettlemoyer, Luke: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks (2021), URL https://arxiv.org/abs/2005.11401

[Li23] Li, Jinyang; Hui, Binyuan; Qu, Ge; Yang, Jiaxi; Li, Binhua; Li, Bowen; Wang, Bailin; Qin, Bowen; Geng, Ruiying; Huo, Nan et al.: Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems* (2023), Bd. 36: S. 42330–42357

[Li24] Li, Zhicong et al.: DMQR-RAG: Diverse Multi-Query Rewriting for RAG. *arXiv preprint arXiv:2411.13154* (2024), URL https://doi.org/10.48550/arXiv.2411.13154

[Li25] Li, Haitao et al.: LexRAG: Benchmarking Retrieval-Augmented Generation in Multi-Turn Legal Consultation Conversation. *arXiv preprint arXiv:2502.20640* (2025), URL https://doi.org/10.48550/arXiv.

2502.20640

[Lin04] Lin, Chin-Yew: Rouge: A package for automatic evaluation of summaries, in: *Text summarization branches out*, Association for Computational Linguistics, Barcelona, Spain, S. 74–81, URL https://aclanthology.org/W04-1013/

[Lin19] Lin, Zhaojiang; Madotto, Andrea; Wu, Chien-Sheng und Fung, Pascale: Personalizing dialogue agents via meta-learning. *arXiv preprint arXiv:1905.10033* (2019), URL https://doi.org/10.48550/arXiv.1905.10033

[Lin22] Lin, Jimmy; Nogueira, Rodrigo und Yates, Andrew: *Pretrained transformers for text ranking: Bert and beyond*, Springer Nature (2022)

[Liu02] Liu, Xiaoyong und Croft, W Bruce: Passage retrieval based on language models, in: *Proceedings of the eleventh international conference on Information and knowledge management*, S. 375–382, URL https://doi.org/10.1145/584792.584854

[Liu21] Liu, Ye; Hashimoto, Kazuma; Zhou, Yingbo; Yavuz, Semih; Xiong, Caiming und Yu, Philip S: Dense hierarchical retrieval for open-domain question answering. *arXiv preprint arXiv:2110.15439* (2021), URL https://doi.org/10.48550/arXiv.2110.15439

[Liu23a] Liu, Nelson F; Lin, Kevin; Hewitt, John; Paranjape, Ashwin; Bevilacqua, Michele; Petroni, Fabio und Liang, Percy: Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics* (2023), Bd. 12: S. 157–173, URL https://doi.org/10.48550/arXiv.2307.03172

[Liu23b] Liu, Yi; Huang, Lianzhe; Li, Shicheng; Chen, Sishuo; Zhou, Hao; Meng, Fandong; Zhou, Jie und Sun, Xu: Recall: A benchmark for llms robustness against external counterfactual knowledge. *arXiv preprint arXiv:2311.08147* (2023)

[Lla24] LlamaIndex: LlamaIndex Documentation: Long Context Reordering (2024), URL https://docs.llamaindex.ai/en/stable/examples/node_postprocessor/LongContextReorder/, accessed: 2025-05-05

[Lua21] Luan, Yi; Eisenstein, Jacob; Toutanova, Kristina und Collins, Michael: Sparse, Dense, and Attentional Representations for Text Retrieval. *Transactions of the Association for Computational Linguistics* (2021), Bd. 9: S. 329–345

[luJI25] last updated Jeffrey Ip: Using the RAG Triad for RAG evaluation (2025), URL https://www.deepeval.com/guides/guides-rag-triad, accessed: 2025-04-14

[Lyu25] Lyu, Yuanjie; Li, Zhiyu; Niu, Simin; Xiong, Feiyu; Tang, Bo; Wang, Wenjin; Wu, Hao; Liu, Huanyong; Xu, Tong und Chen, Enhong: Crud-rag: A

comprehensive chinese benchmark for retrieval-augmented generation of large language models. *ACM Transactions on Information Systems* (2025), Bd. 43(2): S. 1–32

[Mav24] Mavi, Vaibhav; Jangra, Anubhav; Jatowt, Adam et al.: Multi-hop question answering. *Foundations and Trends® in Information Retrieval* (2024), Bd. 17(5): S. 457–586, URL http://dx.doi.org/10.1561/1500000102

[Mie21] Mielke, Sabrina J; Alyafeai, Zaid; Salesky, Elizabeth; Raffel, Colin; Dey, Manan; Gallé, Matthias; Raja, Arun; Si, Chenglei; Lee, Wilson Y; Sagot, Benoît et al.: Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP (2021), URL https://doi.org/10.48550/arXiv.2112.10508

[Min19] Min, Sewon; Wallace, Eric; Singh, Sameer; Gardner, Matt; Hajishirzi, Hannaneh und Zettlemoyer, Luke: Compositional questions do not necessitate multi-hop reasoning. *arXiv preprint arXiv:1906.02900* (2019), URL https://doi.org/10.48550/arXiv.1906.02900

[Min21] Min, Sewon; Lee, Kenton; Chang, Ming-Wei; Toutanova, Kristina und Hajishirzi, Hannaneh: Joint passage ranking for diverse multi-answer retrieval. *arXiv preprint arXiv:2104.08445* (2021), URL https://doi.org/10.48550/arXiv.2104.08445

[Mis24] Mishra, Pradumn; Mahakali, Aditya und Venkataraman, Prasanna Shrinivas: SEARCHD – Advanced Retrieval with Text Generation using Large Language Models and Cross Encoding Re-ranking, in: *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, S. 975–980, URL https://doi.org/10.1109/CASE59546.2024.10711642

[Mur12] Murphy, Kevin P.: *Machine Learning: A Probabilistic Perspective*, The MIT Press (2012), URL https://dl.acm.org/doi/abs/10.5555/2380985

[Nar18] Narayan, Shashi; Cohen, Shay B und Lapata, Mirella: Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745* (2018), URL https://doi.org/10.48550/arXiv.1808.08745

[Nas17] Nasteski, Vladimir: An overview of the supervised machine learning methods. *HORIZONS.B* (2017), Bd. 4: S. 51–62, URL https://doi.org/10.20544/HORIZONS.B.04.1.17.P05

[Nie94] Nielsen, Jakob: *Usability Engineering*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1994), URL https://dl.acm.org/doi/book/10.5555/2821575

[Niu23] Niu, Cheng et al.: Ragtruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. *arXiv preprint arXiv:2401.00396*

(2023)

[Nog19] Nogueira, Rodrigo und Cho, Kyunghyun: Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085* (2019), URL https://doi.org/10.48550/arXiv.1901.04085

[Ope22] OpenAI: Introducing ChatGPT (2022), URL https://openai.com/index/chatgpt/, accessed: 2024-11-25

[Pap02] Papineni, Kishore; Roukos, Salim; Ward, Todd und Zhu, Wei-Jing: Bleu: a method for automatic evaluation of machine translation, in: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, S. 311–318, URL https://doi.org/10.3115/1073083.1073135

[Par] Parliament, The European: REGULATION (EU) 2024/1689 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL, URL https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32024R1689, accessed: 2025-05-25

[Par16] Parliament, The European: General Data Protection Regulation (EU) 2016/679 (2016), URL https://eur-lex.europa.eu/eli/reg/2016/679/oj

[Pat23] Patil, Rajvardhan; Boit, Sorio; Gudivada, Venkat und Nandigam, Jagadeesh: A Survey of Text Representation and Embedding Techniques in NLP. *IEEE Access* (2023), Bd. 11: S. 36120–36146, URL https://doi.org/10.1109/ACCESS.2023.3266377

[pc25] pgvector contributors: pgvector: Open-source vector similarity search for PostgreSQL (2025), URL https://github.com/pgvector/pgvector, accessed: 2025-05-29

[Pen07] Pennachin, Cassio und Goertzel, Ben: *Contemporary Approaches to Artificial General Intelligence*, Springer Berlin Heidelberg, Berlin, Heidelberg (2007), S. 1–30, URL https://doi.org/10.1007/978-3-540-68677-4_1

[Pen24] Peng, Boci; Zhu, Yun; Liu, Yongchao; Bo, Xiaohe; Shi, Haizhou; Hong, Chuntao; Zhang, Yan und Tang, Siliang: Graph retrieval-augmented generation: A survey. *arXiv preprint arXiv:2408.08921* (2024), URL https://doi.org/10.48550/arXiv.2408.08921

[Pru24] Prudhvith, Tavva; Swattik, Chakrabarty und Prakash, Selvakumar: Enhancing Retrieval Augmented Generation Systems with Knowledge Graphs, in: *2024 International Conference on Electrical, Computer and Energy Technologies (ICECET*, IEEE, S. 1–8, URL https://doi.org/10.1109/ICECET61485.2024.10698122

[Pul51] Pulgram, Ernst: Phoneme and grapheme: A parallel. *Word* (1951), Bd. 7(1): S. 15–20, URL https://doi.org/10.1080/00437956.1951.11659389

[Qia25] Qian, Hongjin; Liu, Zheng; Zhang, Peitian; Mao, Kelong; Lian, Defu; Dou, Zhicheng und Huang, Tiejun: MemoRAG: Boosting Long Context Processing with Global Memory-Enhanced Retrieval Augmentation, in: *Proceedings of the ACM on Web Conference 2025*, S. 2366–2377, URL https://doi.org/10.1145/3696410.3714805

[Qu21] Qu, Chen; Zamani, Hamed; Yang, Liu; Croft, W Bruce und Learned-Miller, Erik: Passage retrieval for outside-knowledge visual question answering, in: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 1753–1757, URL https://doi.org/10.1145/3404835.3462987

[Rac24] Rackauckas, Zackary: Rag-fusion: a new take on retrieval-augmented generation. *arXiv preprint arXiv:2402.03367* (2024), URL https://doi.org/10.48550/arXiv.2402.03367

[Raf20] Raffel, Colin; Shazeer, Noam; Roberts, Adam; Lee, Katherine; Narang, Sharan; Matena, Michael; Zhou, Yanqi; Li, Wei und Liu, Peter J: Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research* (2020), Bd. 21(140): S. 1–67

[Rag25a] Ragas Team: Context Entities Recall (2025), URL https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/context_entities_recall/, accessed: 2025-05-19

[Rag25b] Ragas Team: Context Precision (2025), URL https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/context_precision/, accessed: 2025-05-19

[Rag25c] Ragas Team: Context Recall (2025), URL https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/context_recall/, accessed: 2025-05-19

[Rag25d] Ragas Team: Faithfulness (2025), URL https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/faithfulness/, accessed: 2025-05-19

[Rag25e] Ragas Team: LangChain Integration (2025), URL https://docs.ragas.io/en/stable/howtos/integrations/langchain/, accessed: 2025-05-19

[Rag25f] Ragas Team: Metrics - FactualCorrectness (2025), URL https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/factual_correctness/#factual-correctness, accessed: 2025-05-19

[Rag25g] Ragas Team: Metrics - LLMContextRecall (2025), URL https://docs.ragas.io/en/stable/references/metrics/?h=llmcontextrecall#ragas.metrics.LLMContextRecall, accessed: 2025-05-19

[Rag25h] Ragas Team: Noise Sensitivity (2025), URL https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/noise_sensitivity/, accessed: 2025-05-19

[Rag25i] Ragas Team: Overview of Metrics (2025), URL https://docs.ragas.io/en/stable/concepts/metrics/overview/, accessed: 2025-05-19

[Rag25j] Ragas Team: Response Relevancy (2025), URL https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/answer_relevance/, accessed: 2025-05-19

[Raj16] Rajpurkar, Pranav; Zhang, Jian; Lopyrev, Konstantin und Liang, Percy: Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* (2016), URL https://doi.org/10.48550/arXiv.1606.05250

[Ram03] Ramos, Juan et al.: Using tf-idf to determine word relevance in document queries, in: *Proceedings of the first instructional conference on machine learning*, Bd. 242, Citeseer, S. 29–48

[Ras20] Raschka, Sebastian; Patterson, Joshua und Nolet, Corey: Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *Information* (2020), Bd. 11(4): S. 193, URL https://doi.org/10.48550/arXiv.2002.04803

[Reh24] Rehm, Georg; Dietze, Stefan; Schimmler, Sonja und Krüger, Frank (Herausgeber): *Natural Scientific Language Processing and Research Knowledge Graphs*, Bd. LNCS von *Lecture Notes in Computer Science*, Springer Cham (2024), URL https://doi.org/10.1007/978-3-031-65794-8

[Rob95] Robertson, Stephen E; Walker, Steve; Jones, Susan; Hancock-Beaulieu, Micheline M; Gatford, Mike et al.: Okapi at TREC-3. *Nist Special Publication Sp* (1995), Bd. 109: S. 109, URL https://www.researchgate.net/publication/221037764_Okapi_at_TREC-3

[Rob09] Robertson, Stephen und Zaragoza, Hugo: The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends® in Information Retrieval* (2009), Bd. 3(4): S. 333–389, URL http://dx.doi.org/10.1561/1500000019

[Roy24] Roychowdhury, Sujoy; Smith, John; Chen, Li und Müller, Anna: Evaluation of RAG Metrics for Question Answering in the Telecom Domain (2024), URL https://doi.org/10.48550/arXiv.2407.12873

[Saj18] Sajjadi, Mehdi S. M.; Bachem, Olivier; Lucic, Mario; Bousquet, Olivier und Gelly, Sylvain: Assessing Generative Models via Precision and Recall, in: S. Bengio; H. Wallach; H. Larochelle; K. Grauman; N. Cesa-Bianchi und R. Garnett (Herausgeber) *Advances in Neu-*

ral Information Processing Systems, Bd. 31, Curran Associates, Inc., URL https://proceedings.neurips.cc/paper_files/paper/2018/file/f7696a9b362ac5a51c3dc8f098b73923-Paper.pdf

[Sar24] Sarthi, Parth; Abdullah, Salman; Tuli, Aditi; Khanna, Shubh; Goldie, Anna und Manning, Christopher D.: Raptor: Recursive Abstractive Processing for Tree-Organized Retrieval. *arXiv preprint arXiv:2401.18059* (2024), URL https://doi.org/10.48550/arXiv.2401.18059

[Saw24] Sawarkar, Kunal; Mangal, Abhilasha und Solanki, Shivam Raj: Blended rag: Improving rag (retriever-augmented generation) accuracy with semantic search and hybrid query-based retrievers, in: *2024 IEEE 7th International Conference on Multimedia Information Processing and Retrieval (MIPR)*, IEEE, S. 155–161, URL https://doi.org/10.1109/MIPR62202.2024.00031

[Sch22] Schick, Timo; Dwivedi-Yu, Jane; Jiang, Zhengbao; Petroni, Fabio; Lewis, Patrick; Izacard, Gautier; You, Qingfei; Nalmpantis, Christoforos; Grave, Edouard und Riedel, Sebastian: Peer: A collaborative language model. *arXiv preprint arXiv:2208.11663* (2022), URL https://doi.org/10.48550/arXiv.2208.11663

[Sea80] Searle, John R.: Minds, brains, and programs. *Behavioral and Brain Sciences* (1980), Bd. 3(3): S. 417–424, URL https://doi.org/10.1017/S0140525X00005756

[Sen16] Sennrich, Rico; Haddow, Barry und Birch, Alexandra: Neural Machine Translation of Rare Words with Subword Units, in: Katrin Erk und Noah A. Smith (Herausgeber) *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Berlin, Germany, S. 1715–1725, URL https://aclanthology.org/P16-1162/

[Ser25] Services, Amazon Web: Invoke Meta Llama on Amazon Bedrock using Bedrock's Converse API, https://docs.aws.amazon.com/bedrock/latest/userguide/bedrock-runtime_example_bedrock-runtime_Converse_MetaLlama_section.html (2025), accessed: 2025-05-25

[SF23] Saad-Falcon, Jon; Khattab, Omar; Potts, Christopher und Zaharia, Matei: Ares: An automated evaluation framework for retrieval-augmented generation systems. *arXiv preprint arXiv:2311.09476* (2023), URL https://doi.org/10.48550/arXiv.2311.09476

[Sha20] Shah, Krunal; Gupta, Nitish und Roth, Dan: What do we expect from Multiple-choice QA Systems? *arXiv preprint arXiv:2011.10647* (2020), URL https://doi.org/10.48550/arXiv.2011.10647

[Sid21] Sidiropoulos, Georgios; Voskarides, Nikos; Vakulenko, Svitlana und Kanoulas, Evangelos: Combining lexical and dense retrieval for

computationally efficient multi-hop question answering. *arXiv preprint arXiv:2106.08433* (2021), URL https://doi.org/10.48550/arXiv.2106.08433

[Sin01] Singhal, Amit et al.: Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.* (2001), Bd. 24(4): S. 35–43

[Sin18] Singh, Sonit: Natural language processing for information extraction. *arXiv preprint arXiv:1807.02383* (2018), URL https://doi.org/10.48550/arXiv.1807.02383

[Siv24] Sivasothy, Shangeetha; Barnett, Scott; Kurniawan, Stefanus; Rasool, Zafaryab und Vasa, Rajesh: RAGProbe: An Automated Approach for Evaluating RAG Applications. *arXiv preprint arXiv:2409.19019* (2024), URL https://doi.org/10.48550/arXiv.2409.19019

[SJ72] Sparck Jones, Karen: A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* (1972), Bd. 28(1): S. 11–21, URL https://doi.org/10.1108/EB02652

[SMB09] STEVEN M. BEITZEL, ERIC C. JENSEN und FRIEDER, OPHIR: *Mean Average Precision*, Springer US, Boston, MA (2009), S. 1691–1692, URL https://doi.org/10.1007/978-0-387-39940-9_3032

[Sto03] Stoermer, Christoph; Bachmann, Felix und Verhoef, Chris: SACAM: The Software Architecture Comparison Analysis Method, Techn. Ber., Carnegie Mellon University, Software Engineering Institute (2003), URL https://kilthub.cmu.edu/articles/report/SACAM_The_Software_Architecture_Comparison_Analysis_Method/6583532

[Stu24] Stureborg, Rickard; Alikaniotis, Dimitris und Suhara, Yoshi: Large language models are inconsistent and biased evaluators. *arXiv preprint arXiv:2405.01724* (2024), URL https://doi.org/10.48550/arXiv.2405.01724

[Sut14] Sutskever, Ilya; Vinyals, Oriol und Le, Quoc V: Sequence to sequence learning with neural networks. *Advances in neural information processing systems* (2014), Bd. 27, URL https://doi.org/10.48550/arXiv.1409.3215

[Tab16] Taboada, Maite: Sentiment analysis: An overview from linguistics. *Annual Review of Linguistics* (2016), Bd. 2(1): S. 325–347, URL https://doi.org/10.1146/annurev-linguistics-011415-040518

[Tae23] Taecharungroj, Viriya: "What Can ChatGPT Do?" Analyzing Early Reactions to the Innovative AI Chatbot on Twitter (2023), URL https://doi.org/10.3390/bdcc7010035

[Tan24a] Tang, Qiaoyu; Chen, Jiawei; Yu, Bowen; Lu, Yaojie; Fu, Cheng; Yu, Haiyang; Lin, Hongyu; Huang, Fei; He, Ben; Han, Xianpei et al.: Self-retrieval: Building an information retrieval system with one large language model. *arXiv e-prints* (2024): S. arXiv–2403, URL https://

doi.org/10.48550/arXiv.2403.00801

[Tan24b] Tang, Yixuan und Yang, Yi: Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *arXiv preprint arXiv:2401.15391* (2024), URL https://doi.org/10.48550/arXiv.2401.15391

[Tea24] Team, Anthropic: Introducing computer use, a new Claude 3.5 Sonnet, and Claude 3.5 Haiku (2024), URL https://www.anthropic.com/news/3-5-models-and-computer-use

[Tea25a] Team, Chroma: Chroma (2025), URL https://docs.trychroma.com/docs/overview/introduction

[Tea25b] Team, Langchain: Chroma (2025), URL https://python.langchain.com/docs/integrations/vectorstores/chroma/

[Tea25c] Team, Langchain: Faiss (2025), URL https://python.langchain.com/docs/integrations/vectorstores/faiss/

[Tea25d] Team, LangChain: LangChain Documentation (2025), URL https://python.langchain.com/docs/introduction/, accessed: 2025-05-25

[Tea25e] Team, Langchain: Milvus (2025), URL https://python.langchain.com/docs/integrations/vectorstores/milvus/

[Tea25f] Team, Langchain: PGVector (2025), URL https://python.langchain.com/docs/integrations/vectorstores/pgvector/

[Tea25g] Team, Langchain: Qdrant (2025), URL https://python.langchain.com/docs/integrations/vectorstores/qdrant/

[Tea25h] Team, Langchain: Weaviate (2025), URL https://python.langchain.com/docs/integrations/vectorstores/weaviate/

[Tea25i] Team, LlamaIndex: LlamaIndex Documentation (2025), URL https://docs.llamaindex.ai/en/stable/, accessed: 2025-05-25

[Tea25j] Team, Meta: Welcome to Faiss Documentation (2025), URL https://faiss.ai/index.html

[Tea25k] Team, Milvus: What is Milvus? (2025), URL https://milvus.io/docs/overview.md

[Tea25l] Team, Qdrant: Qdrant Documentation (2025), URL https://qdrant.tech/documentation/

[Tea25m] Team, Qdrant: Qdrant helm repository (2025), URL https://github.com/qdrant/qdrant-helm

[Tea25n] Team, Weaviate: Integrations (2025), URL https://weaviate.io/developers/integrations

[Tea25o] Team, Weaviate: The AI-Native, Open Source Vector Database (2025), URL https://weaviate.io/platform

[Tha23] Thakur, Nandan; Bonifacio, Luiz; Zhang, Xinyu; Ogundepo, Odunayo; Kamalloo, Ehsan; Alfonso-Hermelo, David; Li, Xiaoguang; Liu, Qun;

Chen, Boxing; Rezagholizadeh, Mehdi et al.: NoMIRACL: Knowing When You Don't Know for Robust Multilingual Retrieval-Augmented Generation. *arXiv preprint arXiv:2312.11361* (2023), URL https://doi.org/10.48550/arXiv.2312.11361

[Tra24] Tran, The Trung; González-Gallardo, Carlos-Emiliano und Doucet, Antoine: Retrieval Augmented Generation for Historical Newspapers, in: *Proceedings of the 24th ACM/IEEE Joint Conference on Digital Libraries*, S. 1–5, URL https://doi.org/10.1145/3677389.3702542

[Tru25a] Trulens: Getting Started (2025), URL https://www.trulens.org/getting_started/, accessed: 2025-05-31

[Tru25b] Trulens: The RAG Triad (2025), URL https://www.trulens.org/getting_started/core_concepts/rag_triad/, accessed: 2025-04-14

[Tur23] Turpin, Miles; Michael, Julian; Perez, Ethan und Bowman, Samuel: Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems* (2023), Bd. 36: S. 74952–74965, URL https://doi.org/10.48550/arXiv.2305.04388

[Val09] Valizadegan, Hamed; Jin, Rong; Zhang, Ruofei und Mao, Jianchang: Learning to Rank by Optimizing NDCG Measure, in: Y. Bengio; D. Schuurmans; J. Lafferty; C. Williams und A. Culotta (Herausgeber) *Advances in Neural Information Processing Systems*, Bd. 22, Curran Associates, Inc., URL https://proceedings.neurips.cc/paper_files/paper/2009/file/b3967a0e938dc2a6340e258630febd5a-Paper.pdf

[Vas17] Vaswani, Ashish; Shazeer, Noam; Parmar, Niki; Uszkoreit, Jakob; Jones, Llion; Gomez, Aidan N; Kaiser, Lukasz und Polosukhin, Illia: Attention is all you need (2017), URL https://doi.org/10.48550/arXiv.1706.03762

[Ver24] Verma, Sourav: Contextual Compression in Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv preprint arXiv:2409.13385* (2024), URL https://doi.org/10.48550/arXiv.2409.13385

[Wan13] Wang, Yining; Wang, Liwei; Li, Yuanzhi; He, Di und Liu, Tie-Yan: A theoretical analysis of NDCG type ranking measures, in: *Conference on learning theory*, PMLR, S. 25–54, URL https://doi.org/10.48550/arXiv.1304.6480

[Wan22] Wang, Siyuan; Liu, Zhongkun; Zhong, Wanjun; Zhou, Ming; Wei, Zhongyu; Chen, Zhumin und Duan, Nan: From LSAT: The Progress and Challenges of Complex Reasoning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* (2022), Bd. 30: S. 2201–2216, URL https://doi.org/10.1109/TASLP.2022.3164218

[Wan23a] Wang, Boxin; Ping, Wei; Xu, Peng; McAfee, Lawrence; Liu, Zihan; Shoeybi, Mohammad; Dong, Yi; Kuchaiev, Oleksii; Li, Bo; Xiao, Chaowei et al.: Shall we pretrain autoregressive language models with retrieval? a comprehensive study. *arXiv preprint arXiv:2304.06762* (2023), URL https://doi.org/10.48550/arXiv.2304.06762

[Wan23b] Wang, Hongru; Hu, Minda; Deng, Yang; Wang, Rui; Mi, Fei; Wang, Weichao; Wang, Yasheng; Kwan, Wai-Chung; King, Irwin und Wong, Kam-Fai: Large language models as source planner for personalized knowledge-grounded dialogue. *arXiv preprint arXiv:2310.08840* (2023), URL https://doi.org/10.48550/arXiv.2310.08840

[Wan23c] Wang, Peiyi; Li, Lei; Chen, Liang; Cai, Zefan; Zhu, Dawei; Lin, Binghuai; Cao, Yunbo; Liu, Qi; Liu, Tianyu und Sui, Zhifang: Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926* (2023), URL https://doi.org/10.48550/arXiv.2305.17926

[Wen17] Wen, Zeyi; Deng, Dong; Zhang, Rui und Ramamohanarao, Kotagiri: A technical report: entity extraction using both character-based and token-based similarity. *arXiv preprint arXiv:1702.03519* (2017), URL https://doi.org/10.48550/arXiv.1702.03519

[Wu18] Wu, Yu-chen und Feng, Jun-wen: Development and application of artificial neural network. *Wireless Personal Communications* (2018), Bd. 102: S. 1645–1656, URL https://doi.org/10.1007/s11277-017-5224-x

[Xia15] Xia, Long; Xu, Jun; Lan, Yanyan; Guo, Jiafeng und Cheng, Xueqi: Learning maximal marginal relevance model via directly optimizing diversity evaluation measures, in: *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, S. 113–122, URL https://doi.org/10.1145/2766462.2767710

[Xie24] Xie, Weijian; Liang, Xuefeng; Liu, Yuhui; Ni, Kaihua; Cheng, Hong und Hu, Zetian: WeKnow-RAG: An Adaptive Approach for Retrieval-Augmented Generation Integrating Web Search and Knowledge Graphs. *arXiv preprint arXiv:2408.07611* (2024), URL https://doi.org/10.48550/arXiv.2408.07611

[Yao24] Yao, Yifan; Duan, Jinhao; Xu, Kaidi; Cai, Yuanfang; Sun, Zhibo und Zhang, Yue: A survey on large language model (LLM) security and privacy: The Good, The Bad, and The Ugly. *High-Confidence Computing* (2024), Bd. 4(2): S. 100211, URL https://doi.org/10.1016/j.hcc.2024.100211

[Yu21] Yu, Shi; Liu, Zhenghao; Xiong, Chenyan; Feng, Tao und Liu, Zhiyuan: Few-shot conversational dense retrieval, in: *Proceedings of the 44th International ACM SIGIR Conference on research and development in information retrieval*, S. 829–838, URL https://doi.org/10.1145/

3404835.3462856

[Yu23] Yu, Wenhao; Zhang, Hongming; Pan, Xiaoman; Ma, Kaixin; Wang, Hongwei und Yu, Dong: Chain-of-note: Enhancing robustness in retrieval-augmented language models. *arXiv preprint arXiv:2311.09210* (2023), URL https://doi.org/10.48550/arXiv.2311.09210

[Yu24] Yu, Hao; Gan, Aoran; Zhang, Kai; Tong, Shiwei; Liu, Qi und Liu, Zhaofeng: Evaluation of retrieval-augmented generation: A survey, in: *CCF Conference on Big Data*, Springer, S. 102–120

[Zha20] Zhang, Yunyan; Xu, Guangluan; Wang, Yang; Lin, Daoyu; Li, Feng; Wu, Chenglong; Zhang, Jingyuan und Huang, Tinglei: A Question Answering-Based Framework for One-Step Event Argument Extraction. *IEEE Access* (2020), Bd. 8: S. 65420–65431, URL https://doi.org/10.1109/ACCESS.2020.2985126

[Zha23] Zhang, Yichi; Chen, Zhuo; Fang, Yin; Lu, Yanxi; Li, Fangming; Zhang, Wen und Chen, Huajun: Knowledgeable preference alignment for llms in domain-specific question answering. *arXiv preprint arXiv:2311.06503* (2023), URL https://doi.org/10.48550/arXiv.2311.06503

[Zha24a] Zhang, Mingtian; Lan, Shawn; Hayes, Peter und Barber, David: Mafin: Enhancing black-box embeddings with model augmented fine-tuning. *arXiv preprint arXiv:2402.12177* (2024), URL https://doi.org/10.48550/arXiv.2402.12177

[Zha24b] Zhang, Yazhou; Wang, Mengyao; Ren, Chenyu; Li, Qiuchi; Tiwari, Prayag; Wang, Benyou und Qin, Jing: Pushing the limit of LLM capacity for text classification. *arXiv preprint arXiv:2402.07470* (2024), URL https://doi.org/10.48550/arXiv.2402.07470

[Zha24c] Zhao, Penghao; Zhang, Hailin; Yu, Qinhan; Wang, Zhengren; Geng, Yunteng; Fu, Fangcheng; Yang, Ling; Zhang, Wentao; Jiang, Jie und Cui, Bin: Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473* (2024), URL https://doi.org/10.48550/arXiv.2402.19473

[Zho23] Zhou, Kun et al.: Don't make your LLM an evaluation benchmark cheater. *arXiv preprint arXiv:2311.01964* (2023), URL https://doi.org/10.48550/arXiv.2311.01964

[Zho25] Zhou, Jiawei und Chen, Lei: OpenRAG: Optimizing RAG End-to-End via In-Context Retrieval Learning. *arXiv preprint arXiv:2503.08398* (2025), URL https://doi.org/10.48550/arXiv.2503.08398

[Zhu24] Zhu, Kunlun et al.: Rageval: Scenario specific rag evaluation dataset generation framework. *arXiv preprint arXiv:2408.01262* (2024)

List of Figures

List of Tables

7 Appendix

Provision of Test Data and Source Code

Due to the size of the test datasets, evaluation results, and the related source code, these materials are not included as a physical appendix to this thesis.

The complete source code, evaluation scripts, and curated test datasets are publicly available on GitHub under the following repository:

`https://github.com/thng17/MasterThesisAppendix`

The full Chroma vector database used for retrieval experiments is provided separately via Zenodo due to file size limitations. It can be accessed through the following DOI:

`https://doi.org/10.5281/zenodo.15666607`

All sensitive configuration files, credentials, access keys, and other confidential information have been removed prior to publication.