MASTER'S THESIS 2026

Systematic Test Strategies for Generative Question Answering Systems

Jacob Bentzer, Henrik Vester

ISSN 1650-2884
LU-CS-EX: 2026-05

DEPARTMENT OF COMPUTER SCIENCE
LTH | LUND UNIVERSITY

EXAMENSARBETE
Datavetenskap

LU-CS-EX: 2026-05

Systematic Test Strategies for Generative Question Answering Systems

Systematiska teststrategier för generativa frågebesvarande system

Jacob Bentzer, Henrik Vester

Systematic Test Strategies for Generative Question Answering Systems

Jacob Bentzer
ja0564be-s@student.lu.se

Henrik Vester
he5834ve-s@student.lu.se

February 18, 2026

Master's thesis work carried out at
the Department of Computer Science, Lund University.

Supervisor: Emelie Engström, emelie.engstrom@cs.lth.se

Examiner: Elizabeth Bjarnason, elizabeth.bjarnason@cs.lth.se

Abstract

Generative AI question answering systems are increasingly deployed to support knowledge-intensive work. In high-stakes settings, the risk of unsupported or incorrect answers makes rigorous testing essential. However, traditional testing techniques struggle to address challenges such as non-determinism and the oracle problem. This thesis examines evaluation of generative AI-based question answering systems via a literature review and expert inquiries consisting of a workshop and interviews with practitioners. It contributes an extended taxonomy that structures accuracy evaluation into five dimensions: context relevance, answer relevance, faithfulness, correctness, and citation quality. Using three real-world case studies, we illustrate how metrics and frameworks are applied in practice and highlight common difficulties. From these observations, we propose discussion points for future work, including possible trade-offs between scalability and human-aligned judgments, limited mechanisms for continuous post-deployment quality assurance, and partial coverage of relevant evaluation dimensions. We also summarize our overall observations as five practical recommendations to practitioners. Together, the taxonomy, propositions, and recommendations provide a structured basis for designing comprehensive and maintainable evaluation strategies for generative AI question answering systems.

Keywords: question answering; evaluation; metrics; quality assurance; taxonomy

Acknowledgements

We would like to thank our supervisor, Emelie Engström, for her helpful input, invaluable guidance, and continuous support in answering many questions throughout the thesis.

We also wish to thank our examiner, Elizabeth Bjarnason, for her insightful feedback, which strengthened the quality of the thesis.

We would also like to thank Modular Management and the employees who participated in the workshop and individual interviews, for their time and willingness to share practical experiences and perspectives that informed this work.

Contents

Chapter 1
Introduction

In recent years, the number of applications that make use of generative AI for various forms of *question answering* has increased massively. Question answering includes all applications where a user asks an AI model a question, which it then answers. General examples include ChatGPT and Microsoft Copilot, but they may also be specialized to a specific domain, such as an industrial knowledge base [1]. Current state-of-the art question answering systems often make use of a *retrieval-augmented generation* (RAG) architecture, consisting of a retriever component coupled with a generative *large language model* (LLM). For a query, the retriever component retrieves relevant documents from the knowledge base, while the generator synthesizes these into a coherent and relevant natural language answer [2].

While question answering systems using generative AI have already seen an explosive growth in usage and been applied in diverse areas, there are still some obstacles to even wider adoption. Even leading LLMs are affected by *hallucinations*, which is variously defined as the model fabricating or responding confidently with false information [3, 4]. Especially in high-stakes fields with legal and regulatory limitations, such as finance, healthcare, or legal services, wrong answers from a question answering system are unacceptable. This highlights the need for being able to systematically test these kinds of systems and verify them against the requirements.

At the moment, there are no established best practices for how application developers of question answering systems can verify the system against these key requirements. Normally, they would be verified through standardized software testing methods such as unit and integration tests. However, these methods are ill-suited for generative AI because of its non-deterministic nature and the difficulty of defining metrics of response quality. Application developers are thereby confronted with fundamental challenges, including how to define the criteria for a good answer and how to specify what relevance means within the specific context of their business domain.

Due to the lack of standardized processes and tooling for testing such systems, many application developers rely on manually checking specific inputs or crowd-sourced evaluations [5]. The former is labor intensive and hard to reproduce while the latter is incon-

sistent and not suitable for critical systems. This leaves developers and organizations with uncertainty regarding the performance of their systems and hinders even more wide-spread adoption in industry. Thus, this thesis investigates both the challenges and existing solutions, including potential pitfalls for real-world industry applications. This leads us to the following research questions:

RQ1. **What are the current limitations and challenges associated with testing generative AI-based question answering systems?**

RQ2. **What existing methods are there for verifying question answering systems against accuracy requirements?**

RQ3. **What pitfalls can occur when using existing methods to automate question answering system testing in practice?**

The first question was chosen to identify the fundamental difficulties in testing generative AI-based question answering systems and understand the complexities of the field. This directly aids understanding of the second question by giving context to the methods described there. The second question builds on the first, offering insight into the field and provide a selection of methods that, individually or in combination, can be used to effectively test applications. To add practical weight to the findings, the third question was designed to shed light on potential pitfalls that may arise when attempting to apply these methods in a practical development environment. Together these questions were designed to provide a comprehensive understanding of both he theoretical and practical implications of testing generative AI systems.

The first two research questions are more theoretical and descriptive in nature. To answer these two questions, we perform a literature study to get an outline of the current state of the field, including limitations and challenges for testing and what methods already exist. We combine the literature review with expert inquiries with practitioners to validate and evaluate our identified methods from an industry perspective. The results from this part are contained in Chapter 4 and answer RQ1 and RQ2.

To answer RQ3, we focus on practical implications and pitfalls of testing generative AI-based question answering systems. From the identified evaluation methods and reported use cases found in the literature study and expert inquires, we present and analyze relevant case studies of industry-focused question answering systems described in literature. Thus, we can compare practical aspects of their respective evaluation processes and explore how these relate to our theoretical findings. These results answer RQ3 and can be found in Chapter 5.

Chapter 2
Background

This chapter provides the necessary background for understanding the challenges of testing generative AI-based question answering systems. We begin by giving an overview of the concepts of natural language processing and large language models. Then, we introduce the key concepts of *question answering systems* and *retrieval-augmented generation* (RAG), including how they function and how they relate to each other. Then, we address user expectations on such systems and what that means for the requirements that are put on them. Finally, we address common performance and reliability issues, such as hallucinations.

2.1 Overview

Over the past decade, *natural language processing* has made rapid progress in how computers interact with human language [6]. Advances across tasks such as machine reading comprehension, sentiment analysis and semantic similarity measuring have laid the foundation for modern applications that interact naturally with users [7]. Key in this development has been the emergence of *large language models* (LLMs) [8]. Trained on vast amounts of data, these models have catapulted natural language processing to the main-stream with impressive performances. The implementation a consumer is most likely to encounter is the generative pre-trained transformer (GPT) of which ChatGPT [9] is the best known example.

ChatGPT is an example of a question answering system, a system which allows users to pose questions in natural language and receive responses in kind. These systems have existed before but the advent of LLMs has greatly expanded their quality and performance. In particular, such systems are increasingly used for domain-specific question answering such as in the case of internal knowledge bases. Applications range from industry [1] to healthcare and legal services [10]. The use of question answering systems for these kinds of high-stakes applications makes their performance and reliability crucial.

A further step forward came with the development of RAG. While traditional LLMs generate answers based solely on pre-trained knowledge, RAG systems can dynamically retrieve

up-to-date information, making them better suited for domains where accuracy and domain-specific knowledge are critical [2]. These terms are presented here to lay the groundwork for the concepts discussed later in this thesis, and to understand something of how they are connected. In this thesis, we use the term *question answering system* when focusing on user-facing aspects and requirements, and *RAG* when referring to a typical system implementation and its testing.

2.2 Question Answering Systems

A question answering system is any system that is able to answer queries by users in natural language [11]. Typically included components and processes in a question answering system according to Farea and Emmert-Streib [12] are:

- Natural Language Understanding: The system interprets the semantic meaning, intent and context of the query.
- Information Retrieval: The system retrieves relevant information from its data sources, searching and selecting relevant passages or documents that contains potential answers.
- Answer Extraction or Generation: The system either directly extracts the information from its sources or generates a new answer in natural language.
- Scoring and Ranking: The system evaluates and scores potential answers based on specified attributes. Possible attributes include relevance, coherence, confidence and accuracy[1].
- Presentation of Results: The system presents the highest ranking answer to the user in a human-readable format.

2.2.1 Use Cases for Question Answering Systems

Question answering systems are applied throughout a wide variety of areas where fast and accurate information retrieval is highly valued. Common use cases include conversational agents, customer support chatbots, and domain-specific indexing of knowledge bases, for example ones implemented within a company to retrieve product or company-specific documentation. Such systems prove helpful especially in technological companies which amass massive collections of documentation including training material, design documents and research outputs [1]. Manual traversal of these documents can be tedious and challenging, which is why the fast and accurate information retrieval supplied by a question answering system is highly beneficial. Furthermore, these documents often contain product- or company-specific jargon, which a conversational agent can help interpret.

Question answering systems can be implemented in a several ways, depending on the requirements and the desired level of complexity [12]. Simpler methods include keyword-based question answering systems, matching keywords or key phrases from the query with

[1] Farea and Emmert-Streib [12] use the term *correctness* instead of *accuracy*

relevant documents, and rule-based question answering systems, which rely on predefined rules or patterns from the query to find the answer. While effective in narrow and controlled settings, these approaches are limited when handling more diverse or nuanced queries. This thesis therefore focuses on more advanced systems, leveraging a LLM to generate synthesized answers, supported by a retriever component that provides relevant context when needed. This combination makes it possible to deliver responses that are both more precise and more adaptable to specialized domains.

2.3 Retrieval-Augmented Generation

Retrieval-augmented generation (RAG) is an architecture used for question answering systems that augments a standard LLM with a retriever. For a query, the retrieval part retrieves relevant documents from a knowledge base (called *context chunks*), while the generating part synthesizes an answer based on the retrieved documents and the original question [2]. RAG is able to perform better than baseline non-RAG models in key applications such as question answering and fact verification [2].

2.3.1 Architecture

A RAG implementation is divided into three parts. These are the knowledge base, the retriever, and the generator [2]. See Figure 2.1 for an architectural overview of a sample RAG implementation.

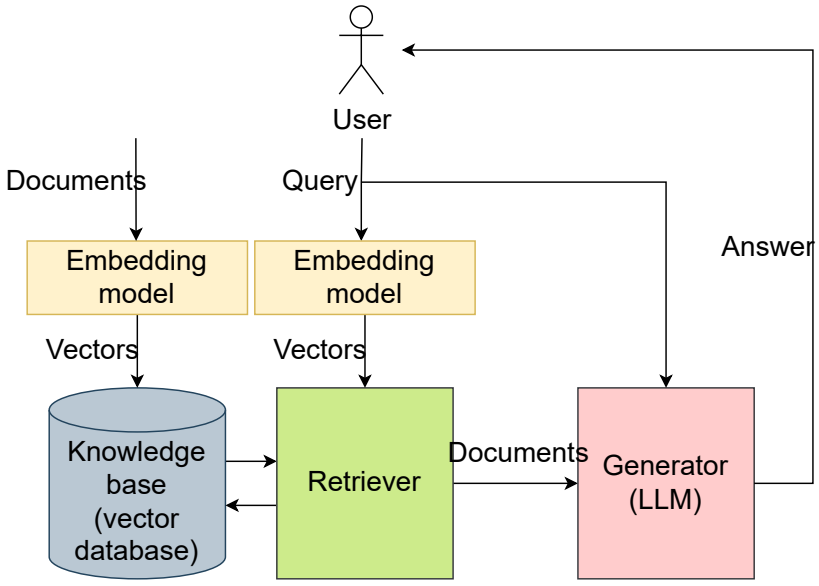

Figure 2.1: A sample RAG architecture where documents are indexed in a vector database, and an LLM synthesizes the query and retrieved documents into an answer.

The Knowledge Base The knowledge base contains and indexes all of the information that should be queryable in a question answering system implementation. Most commonly, textual documents are used but the knowledge base could in principle consist of any sort of media. An important characteristic of this data is that it may be unstructured, i.e. it does not have a predefined organization or structure. There are various ways of indexing this data, but the most common is to use an *embedding model* to generate vector embeddings. A commonly used embedding model, the `text-embedding-3-small` by OpenAI, returns vectors of 1536 dimensions [13]. A vector embedding is a point in a very high-dimensional space which compactly encodes the semantic information of a document or a part of a document. These vector embeddings are typically stored in a *vector database* which enables efficient lookup given a search vector. Note that the vector embeddings are only a way of semantically indexing the data. The documents still need to be stored conventionally, enabling other parts of the RAG to retrieve them when needed.

The Retriever When the user asks the question answering system a question, the user's question is embedded with the same embedding model used for the knowledge base. The *retriever* part of the RAG takes this vector and searches the vector database for the n closest matches (typically $n = 5$ or $n = 10$). Closeness is here defined as the distance between the search vector and the indexed vectors in the high-dimensional space. Due to the computational power required to compute these distances, a valid tradeoff for many applications is to only compute approximate distances, which are much quicker to calculate with an acceptable loss of precision. Cosine similarity [14] is a commonly used distance metric. The retriever retrieves the documents (or parts of documents) corresponding to the found vectors and sends them on to the generator.

The Generator The retrieved documents, as well as the original text of the user's query, are sent to the *generator*. The generator is an LLM which is trained to synthesize an answer to the user's query using the contents of the retrieved documents (which hopefully relate to the query). The generated answer is sent to the user. Optionally, the RAG system may choose to present the retrieved documents to the user separately from the main conversation in order to promote transparency (a form of explainable AI [15]). This provides the user with information about the provenance of the provided answer, and enables cross-checking of the answer should it be required.

2.3.2 Strengths and Weaknesses

RAG offers several advantages over purely parametric question answering models for implementation of question answering systems. To begin with, RAG architectures consistently outperform non-RAG baselines (e.g., BERT [16]) on benchmarks such as question answering and fact verification [2]. Beyond benchmark performance, a major strength of RAG lies in its ability to incorporate new or updated information without costly retraining, making it particularly suitable for settings where knowledge changes rapidly.

Because RAG depends on an external knowledge source, it is also well-suited to domain-specific applications. In such contexts, retraining or fine-tuning an entire large language model may be infeasible, while indexing relevant documents is both simpler and more cost-effective [2, 1]. This externalization of knowledge is also advantageous for *long-tail knowledge*,

i.e., rare or specialized information that parametric memory often fails to capture but which nonetheless may be critical in industrial applications where errors are unacceptable [17, 18]. Another important benefit of RAG is transparency. Unlike purely parametric models, RAG-based question answering systems can cite the sources used for a given answer, enabling users to independently verify responses. This increases trust and makes the system more transparent [19].

From a practical perspective, RAG systems are relatively straightforward to implement thanks to existing frameworks (e.g., Microsoft's Semantic Kernel [20]). Developers also retain flexibility in how retrieval is implemented, whether through live queries [21], vector databases [2], or knowledge graphs [17], allowing the system to be tailored to the specific needs of an application. Taken together, these advantages explain why RAG has become the dominant approach [12, 1] for question answering systems that must index large volumes of information and maintain accuracy in domains where obscure or rare knowledge is common and mistakes are unacceptable.

At the same time, RAG comes with disadvantages. A central limitation is that system performance depends heavily on the retrieval step. If the retriever surfaces irrelevant or misleading documents, the generator is often unable to detect this and may confidently produce incorrect answers [22]. In such cases, responsibility shifts to the user to vet the cited documents for relevance. This risk is heightened in domains with highly specialized jargon, where embedding models may struggle to provide accurate representations [1].

Another drawback is the added complexity of RAG systems. Compared to parametric models, RAG introduces additional infrastructure components which increase operational overhead. Moreover, the multi-step answering process can negatively impact latency and computational efficiency, making deployment more resource-intensive. In summary, RAG provides substantial benefits in accuracy, adaptability, and transparency, but these gains come at the cost of greater system complexity and reliance on high-quality retrieval.

2.4 User Expectations and Requirements

While user expectations and corresponding requirements naturally vary between applications, certain ones recur. Zhou et al. [19] summarize key dimensions of trustworthiness of RAG applications. These include *factuality*, meaning that the system's answers are factually correct, and *robustness*, the ability of the system to adequately handle errors, adversarial attacks, and external threats. Other dimensions include *fairness*, i.e., the mitigation of bias, as well as *transparency*, meaning that users can understand how a question was answered and with what information. Closely related to transparency is *accountability*, the expectation that the system's actions are responsible and traceable. The last dimension is *privacy*, ensuring that the user's personal information is protected throughout the system.

Taken together, these dimensions in turn form the basis of the requirements placed by end users or other stakeholders, and which developers must account for. Among them, factuality (which we from now on will call *accuracy*) is most closely tied to the challenges this thesis addresses, and so we will focus on this dimension. From this dimension, we can derive high-level functional requirements that specify what the system must do to ensure factual answers. In particular, the system must (1) ensure that answers relate to the question asked, (2) ensure answers are correct in themselves, and (3) enable users to understand how answers

were produced.

These high-level requirements then translate, in a given implementation, into more concrete requirements that ensure the overarching goals are met. For RAG systems, Knollmeyer et al. [23] outline such implementation-specific requirements. Accuracy, for example, can be decomposed into requirements on both the retriever and generator: *context relevance* of retrieved documents, *answer relevance* of generated responses, *faithfulness* of generated content to the sources, *citation quality*, and overall answer *correctness*. See Figure 2.2 for a diagrammatic overview of how trustworthiness dimensions give rise to high-level requirements, which in turn decompose into implementation-specific requirements in the case of RAG.

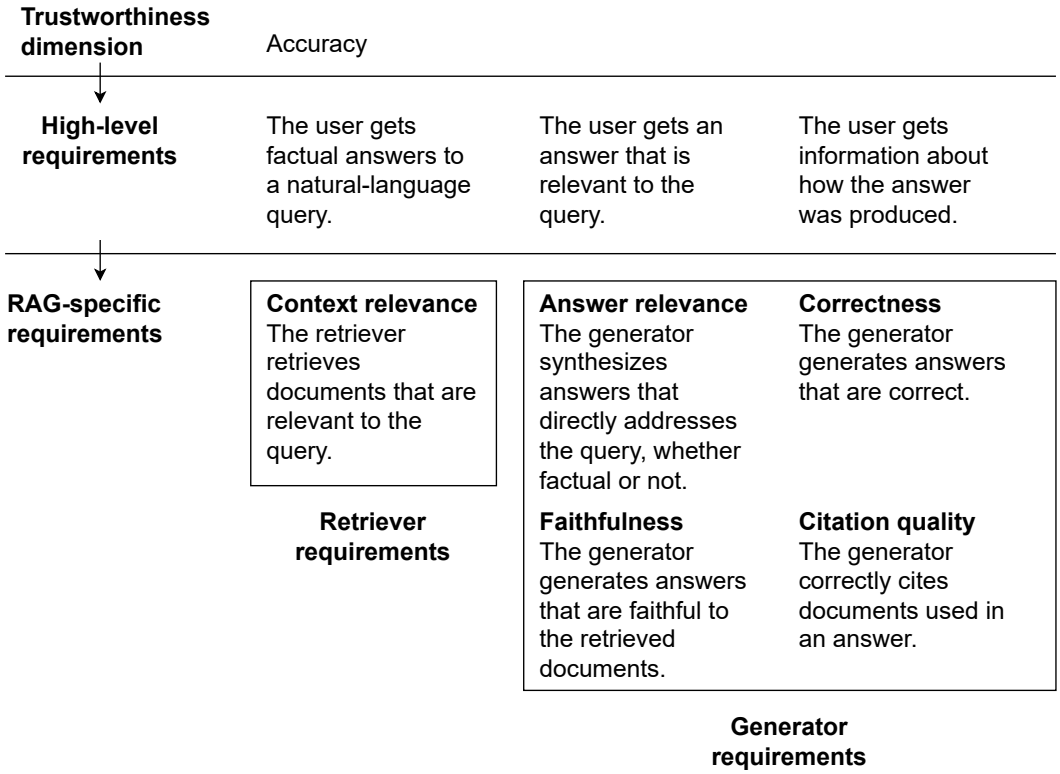

Figure 2.2: An overview of how the trustworthiness dimension of accuracy dimension gets decomposed into high-level requirements, followed by implementation-specific functional requirements that are scoped to either the retriever or generator component.

2.5 Performance and Reliability Issues

Even state-of-the-art LLMs utilizing RAG are still prone to hallucinations, typically understood as producing fabricated or factually incorrect information with unwarranted confidence [4]. In domains with strict legal or regulatory requirements, such as finance, healthcare or law, such errors are unacceptable. For example, there have been a multitude of cases where conversational agents supply users with incorrect medical information, which could

lead to severe personal harm if followed [24]. Further exacerbating these issues, RAG implementations may face challenges when used for increasingly specific applications, especially concerning misinterpretation [1] and vector embedding issues [22] with regard to domain-specific jargon and terminology.

The term hallucination in AI is broad, encompassing many forms of unintended or incorrect behavior [4]. It can refer both to fluent but irrelevant responses and to factually incorrect information that the model generates as its best guess. Due to this ambiguity, we will largely avoid the term and instead discuss user requirements and examine how well particular implementation choices, such as the choice of model architecture, affect how well they are met.

Even if design choices such as using RAG can reduce the frequency of hallucinations [2], they do not eliminate them. This raises the question of how well current question answering systems meet such requirements in practice, and critically, how we can assess how well requirements are met in a systematic way. To answer this, we turn next to the state of art in testing and evaluation of generative question answering systems. By reviewing both academic literature and practical perspectives from industry experts, we explore how limitations and challenges are currently understood, and what methods exist for verifying whether question answering systems live up to expectations.

Chapter 3
Method

This chapter describes the methods used to address the research questions of this thesis. We conducted a systematic literature review to explore existing challenges, evaluation methods and metrics for testing generative question answering systems. The review was performed in iterations, with each iteration using snowballing and exploratory searches on keywords to investigate the subject for the iteration. To complement and validate the findings from the literature, we conducted a workshop and interviews with industry practitioners. This provided practical perspectives on the usefulness and limitations of the identified approaches. Together, these methods enabled a comprehensive and broad investigation into the subject matter.

3.1 Literature Review

In the literature review, we limited ourselves to primarily consider question answering systems with a RAG architecture. The reasons for this are two-fold. First, because RAG is the dominant architecture for real-world applications of question answering systems. Second, because the RAG architecture is similar to other paradigms of generative AI-based question answering systems. For example, RAG differs from pure parametric memory-based LLM question answering systems only in the addition of the retriever component. We therefore believe that focusing primarily on RAG-based architectures in the literature study does not significantly affect the generalizability of the results.

The literature review consisted of a systematic mapping study of the field of testing generative AI question answering systems. The aim of the review is to present an overview of testing challenges, which allows us to answer RQ1, identify and classify existing evaluation methods of generative AI question answering systems, addressing RQ2, and to identify pitfalls that arise when these methods are applied in practice, addressing RQ3.

3.1.1 Data Sources

We limited ourselves to two databases: Scopus and Google Scholar. A comparative study by Valente et al. [25] found that Google Scholar provides the highest availability of documents among surveyed computer science databases. However, Scopus provides significantly more search features than Google Scholar, and returns fewer irrelevant documents while still having good coverage. This leads us to combine the two databases.

3.1.2 Search Strategy

Because the review is of an exploratory nature, we performed the literature retrieval iteratively, with search strings and scope changing with every iteration. Each iteration consisted of the following activities:

1. identifying concepts to be explored in the iteration
2. searching for papers through snowball sampling of existing papers and exploratory searches
3. filtering papers first based on their abstract, then through a full-text review
4. mapping papers in a concept matrix
5. identifying adjacent concepts and keywords for use in the next iteration

See Figure 3.1 for a diagrammatic overview of the activities performed in each iteration.

The iterative process builds on the idea that each iteration reveals new areas of interest, concepts, and questions, which in turn guide the next iteration. This approach enables an exploratory examination of related work and helps ensure that relevant concepts are not overlooked due to our initial unfamiliarity with the field. For instance, one iteration might begin with the question "What are the most common approaches to evaluating generative AI question-answering systems?" The examination of this question could then give rise to subsequent inquiries, such as "Which existing test suites can be applied?" and "What factors influence the selection of evaluation metrics?" While individual iteration topics were not documented separately, the evolution of the study's focus is captured through the expansion of the concept matrix and the final classification of concepts reported in Appendix A. In order to prime the first iteration, we need a set of initial questions and seed sources. The initial questions correspond directly to the research questions RQ1–RQ3, while we use the papers cited in the background section (Chapter 2) as seed papers.

3.1.3 Paper Identification

The primary method used for identifying relevant paper was the use of forward snowballing, i.e., reviewing the references cited in previously identified papers to identify additional sources for inclusion. However, a source with its references is often not relevant in its entirety. The most common case we were subjected to is papers which propose a novel RAG implementation technique and corresponding evaluation measures. In that case, only the sections pertaining to evaluation methods are of interest to us and we would in this case limit our

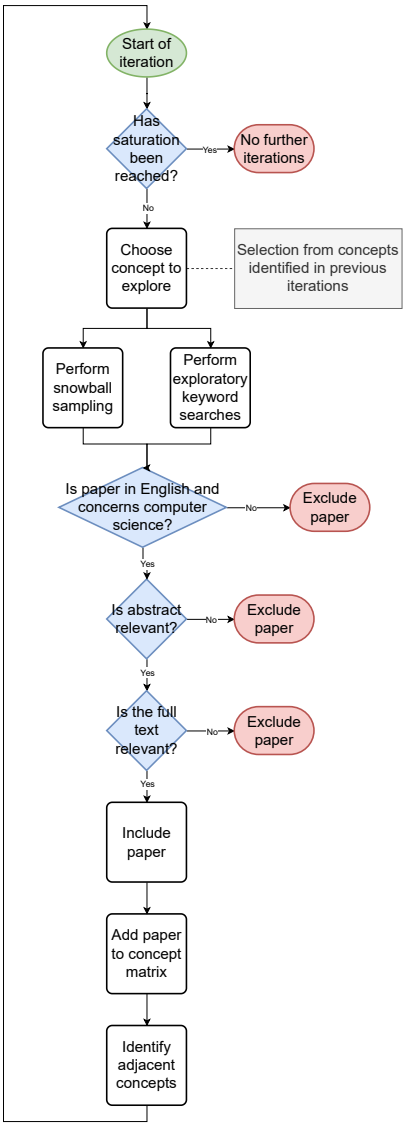

Figure 3.1: The activities performed in each iteration, organized as a flowchart.

snowball sampling to references cited in those sections. When papers found through snowballing did not sufficiently cover a particular concept, we conducted searches on identified keywords in the aforementioned two databases to identify additional, relevant papers. These were treated identically to the papers found through the snowballing process with regard to the subsequent filtering process.

3.1.4 General Selection Criteria

Papers were selected for inclusion in a three-step process. First, papers that were not in English or whose contributions were not in the area of computer science were filtered out. The titles and abstracts of found papers were then screened for relevance. The criteria for relevance evolved across iterations as our focus shifted. For instance, when examining practical evaluation methods, we included papers describing repeatable or automatable approaches and excluding those relying on one-off, manual, or time-intensive evaluations, or on metrics with limited human alignment. We describe the inclusion and exclusion criteria for each concept in Table A.3 in Appendix A.

Finally, we performed a full-text review of each paper as a final selection step. Especially in later iterations, where the areas to be explored shifted to be more and more specific, the full-text review ensured that included papers were truly relevant. Both authors independently reviewed and filtered papers, resolving disagreements through discussion, which increased the robustness and reliability of the review process.

We aimed to only include peer-reviewed papers published in reputable venues. However, in the case of recent developments or niche, technical-report-like papers where the information could not be found elsewhere, we also included preprints. In some cases of specific evaluation metrics and frameworks, neither the use of peer-reviewed or pre-published papers were sufficient to give a detailed enough description of practical usage aspects. In this particular area, we supplement information found in the literature review with official technical documentation. Venue credibility was assessed based on publication type and publisher reputation. Well-known peer-reviewed journals and conferences were considered credible by default. For venues we did not recognize, we verified their legitimacy through web searches and checking indexing in recognized databases. In the case of preprints, we considered both the reputation of the authors and the number and types of citations the work had received, as an indication of its acceptance within the research community.

3.1.5 Selection of Case Studies

In the final iteration of the literature review, we selected three case studies that highlight how evaluation is carried out in practice. One of the case studies had already been identified in earlier iterations. The other two case studies were selected in this final iteration. In this iteration, each paper corresponded to a case study which was screened for relevance according to the following two aspects: *industry relevance* and *focus on the evaluation process*.

Initially, we judged case studies to have *industry relevance* if the system was deployed to users in a production setting. However, we found that this criteria was too strict in that only one case study met it. Accordingly, we relaxed this criteria to allow case studies in which prototype or pilot systems were tested with actual users in an industry setting, but where the system was not used independently outside the case study setting. Case studies were

deemed to *focus on the evaluation process* if the question answering system was (1) evaluated at all according to any metric, and (2) if this evaluation process was described in the case study.

We identified a total of three case studies passing the inclusion criteria. Among papers not included, the most common reasons for exclusion were that it either did not relate enough to practical, industry-focused use cases, or that it focused too little on the evaluation process. None of the three included studies focus entirely on the evaluation process; rather, they describe all stages of the development process of such a system. However, all three highlighted the evaluation process in such detail that it gave practical insights. All three case studies were performed in real-life settings. However, UniAsk [26] is the only one in which the system was deployed to users. The other two consist of experimental systems with the primary goal of assessing the suitability of further, more production-focused efforts.

3.1.6 Concept Mapping

Each included paper was mapped against a concept matrix, where the left-most column contained all relevant papers followed by a column for each concept. A connection between a paper and a concept is marked by an 'X' in the corresponding cell. We initially selected the following concepts:

- testing challenges
- evaluation
- datasets
- hallucinations [1]
- user expectations
- requirements specification

These were chosen because they relate to key aspects of the first two research questions, namely current limitations and challenges, existing methods of evaluation, and what requirements users and other stakeholders place on these systems. Some related concepts were considered for inclusion but ultimately excluded because they did not contribute to directly answering our research questions, such as:

- approaches for improving RAG system performance
- why generative AI suffers from reliability problems
- architectural or implementation concerns of specific systems

Concepts not related to our primary focus of accuracy were also excluded, for example relating to the accountability of RAG systems, protection of personal information, or minimizing bias.

[1] We use the term *hallucinations* as a broad label for system accuracy issues, including but not limited to instances traditionally referred to as hallucinations, which itself is not a well-defined term.

During the course of the literature review, the set of concepts was augmented in two ways. Some were split (for example, the *dataset* concept was split into *evaluation dataset*, *benchmark dataset* and *training dataset*) while others were added, such as *metrics* and *legal challenges*. At the conclusion of the literature review, a total of 27 concepts were identified. Most concepts are directly related to the research questions, while some functioned as a complement to other concepts (for example, we added an *industry* concept in an attempt to distinguish how well varying evaluation methods are adapted for industrial application). We list the concepts used in the matrix in Appendix A. Due to the size of the concept matrix, we do not include it in this thesis. Instead, we have made it available at DOI: 10.5281/zenodo.18164715 [27].

3.1.7 Expanding the Search Space

As we gained more information about the field, new keywords, concepts, and sources were identified and used as a starting point for the next iteration. We conducted iterations until *saturation* was reached, i.e., when additional papers no longer introduced new concepts or perspectives relevant to our research focus.

3.1.8 Mapping Results

At the conclusion of the review, 106 relevant papers were included. Papers were taken from a total of 29 venues (full list available in Table A.2 in Appendix A). We note that it is rare for a venue to appear more than once in our list of papers – only 2 venues (excluding arXiv) pass this criterion. The large number of distinct publication venues indicates a broad and diverse literature base. Due to the large number of distinct venues, the reviewed works likely reflect multiple research communities and perspectives rather than being confined to a specific venue or subfield. This diversity strengthens the generalizability of our findings and reduces the risk of venue-specific bias. Additionally, it suggests that the topic is being explored across multiple research communities, which indicates that this is an emerging or multi-disciplinary area. Most papers were mapped to more than one concept; on average, each paper was mapped to between three and four concepts.

3.1.9 Threats to Validity

Several factors may have influenced the validity of the literature review. In accordance with the categorization proposed by Ampatzoglou et al. [28] for secondary studies, we structure this discussion around *study selection validity*, *data validity* and *research validity*.

Study Selection Validity

Study selection validity concerns whether the included studies provide a representative view of the research field. The literature review was conducted with an exploratory and iterative search strategy rather than relying on a specified set of predefined search strings. While this approach was chosen to gain an overview of a rapidly developing field with in many cases inconsistent terminology, it introduces a risk that relevant studies may have been overlooked. This risk is particularly high with papers weakly connected to the initial seed papers.

To mitigate this risk we combined snowballing with searching for keywords and iteratively expanded the search space as new concepts emerged. The search process continued until saturation was reached, further mitigating the risk. Furthermore, the inclusion of papers from a large number of distinct venues suggests broad coverage of the field and reduces the likelihood of venue-specific bias.

Another potential threat arises from the inclusion of preprints and technical reports in addition to peer-reviewed publications. While these sources were necessary to cover emerging or highly technical topics not yet studied in peer-reviewed literature, they have not undergone formal peer review and need to be treated with caution. To mitigate this risk, such sources were assessed based on author reputation, citations and by cross-checking consistency with more reputable sources.

Data Validity

Data validity concerns the validity of data extraction and analysis. A key challenge in the literature review was the inconsistent terminology used across papers and publications. Similar concepts were often described using different terms, and in some cases it was necessary to judge whether distinct terms referred to the same underlying idea. These interpretative decisions introduce a risk of subjectivity and potential loss of nuance. This risk was mitigated through continuous discussion between authors and by cross-checking terminology across multiple sources. Concepts were only unified when supported by clear conceptual overlap.

Another data validity concern stems from differences in assumptions across the included studies, such as intended application domain, evaluation baselines, or dataset characteristics. Such assumptions affect both the design of evaluation methods as well as how to interpret test results. To address this, the synthesis focused on higher-level conceptual patterns rather than direct statistical comparisons, and clearly stating assumptions where relevant.

Research Validity

Research validity concerns the extent to which the design of the literature review support the reasoning and conclusions in regards to the research questions. One potential threat to research validity is limited repeatability, since synthesizing information from different sources requires interpretive judgment. If not sufficiently explained and documented, other researchers would find it difficult to repeat the study and reach similar conclusions. To mitigate this threat, the thesis includes both a flowchart over our research process (Figure 3.1) and our concept matrix (Appendix A).

Another potential threat to research validity concerns the generalizability of the conclusions drawn from the literature. As previously mentioned in Section 3.1, the literature review primarily considered question answering systems with a RAG architecture, which might limit the ability to apply insights and conclusions from the literature review to generative AI question answering systems in general. This focus reflects the prevalence of retriever components in contemporary research and practice, and that including a retriever component introduces evaluation challenges that are not present otherwise.

While this scoping decision emphasizes systems with retrieval components, it does not exclude systems without one. Many of the identified evaluation metrics and methods concern the generation and evaluation of answers rather than the retrieval mechanism itself. The

potential impact of this focus on the generalizability of the findings is further discussed in Section 6.1.

3.2 Expert Inquiries

The purpose of our expert inquiries is to complement the literature review and validate its findings, as well as evaluating the practical applicability of found methods and metrics. From our literature review, we identify methods and metrics for evaluating AI-based question answering systems, but since the field is immature, practical validation is needed.

We selected participants from Modular Management, a Stockholm-based management consulting firm with a presence in four countries. They specialize in the modularization of product architectures. To support their consultants in this work, they have an in-house development team which maintains the PALMA product management platform. PALMA is used both internally and externally by Modular Management's customers. Modular Management has experience in both developing and using AI-based question answering systems in their daily work, as well as experience in deploying large-scale enterprise software systems and associated quality assurance challenges. A total of five employees of Modular Management participated in the data collection. Three of them, all developers, took part in the initial workshop, together with the *system architect* and *customer success manager*. These participants were chosen to capture perspectives from all relevant stakeholders: technical, managerial, and from the customer's point of view.

The data collection was conducted in two stages. First, a *joint workshop* was held with all five participants, where preliminary results from the literature review and practical work were presented to provide context and stimulate reflection. The workshop included an open *Q&A discussion session*, during which participants were encouraged to share their experiences and opinions freely. This exploratory discussion helped identify which themes and questions were most relevant in practice.

Following the workshop, the *System architect* and the *Customer success manager* were selected for *individual semi-structured interviews*. These interviews followed a predefined guide based on questions aligned with research questions RQ1–RQ3, focusing on testing strategies, quality dimensions, and the perceived applicability of the proposed metrics and methods. This allowed for a deeper and more systematic exploration of the themes that had emerged during the workshop.

Each interview lasted approximately 30–45 minutes and was conducted through the use of video conferencing tools. All participants gave informed consent to the interviews, and detailed notes were taken to support further analysis. We chose not to record the workshop or interviews to encourage open discussion and reduce confidentiality concerns in an industrial context. Additional details about the interview participants and the complete interview guides are provided in Appendix B.

3.2.1 Analysis

The material from the workshop and the interviews consisted of detailed notes. This material was reviewed and thematically analyzed by coding relevant passages according to overarching

themes, using the free software *Taguette* [29]. To reduce researcher bias, both authors participated actively during this process. A total of 14 themes were identified, which can be grouped and summarized as follows. (1) *Quality assurance*: focus on ensuring system quality through structured testing, measurable metrics, and appropriate evaluation methods. (2) *Business requirements and stakeholder alignment*: the need to align quality assurance processes with business goals and ensure stakeholder interests are considered in system design. (3) *Continuous improvement and system learning*: highlights the importance of ongoing system refinement and adaptation based on evaluation and user feedback. (4) *Framing a question answering system and its role*: involves the positioning and communication of a question answering system's function, especially relating to how this affects user expectations and subsequent requirements. (5) *Perception of accuracy and user expectations*: explores how user trust and system accuracy influences expectations and overall satisfaction. (6) *Test automation and testing challenges*: issues pertaining to managing scalability and challenges of maintaining an up-to-date test suite. (7) *Trust and transparency*: the importance of clear communication about system limitations to build user trust in an enterprise system. A full list of themes is available in Appendix B.

Coding the material like this enables us to identify and compare recurring themes across participants and gain new insights related to the research questions. Particular attention was paid to how well the practical experiences of the participants aligned with the findings from the literature study. This enables us to identify areas where academic approaches may need adaptation to fit real-world enterprise contexts.

3.2.2 Threats to Validity

There are several factors that may have influenced the generalizability and validity of the interview study's results.

Researcher influence. Presenting a subset of the preliminary results before the interviews may have shaped participant responses or focused the discussion too narrowly. This step was, however, necessary to establish a shared understanding of key concepts such as testing metrics and methods. Care was taken to present results neutrally, and avoiding evaluative language that could bias the opinions of participants.

Limited representativeness. The study included a small number of participants from a single company, which limits the generalizability of the findings. However, the goal was depth over breadth. The selected participants represented distinct roles that together provide a well-founded view of practical applicability. Furthermore, the focus of the interview study was to validate and elaborate on the findings of the literature review, rather than to uncover entirely new perspectives.

Researcher bias. The thematic analysis relies on interpretations by the authors, which may introduce subjectivity. We mitigate this risk by grounding the analysis in the predefined research questions. Both authors participated actively in the analysis process, which reduces the risk of bias.

Chapter 4

State of the Art in Question Answering System Testing

Based on the literature review and expert inquiries, in this chapter we examine the current state of the art in question answering system testing, outlining the limitations and challenges of testing generative question answering systems (RQ1), and existing methods for verifying question answering systems against accuracy requirements (RQ2). Within the latter, the literature reveals a useful distinction between the metrics employed to measure performance and the evaluation frameworks that structure and employ these metrics. Accordingly, we first review commonly used metrics and then turn to various evaluation frameworks that apply them.

To structure our analysis in this chapter, we condense the 27 mapping concepts, from the literature study into five overarching areas of interest. This enables us to consider similar concepts together. We summarize these in Table 4.1. See Appendix A.1 for a table showing all 27 concepts and which area each concept is mapped to.

Table 4.1: An overview of the included papers according to our classification. Note that a paper may belong to several categories. Some concept labels are used as metadata and are not mapped to any category.

Category	Number of papers
1. Testing challenges	43
2. Evaluation methods	68
3. Evaluation metrics	23
4. Question dataset	28
5. Human alignment	22

The papers in each group were analyzed together with the aim of providing an overview

of testing challenges (RQ1) as well as existing methods and their specific challenges (RQ2). To develop our contributions, we unified and categorized concepts and methods across papers. We also qualitatively compared the identified metrics and methods along dimensions such as human alignment, computational efficiency, and ease of use.

4.1 Limitations and Challenges (RQ1)

Testing generative AI-based RAG question answering systems presents unique challenges compared to traditional software testing. These systems use complex algorithms, often based on neural networks, to synthesize new text based on patterns from their training data in order to answer user queries. Three key challenges arise: (1) highly varied and unpredictable input-output behavior, (2) limited interpretability of model reasoning [30] and (3) additional complexity introduced by retrieval components.

The *first* challenge stems from the combination of an effectively infinite input space and the non-deterministic nature of generated outputs [30]. In traditional software systems the relationship between input and output is deterministic; given the same input, the same output is expected. In contrast, generative question answering systems accept a near-limitless variety of user queries that may differ syntactically, semantically or contextually, and may produce different but plausible answers, even for identical inputs. This variability is often attributed to the stochastic nature of the generation process, such as the use of sampling strategies like top-k or nucleus sampling [31].

From a testing perspective, this dual unpredictability severely undermines conventional test design principles. Designing exhaustive input coverage is infeasible, highlighting the need for careful test suite design (discussed further in Section 4.2.4), and test suites cannot rely on exact matches since multiple correct answers may exist and be generated. Consequently, evaluation approaches may need to consider semantic equivalence rather than strict syntactic equality to avoid failing correct responses expressed differently than expected [32]. This variability complicates automation, reproducibility and regression testing.

These issues are closely related to the well-known *oracle problem*, which refers to the difficulty of determining whether a given output should be considered correct [33]. In this context, oracle refers to an entity which is always able to decide whether or not an observed behavior is to be labeled as correct or incorrect. Common solutions include developers assuming the role of oracle and manually validating the generated answers or creating reference answers to which the generated answer can be compared [34, 35]. These reference answers, often referred to as a *ground truth* or *gold standards* act as benchmarks of correctness. However, for generative question answering systems, defining and maintaining such ground truth is itself problematic since multiple valid answers may exist. The reliance on human oracles therefore remains a serious obstacle to test automation.

The *second* challenge is the limited interpretability of the underlying generative AI models. Modern systems often rely on neural networks with billions of parameters, which as they provide little transparency to their reasoning for their outputs makes it very difficult for a developer to understand how a particular answer was derived [7]. This lack of transparency affects testing in several ways: it makes it difficult to design test cases that achieve meaningful coverage, hinders understanding the root cause of unexpected outputs, and complicates diagnosing failures when they occur. Moreover, the lack of interpretability limits insight into

how changes to inputs or models may influence outputs [32], which is especially problematic for regression testing or safety-critical applications.

The *third* challenge arises from the inclusion of a retrieval component in the RAG system. While RAG systems were developed to increase the accuracy and credibility of LLM outputs [19], the addition a retriever component introduces another layer of complexity. The system's performance now depends not only on the generative model but also on the quality of both the retriever and the retrieved documents.

Yu et al. [36] present several challenges related to evaluating the retriever. The potential knowledge bases, from which information can be drawn, can vary greatly in in size, ranging from structured databases to the entire World Wide Web. Furthermore, the temporal nature of information adds another layer of difficulty. Data that was once accurate and relevant may become invalid and outdated over time, which complicates reproducibility and long term evaluation. Last, the diversity of information sources leads to the possibility of retrieving misleading or low-quality information which creates significant challenges in assessing the effectiveness of filtering and selecting the most pertinent information [36]. As a result of this, effectively evaluating systems with a retriever component may require additional, retriever-specific metrics, which we discuss in detail in Section 4.2.2. Furthermore, evaluating the retriever and generator components separately may not be effective in capturing system behavior that arises as a result of the overall interaction between the two components. This may necessitate the use of additional end-to-end metrics which takes these interactions into account [36].

The challenges presented in this section are closely connected. The inherent variability in generative models complicates not only the evaluation of outputs, but also the understanding of why certain outputs arise. This uncertainty is amplified by the limited interpretability of the models themselves. The addition of a retriever component furthers this complexity since retrieval errors may affect the generated content, and non-deterministic generation makes it harder to isolate whether observed failures stem from retrieval or generation. In practice small variations in one component influence behavior of the entire system, meaning testing must not only assess each component in isolation but also carefully examine their interactions and dependencies to understand system-wide behavior. While this section has focused on general challenges, more specific issues related to measurement methods and implementation details will be discussed throughout the chapter as they arise.

4.2 Testing Activities

We have identified the following four key activities that are performed during the testing of a generative AI-based question answering system.

1. Deciding what the requirements are of the system.
2. Deciding on what metrics should be used for evaluating these requirements.
3. Selecting appropriate methods for testing the system and collecting the metrics.
4. Creating question datasets, either manually or through generation.

We will not discuss requirements specification in further detail here, but refer instead to the discussion in Section 2.4. In the rest of this chapter, we will go into more detail on each of the three subsequent activities: deciding on appropriate metrics, selecting evaluation methods, and creating test suites. We will conclude the chapter by briefly discussing competing methods of improving accuracy of a RAG system other than testing.

Because each of the activities (excluding requirements specification) affects test case design, we need to briefly discuss what a question dataset for a RAG system looks like. A question dataset for a RAG system is typically a carefully constructed dataset of question-answer-context (QAC) tuples [32] where the question serves as input to the system, and the answer and context as evaluation references. For example, for the question "What is the capital of Sweden?", the context might be the string "Stockholm is the capital of Sweden" and the reference answer "Stockholm". The design of these tuples, i.e., what annotations we supply and the kinds of questions and answers they contain, constrains which metrics can be applied to a certain system and the system behaviors that can be measured.

4.2.1 Evaluation Metrics

Before we address testing methodologies for RAG-based question answering systems, it is essential that we describe the process of how metrics are chosen. As previously noted, the central requirement under consideration is accuracy. While accuracy provides a useful high-level description of system performance from a user's perspective, it does not by itself constitute a systematic or quantitative measure. To enable structured evaluation, the testing process therefore relies on well-defined metrics that capture different dimensions of system behavior related to accuracy.

We begin from the set of general requirements that reflect desirable properties of RAG-based question answering systems as outlined in Section 2.4. These requirements serve the basis for formulating evaluation questions about typical system behavior. Metrics are then selected to address these questions, following the *Goal Question Metric* (GQM) approach [37].

Metrics serve several purposes during the testing process [38]. They enable the measurement of performance across system configurations, facilitate the detection of regressions, and ensure alignment with user expectations and broader organizational goals. During development, metrics guide component selection and configuration, provide diagnostic insights during debugging, and support long-term monitoring of system quality.

It is essential that metrics capture genuinely meaningful behavior about the question answering system. *Human alignment* measures the extent to which a metric reflects what humans value or perceive as important with regard to the behavior of a system. Therefore, the degree of human alignment is a critical consideration which we will examine for each metric in the following sections.

The landscape of metrics is diverse. Some are tailored to specific applications, while others are broadly applicable. They range from low-level technical indicators to higher-level proxies for user satisfaction, with the latter often being composed of one or more of the former. To bring structure to this variety, we introduce a taxonomy of metrics (see Section 4.2.2) along with four dimensions.

Level of Application Metrics can be applied in different parts of the system. *End-to-end metrics* treat the system as a black box, capturing overall performance and often aligning

closely with human judgments. *Component-level metrics* target specific components, such as the retriever or generator, and provides developers with actionable feedback. A third category, *business metrics*, relates to organizational outcomes (such as reduced ticket escalation rates in customer service). While highly aligned with user value, business metrics are use-case specific and less directly linked to potential technical adjustments.

Collection Method Metrics also differ in how they are collected. *Non-LLM metrics* typically rely on statistical comparison against reference outputs, and offer a high level of determinism while being practically convenient. *LLM metrics*, by contrast, make use of an LLM to assess system performance. While less deterministic and harder to setup, they are typically more aligned with manual evaluation criteria.

Input Structure Another way metrics differ is in what type of interactions are used to calculate them. *Single-turn metrics* assess isolated user interactions, typically in the form of a single question-answer pair. These are generally easier to construct and interpret, in contrast with *multi-turn metrics* that capture behavior across entire dialogues. These are often more representative of real-world use while being more complex to incorporate [39].

Output Format Metrics can produce *numeric scores* (typically normalized to lie between 0 and 1), *discrete categories* (such as pass/fail or quality tiers), or *rankings* across outputs. Numeric scores lend themselves to aggregation and statistical analysis; categorical outputs provide coarse-grained but easily interpretable judgments; rankings are most useful when comparing relative performance as opposed to absolute scores.

Several considerations arise when selecting which metrics to use for testing a particular application [23, 38]. End-to-end metrics are generally preferable as a starting point, since they directly reflect user requirements. Component-level metrics can then complement these by isolating specific system weaknesses. The prioritization of metrics should reflect that of the requirements: in high-stakes domains, for instance, factual accuracy may take precedence over helpfulness or fluency. Existing system weaknesses may also influence the prioritization of metrics [23].

Metrics should also be reliable and objective. Agreement among human evaluators serve as important criteria for validating their robustness. Without such agreement, automating the collection of metrics risks producing misleading conclusions. Furthermore, having more metrics may not necessarily be better: a smaller set of informative, interpretable, and actionable metrics is preferable to a large collection of loosely correlated metrics [38].

Because question answering systems can be considered information retrieval systems (retrieving an answer given a query), the standard statistical metrics of *precision* and *recall* can be applied to such systems, both on an end-to-end level and for specific components. For example, an answer may be subdivided into claims or tokens. Each correct token or claim in the generated answer is a *true positive*, while each incorrect claim or token is a *false positive*. Tokens or claims which should have been present in the generated answer, but were not, are considered *false negatives*. *True negatives* are those claims or tokens which should not and were not included in the generated answer.

Most of the metrics described in this chapter can be seen as extensions or adaptions of these standard statistical measures, differing primarily in how true positives, false positives,

false negatives and true negatives are defined, and their ratios computed and aggregated. By changing the evaluated part (such as entire answers, individual claims or tokens) these metrics allow for more fine-tuned assessments.

4.2.2 Extended Taxonomy (RQ2)

Knollmeyer et al. [23] propose a taxonomy of metrics building on five key dimensions. These dimensions correspond directly to our implementation-specific requirements outlined in Section 2.4. The dimensions consist of *context relevance*, which focuses on how effectively the retriever component selects relevant information, as well as *faithfulness*, *answer relevance*, *correctness* and *citation quality*, which focuses on the reliability and accuracy of the generated responses. These dimensions serve as the foundation for formulating the evaluation questions and selecting the metrics, linking the metrics to well-defined aspects of system behavior. We do not claim to have exhaustively covered all possible metric dimensions or questions within this taxonomy, rather we aim to illustrate representative examples of relevant evaluation questions and the metrics that address them. We build on the taxonomy by further sub-classifying these dimensions according to implementation concerns, and classifying concrete metrics in the taxonomy. We summarize our surveyed metrics according to our taxonomy in Table 4.2.

Table 4.2: A summary of metrics according to our taxonomy.

Metric dimension	Level of application	Metric	Collection method	QDR	Input	Output	Ref.
Context relevance	Retriever	Precision@k	non-LLM	Q, C	Single-turn	Score	[40]
		Recall@k	non-LLM	Q, C	Single-turn	Score	[36]
		MRR@k	non-LLM	Q, C	Single-turn	Score	[36]
		LLM-as-a-judge	LLM	Q, C	Single-turn	Score	[41]
Faithfulness	Generator	K-Precision	non-LLM	Q	Single-turn	Score	[42]
		BERTScore	LLM	Q	Single-turn	Score	[42]
		LLM-as-a-judge	LLM	Q	Single-turn	Pass/fail, Score	[42, 43, 35]
Answer relevance	Generator	Answer Relevance Score	LLM	Q	Single-turn	Score	[41]
Correctness	End-to-end	Exact string match	non-LLM	Q, A	Single-turn	Pass/fail	[44]
		n-gram matching	non-LLM	Q, A	Single-turn	Score	[42]
		BERTScore	LLM	Q, A	Single-turn	Score	[45]
		LLM-as-a-judge	LLM	Q, A	Single-turn	Score	[42]
Citation quality	Generator	Citation recall	LLM	Q	Single-turn	Score	[46]
		Citation precision	LLM	Q	Single-turn	Score	[46]

Note. QDR = Question Dataset Requirements; Q = Question; C = Reference context; A = Reference answer; Ref. = References.

Context Relevance The metric dimension of *context relevance* focuses on how effectively the system is able to retrieve relevant context from its external source. The metrics in this dimension are therefore scoped to the retriever component.

Following the GQM approach, the metrics in this dimension should aim to answer questions related to the retrieval of documents. While these can be formulated in a variety of ways, we summarize common questions as follows: (1) *To what extent does the retriever select context that is relevant to the query?*, (2) *To what extent are all relevant context elements successfully retrieved?*, (3) *To what extent is the retrieved context ranked and structured in a way that allows the LLM to effectively identify and use relevant information during generation?*.

Because the retriever can be seen as a classifier which outputs whether context data is relevant or irrelevant (i.e., it was retrieved or not), accuracy and precision metrics can be applied here as well [36]:

- A *true positive* is a relevant context chunk that was retrieved.
- A *false positive* is an irrelevant context chunk that was retrieved.
- A *true negative* is an irrelevant context chunk that was not retrieved.
- A *false negative* is a relevant context chunk that was not retrieved.

An example of a metric measuring the first evaluation question is *Precision@k*. Precision@k measures the proportion of retrieved context passages among the top k results that are actually relevant to the query. It quantifies how effectively the retriever selects only useful information, reflecting the selectivity and quality of the retrieved context [40].

$$\text{Precision@}k = \frac{\text{TP}_k}{k} \tag{4.1}$$

To answer the second the second evaluation question, *Recall@k* can be used. Recall@k measures the ratio of relevant context chunks that were retrieved across the number of total relevant context chunks RD across the top k results, rewarding cases where all relevant context chunks were retrieved and penalizing cases where relevant context chunks were not retrieved [36]. In other words, the ratio is produced by dividing the number of true positive chunks contained in the first k results, by the sum of all the true positives and the false negatives.

$$\frac{\left|\text{RD} \cap \text{Top}_k\right|}{|\text{RD}|} = \frac{\text{TP}_k}{\text{TP} + \text{FN}} \tag{4.2}$$

Part of the third evaluation question can be measured by *MRR@k*. MRR@k is a metric across an entire test suite which measures the retriever's ability to correctly rank context chunks considering the top k chunks. This score penalizes cases where less relevant context chunks were ranked higher than more relevant context chunks and vice versa [36, 23]. The above mentioned metrics require ground-truth information about which context is relevant and not and therefore the question dataset must be labeled with such information.

Measuring the relevance of the retrieved context can also be done by using an LLM as a judge in a variety of ways. An example of an LLM-as-a-judge metric measuring the third evaluation question is the *Context Relevance* metric proposed by Es et al. [41]. For each test case, the LLM is instructed to identify sentences in the retrieved context that help answer

the question. The ratio of identified relevant sentences to the total number of sentences in the retrieved context can then be calculated, showing how much of the information that was surfaced by the retriever is relevant [23, 41]. Because this metric does not require references either for the context or the answer, it is considered a reference-free metric.

Faithfulness This metric dimension focuses on metrics evaluating how well a systems response is grounded in the retrieved context, regardless of the actual factuality of the response [23]. While testing this dimension necessitates access to the context and retriever, it is the generator that is the component under test and faithfulness metrics are therefore generator-scoped. The metrics can be used to gain confidence in how the generator component handles the data in the retrieved contexts, as well as for verifying any additional information not explicitly asked for in the question, which is unlikely to be included in a reference answer and so to be covered by correctness metrics [42]. We found the most common evaluation question in this dimension to be *To what extent is the generated response grounded in the retrieved context?*.

Faithfulness metrics can operate either on the lexical level, by comparing tokens of the answer to those of the reference context seeing how much overlap there is, or on the semantic level, comparing the semantic similarity of the generated answer and reference context [42]. An example of a purely lexical metric, which has been shown to align more closely with human judgment than other lexical metrics, is *K-Precision* [42]. This precision measurement calculates the ratio of tokens in the model response that are present in the retrieved context.

The semantic similarity can either be measured by vector embeddings, for example through BERTScore [42] or through using an LLM as a judge [42, 43, 35]. An LLM is either prompted to return a faithfulness score directly, given an answer and a reference context [42], or is tasked with identifying claims in the answer and context and comparing their overlap – either as pass/fail [43] or scored from 0 to 1 [35].

See Figure 4.1 for a categorization of faithfulness metrics according to their level of comparison and implementation.

Answer Relevance The metric dimension of *answer relevance* focuses on how well the generated answer directly addresses the question, regardless of the factuality of the response [23]. The metrics in this dimension are scoped to the generator component. We found the most common evaluation question in this dimension to be: *To what extent does the generated answer directly and completely address the user's question?*.

Due to the semantic nature of evaluating this dimension, lexical techniques are typically not employed. Common metrics in this dimension instead captures semantic alignment via embedding similarity. Es et al. [41] propose an implementation where for each question in the dataset, an LLM generates n alternative questions that may be answered by the generated answer. Each generated question q_i are then compared to the original question q via the cosine distance between their vector embeddings $sim(q, q_i)$ to measure their semantic similarity [41]. The average of these distances gives a measure of how well the semantic information of the question is carried over into the answer and therefore how well the question is answered. See Equation (4.3) for how the answer relevance score AR is calculated.

$$\mathrm{AR} = \frac{1}{n} \sum_{i=1}^{n} \mathrm{sim}(q, q_i) \tag{4.3}$$

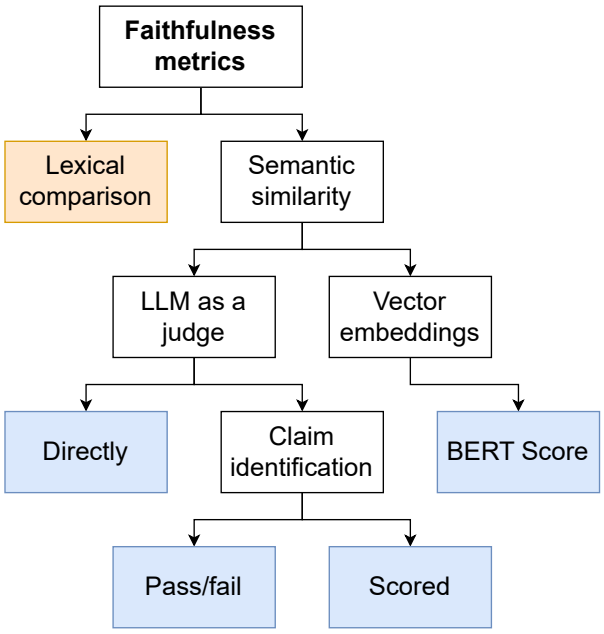

Figure 4.1: A subdivision of faithfulness metrics according to their level of comparison and collection method.

Correctness The metric dimension of *correctness* evaluates how well the generated answer matches the ground truth of the reference answer contained in the QAC tuple [23], and is therefore an end-to-end metric. We found the most common evaluation questions in this dimension to be: (1) *To what extent does the generated answer contain only information that aligns with the reference answer?*, and (2) *To what extent does the generated answer cover the information contained in the reference answer?*.

We consider correctness to be one of the most important metrics for gauging accuracy. In comparison with the other metrics, which are scoped to one component, correctness serves the role of ensuring that all components of a question answering system work well together. Therefore, a good correctness score serves to improve confidence that a system works as intended.

We categorize metrics within this dimension broadly in two categories: *lexical matching* and *semantic similarity*. Lexical matching metrics base their scores on token overlap between the generated and reference answers. This can be done with word or string matching, scoring the overlap and penalizing tokens that are not included in the ground truth. These metrics do not generally correlate well with human judgment due to what is known as the *strictness problem*, i.e., the observation that token comparison often fails to pass answers which humans deem correct due to simple token-level differences [44, 42]. If we consider the question answering system as a classifier, which classifies tokens as correct or incorrect by including them in the output or not (in a similar way as we do in Section 4.2.1), we get:

- A *true positive* as a correct output token,
- A *false positive* as an incorrect output token,

- A *false negative* as a token which would have been correct but was not outputted,
- A *true negative* as an incorrect token which was not outputted,

The classifier metrics of precision and recall, on which many of the metrics in this dimension are based, are defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{4.4}$$

Precision, therefore, measures the ratio of correct output tokens to the total number of output tokens in an answer, directly relating to our first evaluation question. Recall, on the other hand, does not penalize excessive output tokens, only counting the ratio of tokens present in the reference answer that are also present in the generated answer, relating to our second evaluation question. According to Adlakha et al. [42], recall is the lexical metric that shows the greatest correlation with human judgment.

The output of the *exact match* metric, on the other hand, is not a ratio, but instead a categorization of whether the generated answer is exactly equal to the reference answer. In very specialized domains, where the expected output of the system is just a single word or a few words, exact match can work well; in other contexts, exact match suffers even greater from the strictness problem than other lexical matching metrics [44]. It can be seen as a strict special case addressing both evaluation questions simultaneously.

Slightly more complex implementations of lexical metrics consider not only the tokens themselves, but the order in which they appear. Examples of these are BLEU and ROUGE, which use ***n***-gram matching [42]. They work on the same principles, comparing short sequences of tokens for overlap with the reference answer, but differ in the exact scoring formulas – for example, if they measure precision or recall, if short token sequences are penalized, or if they attempt to match on synonyms or on lexical stems [47]. BLEU emphasizes precision aspects, relating closer to the first evaluation questions, while ROUGE emphasizes recall aspects, relating closer to the second evaluation question.

Semantic similarity metrics, on the other hand, estimate semantic similarity between the generated and reference answer [42]. This can be done through leveraging contextual vector embeddings. Zhang et al. [45] propose the BERTScore metric, measuring precision, recall, and F1 scores between generated and reference answer using embeddings computed by a BERT language model. In this sense, BERTScore bridges the two previous evaluation perspectives, as its precision and recall components correspond to the first and second evaluation questions, respectively, while the F1 variant provides a balanced overall assessment.

Another way to measure semantic similarity is by using LLM-as-a-judge. By prompting a separate conversational agent with the question, reference answer, and the generated answer, it can be instructed to output a score assessing the correctness of the generated answer [42].

Citation Quality The metric dimension of *citation quality* focuses on how well a generated answer cites its sources [23]. Citation quality, and more generally knowledge attribution, is an important metric to keep track of since understanding the provenance of data increases user trust in the question answering system [19, 15]. We found the most common evaluation questions in this dimension to be: (1) *To what extent does the generated answer appropriately reference relevant sources?*, and (2) *To what extent are the cited sources actually used in supporting the answer content?*.

Knollmeyer et al. [23] divide the dimension of citation quality into two concrete measurements: *citation recall* and *citation precision*. Gao et al. [46] define citation recall as how well the generated response is supported by the cited context chunks, giving each statement in the generated response a score of 0 or 1 if it is supported or not, which addresses the first evaluation question. Citation precision, on the other hand, identifies citations which do not support any claims, penalizing irrelevant citations. This is done by assigning a score of 0 or 1 to each citation in the generated response, deeming if it is needed or not, which addresses the second evaluation question. Both of these metrics have shown good correlation with human judgment of citation quality [46].

4.2.3 Testing Methods

This section surveys existing approaches to evaluating generative AI question-answering systems, including dataset construction methods, established frameworks, and alternative evaluation and improvement strategies.

Combining Metrics Since most of the surveyed metrics apply to a single test case, scores across many test cases usually need to be aggregated. Typical is to combine scores by standard statistical methods, such as taking the mean or median. However, there are cases where other aggregation mechanisms may be preferable. An example is if certain test cases are more important than others, such as being more central to system performance or critical in other ways. The mean may in this case be combined with another overall aggregate, such as the proportion of test cases falling under a threshold score, or differentiating test suites and reporting them separately.

Different metrics that capture the same behavioral aspects can be combined for better evaluation reliability. The mtRAG [48] evaluation framework scores RAG retriever performance by using three separate non-LLM metrics and taking the harmonic mean. For LLM-based metrics, they propose testing with several different LLM models and using the median between their respective scores.

Concrete Frameworks There exist more-or-less concrete frameworks that combine these ideas about metric selection and question dataset generation into a coherent whole. Some of these frameworks are primarily academic in nature and designed to test out new approaches to testing, but others can be applied in mature development environments. Frameworks also differ in how much customization is possible by the developer, such as fine-tuning applied metrics. In this work, we focus not on cataloging all available frameworks, but rather on highlighting those that stand out through unique features or approaches.

One of the most widely used frameworks, as of the writing of this thesis, is *Retrieval-Augmented Generation Assessment* (Ragas) [49]. Ragas was introduced by Es et al. [41] in 2023, and has since evolved into an open-source evaluation framework. Ragas has built-in integrations with many of the most common AI integration frameworks in the field, including LangChain and LlamaIndex [50]. Ragas includes support for all main dimensions of metrics covered in Section 4.2.1, which can be selected between to support evaluation of various question answering systems. Ragas also includes dataset generation functionality as discussed in Section 4.2.4. The main limitations of Ragas lie mostly in its heavy reliance on LLM-as-a-

judge approaches, which as mentioned before suffer problems with the test oracle problem and potential bias, and with little support for alternative, deterministic metrics.

Saad-Falcon et al. [34] propose the *Automated RAG Evaluation System* (ARES). As the name suggests, the main focus of ARES is on automating tedious parts of the testing process. Its main contribution is its concept of LLM judges, which are trained on three things: the context data, a human validation set of approximately 150 data points, and few-shot examples of domain-specific queries and answers. Using this data, ARES generates synthetic queries and their answers, filters them according to their quality, and uses them for training LLM judges. These fine-tuned judges, which are aware of domain-specific knowledge, can be used for evaluating a separate RAG system. The judges score generated answers among the three metric dimensions of context relevance, faithfulness, and answer relevance.

ARES is mainly limited in its reliance on generated datasets, which are shown to perform worse than human-created ones, as well as not supporting relevant metrics such as correctness. Additionally, the sole use of LLM-as-a-judge metrics may lower the trustworthiness of evaluation results due to concerns of limited human alignment and the possibility of bias. The framework nonetheless presents an interesting approach in exploring ways to fully automate system evaluation.

The majority of covered metrics and benchmarks are focused on single-turn questions and answers, leaving multi-turn evaluation (where previous conversation history is considered) largely overlooked. mtRAG is a framework developed by IBM Research specifically for evaluating multi-turn RAG conversations [48]. The mtRAG framework supports several metrics in the correctness dimension, one metric measuring faithfulness and one metric measuring answer relevance. Due to its recentness (released in January 2025), it is hard to verify how well mtRAG works in more real-world settings.

4.2.4 Creating Question Datasets

Brehme, Ströhle, and Breu [32] propose a categorization of question datasets according to what kind of answer is expected for each question and the mechanisms that are required to answer it appropriately. First, questions eliciting (1) *short answers* are those that can be answered by a single word, number, boolean, or a choice between alternatives. Questions which require (2) *long answers*, in contrast, require more detailed, logical, and well-reasoned explanations. More complex (3) *multi-hop questions* require information from several context chunks (as opposed to just one) to form the correct answer. Some datasets further include (4) *error-type questions* or (5) *misleading context* questions. These refer to QAC tuples that contain inaccuracies (such as an illogical premise or false information) in the question or reference context entity, respectively.

The construction of the question dataset is influenced by several factors. Brehme, Ströhle, and Breu [32] identify the following four key factors to consider when designing a question dataset. First, the *intended use case* plays a central role. Using domain-specific question datasets increases confidence that the system performs as expected within a particular application context [32]. The *chosen evaluation metrics* also impose requirements on the dataset. Certain metrics necessitate supplemental information in addition to the question and reference answer. For example, context relevance metrics require a reference context [32]. We summarize question dataset requirements for the surveyed metrics in Table 4.3. The dataset design may also aim to *elicit specific system behaviors*. Evaluating robustness with malformed or

Table 4.3: Summary of requirements on question dataset contents for each surveyed metric.

Metric	Dataset requirements
Context relevance	Question, Context
Faithfulness	Question
Answer relevance	Question
Correctness	Question, Answer
Citation quality	Question

erroneous inputs requires the inclusion of such questions, while testing noise sensitivity may require augmenting the context with deliberately misleading or irrelevant information [32]. Finally, the dataset must be constructed to ensure that the question answering system relies on retrieved information from its context base, rather than defaulting to its internal parametric knowledge. In certain applications, this can be done by restricting the set of questions to those which require information which is beyond the LLM model's training cutoff date [32].

The most general way of creating datasets is by manually creating question-answer tuples (optionally including a reference context). However, because of the various dataset requirements previously outlined, datasets can be very time-consuming to manually create [32]. One approach to solve this problem is to use *existing datasets*. Brehme, Ströhle, and Breu [32] summarize freely available datasets of question-answer tuples (some including a context reference) which can be used for evaluating RAG systems. Most datasets are general-purpose, such as Google's Natural Questions [51] which contains around 300 000 real queries issued to the Google search engine together with human-annotated reference answers. Open datasets vary in the type of data they contain, and therefore in which ways they may be used for evaluation. The most advanced datasets, such as HotpotQA [52], combine long and short answers, single and multi-hop questions, as well as purposefully including a certain amount of false information in the context to be able to test the system's sensitivity to noise. There are also datasets which are specialized to certain domains, such as legal [53], medicine [54], and energy [55].

However, as a question answering system gets more specific, the likelihood of finding a dataset of good quality that covers necessary domain-specific information is not very high. To address this difficulty, various techniques have been proposed for *automatic dataset generation*. Brehme, Ströhle, and Breu [32] propose using an LLM to generate QAC tuples for a given context, leveraging various filtering and special prompting techniques to increase accuracy. Ragas takes a different approach [56] using knowledge graphs, a graph of semantic relationships between entities [57]. The user specifies how the knowledge graph is created through defining custom transformers that split, extract, and link entities found in the context. An LLM can traverse the knowledge graph and generate appropriate questions, decide what context is needed for the answer, and even provide a reference answer. Personas, i.e., information about the end users and how they ask questions, can be defined to allow for variations in how different user categories ask questions, and what kind of information they seek.

Synthetic generation of datasets is a rapidly-evolving field and studies performed on the efficacy of such methods have been limited in size [58, 59]. Most approaches face difficulties

regarding the oracle problem [33] and the risk of introducing biases when the same LLM is used for generating the dataset as is used in the RAG system itself [32]. It is also difficult to ensure benchmark validity of synthetic dataset tuples as compared to human-generated data.

We would like to note, however, that not all applications of dataset generation require high-quality or accurate data. For example, when augmenting a question dataset with intentionally flawed or misleading questions to test a system's robustness to noise, perfect accuracy is not only unnecessary but even counterproductive. In such cases, using LLMs to generate question-answer pairs can be an effective strategy.

One way of side-stepping these issues, while still not relying entirely on manual creation of datasets, is by leveraging *metamorphic testing* (MT) [60]. This form of testing is centered on defining and using metamorphic relations, which describe how outputs of a system should relate when inputs are transformed in some way. While a general testing concept, Ji et al. [7] successfully make use of metamorphic relations in testing generative AI question answering systems and use them for validating generated test suites without a ground truth. In the case of question answering systems, MT is typically applied by mapping an existing question-answer tuple to a new one by building a new question and reference answer, transformed in some way while preserving overall semantics [7]. For example, given the correctness of an existing question-answer-pair **Q:** *Is it raining outside?* **A:** No, the correspondingly negated pair should also be valid: **Q:** *Is it not raining outside?* **A:** Yes. While failure of the relation to hold does not give any information about what individual test case failed, it does indicate that the system under test violates an expected invariant and exposes a potential fault (in this case, related to either how the system interprets weather-related questions or an inconsistency in retrieving the current weather). Testing with metamorphic relations is a form of property-based testing that does not require a ground truth, making it possible to avoid manually annotating question-answer tuples while augmenting an existing dataset with additional data points.

4.3 Interview and Workshop Findings

In this section, we present the key insights derived from our workshop and expert interviews regarding the application of our proposed taxonomy of metrics and methods for quality assurance in generative AI-based question answering systems. These findings provide valuable perspectives on how quality is assessed in practice, the challenges of selecting and interpreting relevant metrics, and how the continuous evaluation of system performance aligns with both user expectations and organizational goals. We split our results across the four testing activities outlined earlier: requirements specification, metrics selection, methods, and test case creation. Finally, we present some related thoughts that do not neatly fit into any one of these categories.

4.3.1 Requirements Specification

A key theme that emerged was the importance of grounding the testing process in business and user requirements, and the need for understanding what users value. Requirements for a question answering system can be collected in much the same way as other system requirements, for example by using focus groups. How the question answering system is intended to

be framed and presented to users is important when deciding on requirements. A recurrent opinion was that systems which intend to replace human contact elicit higher expectations of quality (and therefore stricter requirements) than systems which are designed to be used as a complementary tool. Participants relate this observation to the overall notion that there is a difference between critical and non-critical features.

Especially in the context of enterprise-grade systems and domain-specific knowledge bases, participants agree with regard to the need for accuracy while also emphasizing transparency and correct handling of sensitive information as key factors in securing user adoption of new and unproven generative AI-based features. Participants generally agree that user trust stems mainly from these factors, while promoting the importance of approaching the issue of user trust from a wider perspective including the entire software offering. This was expressed by one participant as users having a higher default level of trust in a question answering system if they consider other parts of the system to be robust and of high quality.

4.3.2 Metrics Selection

Participants generally express that our proposed taxonomy of metrics is sound and makes useful distinctions, albeit with some indicating a potential overlap between the dimensions of *faithfulness* and *correctness*. They viewed the main value of metrics as enabling a developer to easier detect improvements and regressions in a question answering system. One participant remarked that metric scores must never be considered a goal by themselves, but that their value lies in the help they provide a developer to understand the quality of a system.

In deciding which metrics to choose for a particular application, there was general agreement that the process must be based on two factors. *First*, that the metrics must be directly connected to concrete requirements. *Second*, that developers must understand how a particular metric is calculated. As one participant expressed it: "A metric is useless if you do not understand how the score was reached." *Correctness* was viewed as the metric which most accurately reflects overall behavior of the system in terms of accuracy. Actionability, i.e., how well decisions can be made based on scores from a metric, was also perceived as important. Here, one participant expressed that the most actionable feedback is whether the user liked the answer or not. However, good human alignment of a metric does not by itself guarantee that the metric will be useful.

4.3.3 Evaluation Methods

A common theme was that the proposed evaluation methods, including examples of concrete frameworks, showed promise. However, concerns were raised that they were currently too immature to be viewed as something more than an experimental endeavor. It was not regarded as viable to incorporate most testing methods in a continuous integration pipeline due to the non-determinism exhibited by most methods and metrics. Instead, suggestions were raised to treat these tests the same as other possibly flaky tests, such as performance tests. Testing question answering systems is viewed as more of a process enabling continuous improvement, rather than as a form of pre-deployment validation.

4.3.4 Test Case Creation

A sound testing dataset was regarded as a key factor in enabling metrics to give good insights into system behavior. Participants expressed that the question dataset must reflect real-world use and be tailored to the application in question. However, major concerns were raised as to the practicality of manual creation of test cases, which was viewed as very costly and hard to justify. Keeping the test suite up-to-date was viewed as another practical challenge. Alternatives, including LLM-generated test cases was dismissed due to concerns of quality and bias. Some participants expressed interest in the application of metamorphic testing techniques to allow for better scalability of the question dataset, but communicated uncertainty about the current lack of practical methods tailored to the application developer.

4.3.5 Other Thoughts

Participants noted that there are other ways of ensuring user trust and enabling improvement of a question answering system than rigorous testing. For example, one participant expressed that systems which learn from user interactions could ensure systems improve over time without manual intervention. Participants also expressed the need for being able to test other trustworthiness dimensions, such as handling malicious or inappropriate user requests and general robustness of the system.

4.4 Alternative Evaluation Methods

Manual testing can sometimes be a viable alternative to automated testing, particularly for safety-critical or high-stakes applications. Although it is neither highly scalable or cost-efficient, it can serve as an additional safeguard to ensure system reliability. In practice, many testing frameworks [48, 32, 41] benchmark their automated evaluations against manual testing, which is often regarded as the empirical gold standard for accuracy and for validating human alignment of metrics. Human evaluation is typically done by annotating the generated answer against the reference answer. Because assigning precise numerical scores can be challenging for human evaluators, the output of most human evaluation methods are categorical in nature and typically expressed through a limited set of labels such as *pass/fail* or a Likert scale [61].

Incorporating *user feedback* provides application developers with additional means of collecting system-level metrics [62]. Although not an evaluation method in the strict sense, it offers human-centered insights that either complement or validate other evaluation measures. Unlike other forms of evaluation, user feedback is generated by real users during real-world interactions, such as by letting users flag inappropriate or inaccurate responses. For example, ChatGPT allows users to signal satisfaction with an answer by interacting with a thumbs-up or thumbs-down button. Such feedback is typically categorical in form and is naturally closely aligned with human judgment. However, the resulting metrics may be limited in scope or subject to various forms of bias, and can not always be used for targeted improvements to the question answering system.

An interesting way to incorporate user feedback is to directly use it for improvement of the question answering system through *reinforcement learning through human feedback* (RLHF).

This can take many forms. One example is to generate two versions of a response and letting the user choose which one is better. Goodman [63] propose using a separate LLM that analyzes user satisfaction through identifying indicators in the chat history, such as the user having to rephrase the question or rebutting the response, and automatically modifying parameters such as the system prompt in turn. This demonstrates the potential of layered evaluation of question-answering systems where one LLM monitors another. Even if both models share the same weaknesses (e.g., hallucinations), separating the "conversation" role from the "evaluation" role gives the system a new perspective that can lead to improvement in identifying question answering system weaknesses. However, there are limitations to RLHF, such as ensuring sufficient quality of human feedback and avoiding biases [64].

Chapter 5

Practical Lessons for Evaluating Generative AI

The purpose of this chapter is to explore how the evaluation metrics and methods identified in the literature study can be applied in concrete real-world scenarios, and what practical challenges arise in doing so. This answers RQ3: *what pitfalls can occur when using existing methods to automate question answering system testing?* In order to do this, we first present a conceptual background built on the analysis from Chapter 4, along with a set of three propositions. Then, we turn to three case studies performed on question answering systems in real-world settings. We present and analyze these case studies, highlighting strengths and limitations of the evaluation processes, qualitatively comparing recurring practical evaluation aspects including potential pitfalls, and how the processes relate to our taxonomy of metrics and methods presented in Section 4.2.2. The chapter concludes with a synthesis of these findings in relation to the proposed propositions, assessing how they are supported or challenged by empirical observations from the case studies.

5.1 Conceptual Background

In this subsection, we present themes distilled from the literature reviewed in the previous chapter. These themes describe recurring considerations and challenges and serve as a conceptual background for analyzing the practical evaluation efforts in the three case studies. The themes discussed here are derived from the subset of literature in the previous chapter that in the concept matrix were mapped to the concepts of *automated test frameworks* or *automated evaluation*.

The level of automation desired depends on the requirements and specifics of the system in question. Evaluation performed before the initial deployment, called *pre-deployment evaluation*, focuses on verifying correct behavior of the system [65]. For evaluation performed after the deployment, called *post-deployment evaluation*, the focus shifts towards continuously ob-

serving the system for possible changes (*model drift*) with the goal of maintaining satisfactory performance over time [65]. Both of these activities can be performed using the methods and metrics as described in Chapter 4. Based on our synthesis of the reviewed literature, a recurring theme is the disproportionate focus on pre-deployment evaluation, while post-deployment evaluation is discussed more sparsely. As a result, concrete mechanisms for continuous, post-deployment quality assurance are rarely addressed in detail.

A recurring theme that emerges across the reviewed literature relates to the scalability of automated evaluation methods. Certain automation measures necessitate a considerable amount of manual setup before they can operate effectively. Among the methods described in the previous chapter, we argue that implementations utilizing ground-truth metrics are the least scalable. For each test case where these metrics are used, some sort of annotation to determine the ground-truth is needed, demanding manual labor [66]. How large this burden is depends on the type of metric. For example, most context relevance metrics only require annotating which context chunks are relevant, a list that does not always need to be exhaustive, while some lexical correctness metrics require several ground-truth answers to reduce the number of false negatives [67]. On the other end of the spectrum, reference-free metrics impose fewer requirements on the dataset since annotations are not needed at all. The manual labor can be alleviated through synthetic generation of the dataset. The synthetic generation can be fully automated or a hybrid approach can be adopted, where synthetic test cases are created by building on a snapshot of manually created ones.

5.1.1 Propositions

Taken together, these themes highlight recurring tensions and limitations in the reviewed approaches to automated evaluation. Using these themes as an analytical foundation, the case studies were examined and insights gained from this examination were used to formulate a set of propositions. The propositions are supported to varying extents by empirical observations, which are discussed in Section 5.5. The propositions are:

1. Evaluation approaches for question answering systems that emphasize high human alignment are commonly associated with limited scalability.

2. Existing evaluation frameworks for question answering systems frequently lack mechanisms for continuous, post-deployment quality assurance.

3. Existing evaluation implementations often cover only a subset of relevant evaluation dimensions.

5.2 UniAsk

UniAsk is a question answering system for employees at the European bank UniCredit, described in a case study by Bordino et al. [26]. Over time, the bank has built up an internal knowledge base of about 60 thousand internal documents on topics such as internal processes, banking applications, and information about the bank's software tools. Previously,

this information was indexed by a custom search engine limited to keyword searches. However, the bank has identified the need to support more complex queries which cannot be answered satisfactorily by the current system. In order to improve user satisfaction and reduce the number of support tickets raised by users unable to find relevant information, the bank decided to develop and deploy UniAsk, a RAG-based question answering system aiming to augment keyword searches with natural language queries and responses.

UniAsk consists of a traditional RAG architecture with a retriever and generator. Documents from the knowledge base are chunked and then indexed by semantic meaning in a vector database. Because the knowledge base is dynamic, modified documents are re-indexed every 15 minutes. Additional features of the UniAsk architecture include a content filter for malicious or unrelated queries, and a correctness guardrail that prevents the most apparent cases of hallucinations by comparing the semantic similarity of generated answers to the retrieved context and discarding irrelevant responses. UniAsk leverages testing and evaluation at two stages in the deployment process. First, before initial deployment, and then continuously after each system change (such as switching retrieval strategies or finetuning hyperparameters).

Initial Evaluation The initial evaluation consisted of four stages. *First*, an initial automated evaluation stage to compare performance against the earlier search engine. *Then*, three rounds of user testing. Four categories of metrics were used for evaluation: context relevance, faithfulness[1], correctness, and citation quality.

The automated testing was done through the use of around 2700 manually created question-context tuples. Because a goal of the new system was to be able to respond to both keyword questions and natural language questions, both query types were present in the dataset. The retriever component was tested against these tuples and evaluated automatically against metrics including Precision@k, Recall@k, and MRR, scoring improvements of up to 500% against the previous system. The UniAsk team also attempted to test the generator component according to faithfulness metrics, using LLM-as-a-judge methods to judge how well a generated answer sticks to the retrieved context. However, the team abandoned this method after manual inspection revealed that the scores were very poorly aligned with human judgment. Therefore, faithfulness testing was deferred to later user-testing stages.

After the initial automated evaluation proved that the system performed well, the *initial phase of user testing* was done by inviting subject-matter experts (SMEs) to test the system according to their domain knowledge. Data was collected both through a feedback form filled in by the user (consisting of both ratings and free-text comments), as well as through analyzing the trigger prevalence of the guardrails or content filter. During this phase, the SMEs were asked to create a total of 500 corner-case questions, i.e., those questions for which an incorrect answer is deemed unacceptable (as compared to simply undesirable for the rest of the questions). This differentiation between questions enabled separate monitoring of system performance according to use-case criticality.

The *second phase of user testing* involved 500 employees being asked to use the system in their daily work, while providing feedback similar to that collected from the SMEs in phase one. This feedback (from both phases) was collected both to improve evaluation criteria and for further improvement of the RAG system itself.

[1] Bordino et al. [26] use the term *groundedness* for this metric category.

The *third phase of user testing* consisted of selecting 210 representative questions collected from the first two phases to form a user acceptance test for deployment approval. These questions consisted of various query types, including irrelevant and erroneous queries, and variations in query formatting. Generated answers for these questions were then manually evaluated by the SMEs.

Because of the nature of the user testing, gathered feedback could not be categorized directly into one of the metric categories. Instead, the UniAsk team reviewed the feedback and linked it to a system behavior, which corresponds to a specific metric category. For example, if the user feedback regards an incorrectly cited reference, the user rating is counted against citation quality metrics.

Continuous Testing The user acceptance test, formulated in the last stage of testing, forms a dataset that can be used for ensuring that the system continues to perform well even after initial deployment. Since the case study focuses on pre-deployment concerns, continuous post-deployment testing is not discussed further.

Summary and Discussion The UniAsk team has leveraged many of the evaluation methods and techniques that have been discussed previously in this thesis. We note that automation has been confined to the context relevance metrics used in for evaluating the retriever component in the very first stage of pre-deployment testing, with all other stages consisting of considerable manual work (creating the question dataset, rating test outputs, and users leaving detailed feedback). We also note that the evaluation process is highly grounded in real-world use cases, both with regard to the question dataset and in the way in which user experiences are incorporated.

Notably, the use of LLM-as-a-judge evaluation methods was considered, but rejected as not being sufficiently aligned with human judgment. The case study does not discuss implications of the required manual workload with regard to scalability; however, because of the overall success and scale of the evaluation process, we assume that this challenge could be overcome satisfactorily. No other issues regarding the chosen evaluation process were specifically lifted in the case study. Overall, the process was regarded as successful in achieving its goal of ensuring the quality and performance of the system, while also demonstrating these outcomes to shareholders and end users.

5.3 Aichi Prefecture Administrative RAG

The Civil Aviation Administration Division of the Aichi Prefectural Government is the bureau responsible for civil aviation in the Japanese prefecture. Over time, it has amassed over 2 TB of administrative documents, the majority consisting of unstructured Microsoft Office files (Word, Powerpoint, and Excel). Especially employees who are new to the bureau experience difficulties in navigating this database, which has led to perceived inefficiencies. To remedy this, in 2024, Kawashima, Shiramatsu, and Mizumoto [68] performed a case study consisting of the development and deployment of an experimental RAG system for use by the bureau's employees. The RAG system follows the standard architecture of separate generator and retriever components, the initial version not including any form of guardrail or filters.

Evaluation Process As part of the case study, the authors performed a structured evaluation of the RAG system with two goals. *First*, to understand if the RAG system helps effectivize operations at the bureau. *Second*, to guide implementation choices, especially regarding context chunking strategies.

The RAG system was evaluated according to two main metric categories: retrieval accuracy and answer accuracy. *For retrieval accuracy*, falling under *context relevance* in our taxonomy, they used the context metrics of precision, recall, and the F-measure (the harmonic mean of precision and recall). These metrics were collected on a question dataset consisting of only seven question-context tuples. In addition to the question and reference context, each tuple contained a separate string defining the user's situation (specifying the user's intended goal when asking the question) which was injected as part of the system prompt. The questions were divided into two categories depending on if they require short or long answers (see Section 4.2.4 for the previous discussion on this categorization). The retrieved contexts from the system were graded manually against the question dataset.

For *answer accuracy*, which falls under *correctness* according to our taxonomy, the authors devised a series of experiments requiring end-user participation. Three individuals with differing roles and experience levels were instructed to use the system to find answers to predefined questions, requiring either short or long answers. After each test, the participant was asked to rate their confidence in each answer using a 7-point scale, after which the answers were manually graded against a reference answer on the same 7-point scale according to *correctness*[2] and *completeness*. The time taken for the participant to find a satisfactory answer to the question was also recorded. All in all, four metrics were therefore used to gauge answer accuracy.

Summary and Discussion No part of the evaluation process of this RAG system was automated. Additionally, the low number of question dataset tuples (seven) and end-user participants (three) may threaten the statistical power of the evaluation. The authors note that this is a preliminary study and that future research may address these problems, but it is unclear how well the chosen evaluation process scales to such settings.

The authors use a limited number of metrics, focusing primarily on context relevance and correctness. The human alignment of the evaluation results is good, because of the use of specific business metrics (e.g., time taken to complete a task) and because the questions participants asked the system during the second part of the evaluation were not predefined but instead allowed to vary naturally according to participant personas (in this case, role and experience). The authors find that the scored confidence in the correctness of generated answers is highly correlated to actual correctness in the case of experienced participants, while there is no such correlation at all in the case of inexperienced participants. This highlights the importance of evaluators possessing sufficient domain expertise, as plausible-but-ultimately-wrong answers may otherwise be graded incorrectly during a manual evaluation process.

5.4 Shiga University Hospital RAG

The Shiga University of Medical Science Hospital is located in Kusatsu, Japan. Practitioners of nuclear medicine at its radiology department make daily use of internal manuals and doc-

[2]Kawashima, Shiramatsu, and Mizumoto [68] use the term *accuracy*.

uments to support their work. In 2025, Fukui et al. [69] performed a case study assessing the usability and accuracy of a RAG-based question answering system indexed on this data. The documents, all in Japanese, are chunked and indexed in a vector database, with a retriever component fetching relevant documents using either vector or keyword searches.

Evaluation Process The system was evaluated to answer two questions. *First*, whether a RAG system is accurate and useful for medical practitioners at the hospital. *Second*, to guide implementation choices including choice of retrieval mechanism and generator model.

In the *first stage of the evaluation*, a question dataset consisting of 100 question-reference-answer pairs was created by expert radiological technologists and medical physicists. The questions were then reviewed by two separate nuclear medicine technologists to ensure clinical relevance. The questions were additionally tagged according to type; categories included the amount of needed reference documents, and whether the question required simple factual recall or more advanced, analytical reasoning.

In the *second stage of evaluation*, three experts with various medical backgrounds in the field of nuclear medicine were tasked with manually evaluating responses generated by the various RAG system configurations. Two overall metric categories were used, *correctness* and *context relevance*, with participants ranking retrieved references and generated answers against the ground-truths on a four-point Likert scale.

In the *final stage of the evaluation*, two automated evaluation metrics from the RAGAS framework were selected: *correctness* and *context recall*. In addition to these, three other metrics were used: *semantic similarity* through vector embeddings of the generated and reference answers, *ROUGE*, and *Levenshtein distance* (the edit distance between character strings showing structural similarity). Scores according to these metrics were collected on the same question dataset as during the manual evaluation. This final stage of the process was the only evaluation that was performed automatically.

Scores collected from each of the evaluation stages were compared across several dimensions. *First*, how well the manual and automated scores agreed. *Second*, how implementation differences affected the scores. *Third*, how the scores differed depending on the type of question. This enabled Fukui et al. [69] to measure human alignment of the chosen metrics, as well as how different configurations of the system affected its performance in different use cases.

Summary and Discussion The authors find that the RAGAS-based metrics were more closely aligned to human judgment as compared to the lexical metrics. Following examination of this discrepancy, they find that the traditional metrics disproportionately penalize cases where the system responds in a correct way, only differently formulated than the expected reference answer. This is a manifestation of the *strictness problem* as discussed in Section 4.2.2. The authors note, however, that the better human alignment of RAGAS answers applies mostly when considered across the entire question dataset; RAGAS still shows human alignment difficulties on the level of single test cases.

Interestingly, the authors note that the inter-rater agreement of human judges is often not as high as expected, considering their similar backgrounds. They theorize that this is because of different expectations and biases during the evaluation process. This emphasizes the need for alignment between stakeholders during the evaluation process. However, because of the limited number of expert participants (three), the authors are careful to draw any wider

conclusions from this. The authors further note that, while the chosen metrics are useful for assessing whether the system performs as expected, they often do not offer useful diagnostic insight about what led to a particular score. Instead, qualitative *failure case analysis* of specific test cases proves to be an important companion to the quantitative metrics in gaining wider insights about system behavior.

The authors conclude the case study by stating that while the chosen evaluation process is useful for identifying erroneous system behaviors, it is not by itself enough for ensuring sufficient quality and trust in a critical application such as medicine. The authors, therefore, outline areas of future research consisting of various explainable AI techniques, such as reliable citations for verifying the provenance of information in generated answers. We discuss this in further detail in Section 4.2.2.

Table 5.1: A summary of surveyed case studies according to setting and goals, as well as question dataset characteristics including what types of questions are used and how they are created.

Study	UniAsk [26]	Aichi [68]	Shiga [69]
Setting	Banking	Government	Medical
Maturity	Production	Experimental	Experimental
Goals	Comparison against previous system and guiding implementation choices	Assessing suitability and guiding implementation choices	Assessing suitability and guiding implementation choices
Question dataset size	2700	7	100
Question types	Short-form, long-form, erroneous, persona-differentiated	Short-form, long-form	Factual recall and analytic questions
Question dataset creation	Manual	Manual	Manual

Table 5.2: A summary of metrics used in each of the three surveyed case studies, including metric dimension, component scope, and degree of automation of the collection process.

Study	Dimension	Metric	Scope	Automation
UniAsk [26]	Context relevance	Precision@k	Retriever	Automated
	Context relevance	Recall@k	Retriever	Automated
	Context relevance	MRR	Retriever	Automated
	Faithfulness	*free-text user feedback*	Generator	Manual
	Citation quality	*free-text user feedback*	Generator	Manual
	Correctness	*free-text user feedback*	End-to-end	Manual
Aichi [68]	Context relevance	Precision	Retriever	Manual
	Context relevance	Recall	Retriever	Manual
	Context relevance	F1	Retriever	Manual
	Correctness	Likert	End-to-end	Manual
	Business	Time for task	End-to-end	Manual
Shiga [69]	Context relevance	RAGAS	Retriever	Automated
	Correctness	RAGAS	End-to-end	Automated
	Correctness	Likert	End-to-end	Manual
	Correctness	Semantic similarity	End-to-end	Automated
	Correctness	ROUGE	End-to-end	Automated
	Correctness	Levenshtein distance	End-to-end	Automated

5.5 Summary of Findings (RQ3)

The insights gained from these cases provide a foundation for evaluating each of the propositions defined in Section 5.1.1, highlighting patterns, limitations and implications. In the following section we discuss each proposition in turn, drawing on the evidence we presented to assess its validity and relevance. We present summaries of key aspects of each surveyed system in Tables 5.1 and 5.2.

Proposition 1: Evaluation approaches for question answering systems that emphasize high human alignment are commonly associated with limited scalability Across all three case studies, a persistent trade-off between scalability and human alignment was evident. UniAsk deliberately rejected automated evaluation methods such as LLM-as-a-judge due to poor alignment with expert judgment, instead opting for manual review despite the additional workload. Similarly, the Shiga Hospital's use of RAGAS demonstrated moderate correlation with human assessments but also produced inconsistent results, depending on the question type. In the Aichi case study, the authors conducted exclusively manual evaluation, foregoing any automation. These decisions reflect deliberate prioritizations of trust and reliability over scalability in the assessment process, ensuring the results were credible and accurate.

Scalability issues in UniAsk stem from the requirements imposed on the question dataset, which necessitates the annotation of both reference contexts and reference answers. While not specifically mentioned as such in the study, we regard the creation and maintenance of 2700 QAC tuples as inherently unscalable, particularly given the required level of subject-matter expertise. Nevertheless, the scalability limitations of the UniAsk evaluation process did not constitute a sufficient obstacle to hinder the project's execution, as evidenced by its successful completion.

We note that no case addressed the question of automating question dataset creation. Since all case studies place great weight on the importance of having vetted experts create the dataset, we speculate that this omission is deliberate and due to perceived quality issues with regard to correctness and bias of the automated approach. These concerns are particularly relevant in the complex, and in some cases critical, subject domains covered by the case studies.

Proposition 2: Existing evaluation frameworks for question answering systems frequently lack mechanisms for continuous, post-deployment quality assurance Another recurring limitation across the case studies concerns the absence of mechanisms for continuous or post-deployment evaluation. Both the Shiga Hospital RAG and Aichi RAG conducted evaluation activities as one-time exercises during system development, without integrating ongoing monitoring or regression testing after deployment.

Among the three, UniAsk was the only study that explicitly discussed the importance of post-deployment evaluation and the need to monitor system performance over time. However, this part of the evaluation process remained conceptual and was not implemented in practice. The Shiga Hospital RAG, in contrast, did not explicitly address post-deployment monitoring, but the inclusion of automated evaluation metrics suggests potential for such integration in future iterations. These metrics could, in principle, support periodic evaluation

with minimal manual intervention, but this capability was not realized within the reported implementation.

Proposition 3: Existing evaluation implementations often cover only a subset of relevant evaluation dimensions All three case studies demonstrate an imbalance in the dimensions evaluated, although to different degrees. Context relevance and correctness metrics were employed in all three case studies. Furthermore, the UniAsk system incorporates dimensions of faithfulness and citation quality, assessed qualitatively through a feedback system. It is worth noting that answer relevance metrics were not used in any of the case studies surveyed. To summarize, UniAsk's evaluation metrics covered four out of the five dimension categories outlined in our taxonomy, while the remaining two case studies only covered two.

In all surveyed case studies, the chosen evaluation processes were found to be useful for highlighting system behavior and identifying bugs by the authors. In the case of UniAsk, successful completion of the evaluation process was considered sufficient for general deployment. For Shiga Hospital RAG, the authors noted that more testing, in addition to other ways of ensuring trust and quality, was needed before the system could be developed further and deployed.

Chapter 6
Discussion

This chapter is organized as follows. First, we discuss why the evaluation of RAG-based question answering systems is typically narrow in scope despite the diversity of available evaluation methods, exploring this from a technical and an organizational standpoint. Then, we discuss how well our results translate to (1) testing other system architectures and (2) behavioral aspects other than accuracy. Finally, we end the chapter with a discussion of how our results affect practitioners in their work, as well as as briefly exploring possible future research in the field.

6.1 On the Lack of Testing

Throughout this thesis, practitioners generally express the need for being able to test RAG-based question answering systems and verify them against accuracy requirements. However, even in cases where such systems are in fact tested, we see that the testing is narrow in scope with many evaluation efforts being limited to just one or two categories of accuracy metrics, or that much testing is still performed entirely manually and ad-hoc. In our taxonomy, as defined in Section 4.2.2, we have identified many metrics and methods that can be applied in such evaluation processes, which leads us to the question: why are these methods not applied more systematically and thoroughly?

Technical Concerns As has been discussed, testing RAG-based question answering systems is fundamentally different than traditional software testing with respect to determinism, predictability of input-output behavior, interpretability of test results, and the presence of reliable test oracles.

A sought-after aspect in all evaluation processes covered so far is that evaluation results are well-aligned with human judgment. As discussed, high human alignment is achievable in typical evaluation processes, but is associated with high levels of human involvement either through the usage of manual evaluation in part or full, or when creating and curating

test datasets for use by automated methods. Manual involvement is time-consuming and often requires domain expert involvement, which both makes it expensive and ill-suited for continuous monitoring use cases and can be hard to justify.

At the same time, fully automated evaluation metrics introduce their own limitations. Practitioners emphasized that metric scores are only useful if developers understand what drives them, something seldom the case with, for example, LLM-as-a-judge metrics. Consequently, evaluation approaches with limited alignment to human judgment provide restricted support for quality assurance. This issue is further compounded by the low interpretability of RAG-based question answering systems in general.

Furthermore, while there exist many methods and metrics, few of them are suitable or aimed for mature production environments, many of them being purely academic in nature. And while there are concrete frameworks that package these methods and metrics, lack of confidence in these frameworks with regard to interpretability of test results and human alignment is an obstacle for wider adoption of even these production-focused efforts. Many practitioners view evaluation of RAG systems as immature and that it does not give enough value to the developer in contrast with the required effort.

Organizational Concerns Because RAG-based question answering systems are a new and rapidly evolving technology, it is hard for end users and developers alike to accurately articulate and specify requirements put on such systems. Requirements may also differ depending on the criticality and use case of a given system. Therefore, accurate and systematic requirements specification is an important first step when designing evaluation processes for a RAG-based question answering system. It is possible that unclear requirements preclude the application of sophisticated evaluation processes (i.e., if you do not know what to test you cannot test it). The high level of manual involvement required for certain activities, such as the creation of test datasets, is another concern. Systematically evaluating RAG systems may be hard to justify because of this cost, especially if there are organizational concerns of the value testing brings to the team or product.

Taken together, these two factors suggest that the lack of systematic testing is not primarily due to a lack of interest, but due to a combination of (1) difficult technical properties of generative systems and (2) poor cost/benefit alignment for many real deployments. This also helps explain why many teams still rely on manual spot-checking or ad-hoc evaluation despite known limitations.

6.2 Testing Other System Architectures

We believe that several of the core testing challenges and evaluation ideas generalize beyond RAG to other question answering system architectures. The oracle problem, nondeterminism, and difficulty of determining correctness for open-ended natural language outputs apply to essentially all LLM-based question answering systems. In principle, this means that the testing challenges that we have identified are just as pressing for non-RAG, LLM-based systems.

Correctness, which is an end-to-end metric applies just as well to question answering systems with pure parametric memory (i.e., those without a separate retriever component),

because it is not dependent on the particular architecture of the system under test. The generator-specific metrics (*faithfulness* and *answer relevance*) can also be applied if you consider the LLM to be the generator. It is unclear to what extent *citation quality* is relevant for purely LLM-based question answering systems, because despite being a generator-scoped metric, parametric memory is typically not able to give accurate citations due to how the model is trained.

On the other hand, *context relevance* metrics are not useful for purely LLM-based question answering systems due to the absence of the retriever component. While the dominating implementation for a retriever component is through a vector store, we believe that context relevance metrics are still relevant for other implementations, including keyword search, knowledge graphs, or agentic retrieval. We also believe that our defined evaluation process (with the four steps of *requirements gathering*, *metrics selection*, *choosing testing methods*, and *dataset generation*) is widely applicable, albeit with the specific activities needing to be modified for each type of system.

6.3 Testing Other Behavioral Aspects

Although this thesis focused mainly on the trustworthiness dimension of *accuracy*, several of the insights gained from the literature review and expert inquiries extend beyond this scope. Of particular note are the identified trade-offs and limitations, many of which are not tied to to the specific dimension of accuracy but to the way evaluation is conducted in practice. In particular, the trade-offs between scalability and human-aligned evaluation, as well as the limited support for post-deployment evaluation, is likely to be relevant for the trustworthiness dimensions of, for example, *robustness* and *fairness*. The findings may also inform the evaluation of other system aspects, such as creativity and stylistic variation. These aspects rely heavily, at least in our view, on human judgment and are therefore likely to encounter many of the same limitations as discussed in this thesis.

In short, rather than being able to directly use specific metrics discussed in this thesis to test other aspects of system behavior, the usability lies in the limitations and principles discussed which may transfer to other areas. However, extending the results to new dimensions requires careful consideration, since they may not translate directly.

6.4 Implications

The findings point to several ways in which practitioners can draw on the results of this thesis when designing evaluation processes for generative AI-based question answering systems. The extended taxonomy of metrics and methods can serve as a reference when constructing evaluation pipelines, helping practitioners reason about coverage across different dimensions of accuracy and about which aspects of system behavior are being assessed.

The insights concerning the relationship between scalability and human alignment can support decisions regarding when automation is appropriate, and when human judgment remains necessary. In addition, the practical pitfalls identified can inform practitioners by clearly laying out common limitations and challenges. Awareness of these pitfalls may help teams set realistic goals and avoid common mistakes.

Taken together, the implications of this thesis lie in providing a structured analytical foundation that is useful in the reflection, design, and evolution of evaluation processes in ways that are consistent with the constraints and characteristics of modern RAG-based question answering systems. Beyond practice, the results also highlight areas where existing evaluation frameworks and tooling fall short, which may inform future research on scalable and human-aligned evaluation methods.

6.4.1 Recommendations for Practitioners

To complement the high-level perspective offered before, we can shift the focus more towards concrete and actionable guidance aimed directly at practitioners. These recommendations highlight practical considerations that can inform testers in their decision-making. These practical recommendations, stated as practical tips, are as follows:

1. **Treat evaluation as a continuous process, rather than as a one-off activity.** The behavior of a question answering system may change over time due to implementation changes or updates to the knowledge base. Continuous evaluation is therefore necessary to detect regressions and ensure that accuracy is sustained post deployment. If the system is critical enough to warrant testing, how can testing only at launch be deemed enough?

2. **Start with end-to-end testing, then evaluate components separately.** This is because end-to-end testing generally captures the overall performance of a system, while component-level testing enables reasoning about component-specific implementation choices in order to make the overall performance better.

3. **Expect and manage the possible trade-off between scalability and human-alignment early in the test design process.** Understanding this trade-off early enables informed decisions about which to prioritize, which impacts the design of subsequent testing activities.

4. **Invest in a qualitative, representative, and comprehensive test dataset, treating it as a core product asset.** The quality and scope of the test dataset is a key success factor for being able to draw conclusions and gain actionable insights from the testing process.

5. **Use metrics as a way to understand system behavior and avoid using them as opaque goals.** Focusing too narrowly on improving metrics and using them as goals in themselves risks practitioners missing the point of why metrics are used. A failing metric should be treated as a symptom of a defect in the system, and should be used as a tool to better understand and improve the underlying implementation.

6.5 Ethical and Societal Considerations

Beyond technical considerations, this thesis also has ethical and societal implications. Generative question answering systems are increasingly deployed at scale and can have a real impact on people's lives. This is particularly evident in high-stakes domains, where incorrect or misleading answers may cause direct harm. Viewed from this lens, testing is not just a technical

concern but an ethical responsibility. While risks are most apparent in safety-critical applications such as healthcare, general-purpose question answering systems can also lead to harm by fostering misplaced trust or spreading misinformation.

The findings and recommendations of this thesis aims to support more transparent and accountable testing practices, by helping practitioners identify limitations. Furthermore, the results highlight the importance of human oversight, as automated evaluation alone cannot fully capture user experience or societal impact. Taken together, these considerations emphasize the role of rigorous testing in the responsible development and deployment of generative AI systems.

Chapter 7
Conclusions

This thesis set out to understand how generative AI-based question answering systems built on RAG architectures can be evaluated in a structured and reliable way. Through a combination of a literature review and expert inquiries, the work identifies key challenges of testing such systems, catalogs the methods currently used in practice, and highlights practical pitfalls that arise during the application of these methods.

Our main contribution is an extended taxonomy of metrics and methods that organizes the fragmented landscape of existing approaches to accuracy testing into five coherent dimensions: *correctness*, *answer relevance*, *context relevance*, *faithfulness*, and *citation quality*, answering RQ2. It provides a concrete framework that practitioners and researchers can use to design more complete and performative evaluation pipelines.

In addition to this taxonomy, the thesis identifies fundamental challenges that complicate the evaluation of generative question answering systems: *unpredictable output behavior*, *the oracle problem*, *limited interpretability*, and *the added complexity of a separate retriever component*, answering RQ1. Together, these characteristics explain why traditional software testing techniques fall short and why new evaluation strategies are needed for modern question answering architectures.

The examination of three real-world cases revealed recurring pitfalls in how evaluation is carried out today, answering RQ3, with the most pressing pitfall being the difficulty of balancing automation with good human alignment. We present three propositions based on these case studies: that (1) *evaluation approaches emphasizing high human alignment are commonly associated with limited scalability*, that (2) *existing evaluation frameworks frequently lack mechanisms for continuous quality assurance*, and that (3) *existing evaluation implementations often cover only a subset of relevant evaluation dimensions*.

Overall, we provide a structured conceptual model for understanding evaluation methods, and an empirical analysis of the applicability of these methods in practice. These results provide a structured basis for understanding and evaluating RAG-based question answering systems, and are complemented by five concrete recommendations that translate the findings into actionable guidance for practitioners. Taken as a whole, the contributions of this thesis

support more systematic and reliable evaluation of RAG-based question answering systems.

Future research can build directly on the three propositions identified in this thesis. For Proposition 1, given the observed trade-off between scalability and human alignment, further studies should investigate evaluation approaches that retain strong human grounding while remaining feasible at scale. Regarding Proposition 2, future work can focus on designing evaluation pipelines that support ongoing monitoring and regression testing in production environments. For Proposition 3, further research is needed to study how different combinations of evaluation dimensions interact, and how trade-offs between coverage, cost, and reliability can be managed.

References

[1] Zhiyu An et al. *Golden-Retriever: High-Fidelity Agentic Retrieval Augmented Generation for Industrial Knowledge Base*. July 20, 2024. DOI: 10.48550/arXiv.2408.00798. arXiv: 2408.00798 [cs]. Pre-published.

[2] Patrick Lewis et al. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 9459–9474.

[3] Ziwei Ji et al. "Survey of Hallucination in Natural Language Generation". In: *ACM Comput. Surv.* 55.12 (Mar. 3, 2023), 248:1–248:38. ISSN: 0360-0300. DOI: 10.1145/3571730.

[4] Negar Maleki, Balaji Padmanabhan, and Kaushik Dutta. "AI Hallucinations: A Misnomer Worth Clarifying". In: *2024 IEEE Conference on Artificial Intelligence (CAI)*. 2024 IEEE Conference on Artificial Intelligence (CAI). June 2024, pp. 133–138. DOI: 10.1109/CAI59869.2024.00033.

[5] Long Ouyang et al. *Training Language Models to Follow Instructions with Human Feedback*. Mar. 4, 2022. DOI: 10.48550/arXiv.2203.02155. arXiv: 2203.02155 [cs]. Pre-published.

[6] Changchang Zeng et al. *A Survey on Machine Reading Comprehension: Tasks, Evaluation Metrics and Benchmark Datasets*. Oct. 21, 2020. DOI: 10.48550/arXiv.2006.11880. arXiv: 2006.11880 [cs]. Pre-published.

[7] Pin Ji et al. "NLPLego: Assembling Test Generation for Natural Language Processing Applications". In: *ACM Trans. Softw. Eng. Methodol.* 34.2 (Jan. 21, 2025), 49:1–49:36. ISSN: 1049-331X. DOI: 10.1145/3691631.

[8] *What Are Large Language Models (LLMs)?* IBM. Nov. 2, 2023. URL: https://www.ibm.com/think/topics/large-language-models (visited on 2025-08-13).

[9] *Introducing ChatGPT*. OpenAI. Nov. 30, 2022. URL: https://openai.com/index/chatgpt/ (visited on 2025-08-13).

[10] Legora. *Collaborative AI for Lawyers*. 2025. URL: https://legora.com/ (visited on 2025-09-09).

[11] Bolanle Ojokoh et al. “A Review of Question Answering Systems”. In: *Journal of Web Engineering* 17.8 (2019), pp. 717–758. ISSN: 1540-9589. DOI: 10.13052/jwe1540-9589.1785.

[12] Amer Farea and Frank Emmert-Streib. “Understanding Question-Answering Systems: Evolution, Applications, Trends, and Challenges”. In: *Engineering Applications of Artificial Intelligence* 156 (Sept. 2025), p. 110997. ISSN: 09521976. DOI: 10.1016/j.engappai.2025.110997.

[13] *New Embedding Models and API Updates*. Mar. 13, 2024. URL: https://openai.com/index/new-embedding-models-and-api-updates/ (visited on 2025-08-29).

[14] TigerData. *A Guide to Cosine Similarity*. Oct. 16, 2024. URL: https://www.tigerdata.com/learn/understanding-cosine-similarity (visited on 2025-08-29).

[15] Rudresh Dwivedi et al. “Explainable AI (XAI): Core Ideas, Techniques, and Solutions”. In: *ACM Computing Surveys* 55.9 (Sept. 30, 2023), pp. 1–33. ISSN: 0360-0300. DOI: 10.1145/3561048.

[16] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. “A Primer in BERTology: What We Know About How BERT Works”. In: *Transactions of the Association for Computational Linguistics* 8 (Dec. 2020), pp. 842–866. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00349.

[17] Simon Knollmeyer, Oğuz Caymazer, and Daniel Grossmann. “Document GraphRAG: Knowledge Graph Enhanced Retrieval Augmented Generation for Document Question Answering Within the Manufacturing Domain”. In: *Electronics* 14.11 (May 22, 2025), p. 2102. ISSN: 2079-9292. DOI: 10.3390/electronics14112102.

[18] Dongyang Li et al. *On the Role of Long-tail Knowledge in Retrieval Augmented Large Language Models*. June 24, 2024. DOI: 10.48550/arXiv.2406.16367. arXiv: 2406.16367 [cs]. Pre-published.

[19] Yujia Zhou et al. *Trustworthiness in Retrieval-Augmented Generation Systems: A Survey*. Sept. 16, 2024. DOI: 10.48550/arXiv.2409.10102. arXiv: 2409.10102 [cs]. Pre-published.

[20] Microsoft. *Text Search with Semantic Kernel*. Nov. 15, 2024. URL: https://learn.microsoft.com/en-us/semantic-kernel/concepts/text-search/ (visited on 2025-09-02).

[21] Microsoft. *Retrieve Data from Plugins for RAG*. June 25, 2024. URL: https://learn.microsoft.com/en-us/semantic-kernel/concepts/plugins/using-data-retrieval-functions-for-rag (visited on 2025-06-07).

[22] Ambuje Gupta et al. “REFINE on Scarce Data: Retrieval Enhancement Through Fine-Tuning via Model Fusion of Embedding Models”. In: *Lect. Notes Comput. Sci.* Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Vol. 15442. Springer Science and Business Media Deutschland GmbH, 2025, pp. 73–85. ISBN: 978-981-96-0348-0. DOI: 10.1007/978-981-96-0348-0_6.

[23] Simon Knollmeyer et al. “Benchmarking of Retrieval Augmented Generation: A Comprehensive Systematic Literature Review on Evaluation Dimensions, Evaluation Metrics and Datasets:” in: *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. 16th International Conference on Knowledge Management and Information Systems. Porto, Portugal: SCITEPRESS - Science and Technology Publications, 2024, pp. 137–148. ISBN: 978-989-758-716-0. DOI: 10.5220/0013065700003838.

[24] Rachel L. Draelos et al. *Large Language Models Provide Unsafe Answers to Patient-Posed Medical Questions*. Aug. 4, 2025. DOI: 10.48550/arXiv.2507.18905. arXiv: 2507.18905 [cs]. Pre-published.

[25] Aline Valente et al. “Analysis of Academic Databases for Literature Review in the Computer Science Education Field”. In: *2022 IEEE Frontiers in Education Conference (FIE)*. 2022 IEEE Frontiers in Education Conference (FIE). Oct. 2022, pp. 1–7. DOI: 10.1109/FIE56618.2022.9962393.

[26] Ilaria Bordino et al. “UniAsk: AI-powered Search for Banking Knowledge Bases”. In: *Adv. Database Technol. - EDBT*. Advances in Database Technology - EDBT. Vol. 28. 3. OpenProceedings.org, 2025, pp. 1057–1065. ISBN: 978-3-89318-098-1. DOI: 10.48786/edbt.2025.89.

[27] Henrik Vester and Jacob Bentzer. *Concept Matrix for Quality Assurance in Generative AI*. Zenodo, Jan. 6, 2026. DOI: 10.5281/ZENODO.18164715.

[28] Apostolos Ampatzoglou et al. “Guidelines for Managing Threats to Validity of Secondary Studies in Software Engineering”. In: *Contemporary Empirical Methods in Software Engineering*. Ed. by Michael Felderer and Guilherme Horta Travassos. Cham: Springer International Publishing, 2020, pp. 415–441. ISBN: 978-3-030-32489-6. DOI: 10.1007/978-3-030-32489-6_15.

[29] Taguette. *Taguette, the Free and Open-Source Qualitative Data Analysis Tool*. URL: https://www.taguette.org/ (visited on 2025-10-24).

[30] Aldeida Aleti. *Software Testing of Generative AI Systems: Challenges and Opportunities*. Sept. 11, 2023. DOI: 10.48550/arXiv.2309.03554. arXiv: 2309.03554 [cs]. Pre-published.

[31] Shuyin Ouyang et al. “An Empirical Study of the Non-Determinism of ChatGPT in Code Generation”. In: *ACM Trans. Softw. Eng. Methodol.* 34.2 (Jan. 22, 2025), 42:1–42:28. ISSN: 1049-331X. DOI: 10.1145/3697010.

[32] Lorenz Brehme, Thomas Ströhle, and Ruth Breu. “Can LLMs Be Trusted for Evaluating RAG Systems? A Survey of Methods and Datasets”. In: *2025 IEEE Swiss Conference on Data Science (SDS)*. June 26, 2025, pp. 16–23. DOI: 10.1109/SDS66131.2025.00010. arXiv: 2504.20119 [cs].

[33] Earl T. Barr et al. “The Oracle Problem in Software Testing: A Survey”. In: *IEEE Transactions on Software Engineering* 41.5 (May 1, 2015), pp. 507–525. DOI: 10.1109/TSE.2014.2372785.

[34] Jon Saad-Falcon et al. *ARES: An Automated Evaluation Framework for Retrieval-Augmented Generation Systems*. Mar. 31, 2024. DOI: 10.48550/arXiv.2311.09476. arXiv: 2311.09476 [cs]. Pre-published.

[35] Selvan Sunitha Ravi et al. *Lynx: An Open Source Hallucination Evaluation Model*. July 22, 2024. DOI: 10.48550/arXiv.2407.08488. arXiv: 2407.08488 [cs]. Pre-published.

[36] Hao Yu et al. "Evaluation of Retrieval-Augmented Generation: A Survey". In: *Big Data*. Ed. by Wenwu Zhu et al. Singapore: Springer Nature, 2025, pp. 102–120. ISBN: 978-981-96-1024-2. DOI: 10.1007/978-981-96-1024-2_8.

[37] Victor R Basili, Gianluigi Caldiera, and H Dieter Rombach. "The Goal Question Metric Approach". In: ().

[38] *Metrics Overview*. Ragas. Sept. 10, 2025. URL: https://docs.ragas.io/en/stable/concepts/metrics/overview/ (visited on 2025-09-24).

[39] Wai-Chung Kwan et al. *MT-Eval: A Multi-Turn Capabilities Evaluation Benchmark for Large Language Models*. Jan. 30, 2024. DOI: 10.48550/arXiv.2401.16745. arXiv: 2401.16745 [cs]. Pre-published.

[40] Tobias Schimanski et al. "ClimRetrieve: A Benchmarking Dataset for Information Retrieval from Corporate Climate Disclosures". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2024. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 17509–17524. DOI: 10.18653/v1/2024.emnlp-main.969.

[41] Shahul Es et al. *Ragas: Automated Evaluation of Retrieval Augmented Generation*. Apr. 28, 2025. DOI: 10.48550/arXiv.2309.15217. arXiv: 2309.15217 [cs]. Pre-published.

[42] Vaibhav Adlakha et al. "Evaluating Correctness and Faithfulness of Instruction-Following Models for Question Answering". In: *Transactions of the Association for Computational Linguistics* 12 (May 16, 2024), pp. 681–699. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00667.

[43] *Faithfulness*. Ragas. Aug. 26, 2025. URL: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/faithfulness/ (visited on 2025-09-24).

[44] Sewon Min et al. "NeurIPS 2020 EfficientQA Competition: Systems, Analyses and Lessons Learned". In: *Proceedings of the NeurIPS 2020 Competition and Demonstration Track*. NeurIPS 2020 Competition and Demonstration Track. PMLR, Aug. 7, 2021, pp. 86–111.

[45] Tianyi Zhang et al. *BERTScore: Evaluating Text Generation with BERT*. Feb. 24, 2020. DOI: 10.48550/arXiv.1904.09675. arXiv: 1904.09675 [cs]. Pre-published.

[46] Tianyu Gao et al. *Enabling Large Language Models to Generate Text with Citations*. Oct. 31, 2023. DOI: 10.48550/arXiv.2305.14627. arXiv: 2305.14627 [cs]. Pre-published.

[47] Nayyer Aafaq et al. "Video Description: A Survey of Methods, Datasets, and Evaluation Metrics". In: *ACM Computing Surveys* 52.6 (Nov. 30, 2020), pp. 1–37. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3355390.

[48] Yannis Katsis et al. *MTRAG: A Multi-Turn Conversational Benchmark for Evaluating Retrieval-Augmented Generation Systems*. Jan. 7, 2025. DOI: 10.48550/arXiv.2501.03468. arXiv: 2501.03468 [cs]. Pre-published.

[49] *Get Started*. Ragas. Sept. 17, 2025. URL: https://docs.ragas.io/en/stable/getstarted/ (visited on 2025-10-07).

[50] *Integrations*. Ragas. Sept. 30, 2025. URL: https://docs.ragas.io/en/stable/howtos/integrations/ (visited on 2025-10-07).

[51] Tom Kwiatkowski et al. "Natural Questions: A Benchmark for Question Answering Research". In: *Transactions of the Association for Computational Linguistics* 7 (Nov. 2019), pp. 453–466. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00276.

[52] Zhilin Yang et al. *HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering*. Sept. 25, 2018. DOI: 10.48550/arXiv.1809.09600. arXiv: 1809.09600 [cs]. Pre-published.

[53] Nicholas Pipitone and Ghita Houir Alami. *LegalBench-RAG: A Benchmark for Retrieval-Augmented Generation in the Legal Domain*. Aug. 19, 2024. DOI: 10.48550/arXiv.2408.10343. arXiv: 2408.10343 [cs]. Pre-published.

[54] Guangzhi Xiong et al. *Benchmarking Retrieval-Augmented Generation for Medicine*. Feb. 23, 2024. DOI: 10.48550/arXiv.2402.13178. arXiv: 2402.13178 [cs]. Pre-published.

[55] Rounak Meyur et al. *WeQA: A Benchmark for Retrieval Augmented Generation in Wind Energy Domain*. June 9, 2025. DOI: 10.48550/arXiv.2408.11800. arXiv: 2408.11800 [cs]. Pre-published.

[56] *Testset Generation for RAG*. Ragas. Sept. 15, 2025. URL: https://docs.ragas.io/en/stable/concepts/test_data_generation/rag/ (visited on 2025-10-02).

[57] *What Is a Knowledge Graph?* IBM. Oct. 14, 2021. URL: https://www.ibm.com/think/topics/knowledge-graph (visited on 2025-10-03).

[58] Jonas van Elburg, Peter van der Putten, and Maarten Marx. *Can We Evaluate RAGs with Synthetic Data?* Aug. 15, 2025. DOI: 10.48550/arXiv.2508.11758. arXiv: 2508.11758 [cs]. Pre-published.

[59] Rafael Teixeira de Lima et al. *Know Your RAG: Dataset Taxonomy and Generation Strategies for Evaluating RAG Systems*. Nov. 29, 2024. DOI: 10.48550/arXiv.2411.19710. arXiv: 2411.19710 [cs]. Pre-published.

[60] T. Y. Chen, S. C. Cheung, and S. M. Yiu. *Metamorphic Testing: A New Approach for Generating Next Test Cases*. 1998. DOI: 10.48550/arXiv.2002.12543. arXiv: 2002.12543 [cs]. Pre-published.

[61] Chris Van Der Lee et al. "Human Evaluation of Automatically Generated Text: Current Trends and Best Practice Guidelines". In: *Computer Speech & Language* 67 (May 2021), p. 101151. ISSN: 08852308. DOI: 10.1016/j.csl.2020.101151.

[62] Yu Bai et al. *Pistis-RAG: Enhancing Retrieval-Augmented Generation with Human Feedback*. Oct. 31, 2024. DOI: 10.48550/arXiv.2407.00072. arXiv: 2407.00072 [cs]. Pre-published.

[63] Noah Goodman. *Meta-Prompt: A Simple Self-Improving Language Agent*. Noah's Substack. Apr. 8, 2023. URL: https://noahgoodman.substack.com/p/meta-prompt-a-simple-self-improving (visited on 2025-08-28).

[64] Stephen Casper et al. *Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback*. Sept. 11, 2023. DOI: 10.48550/arXiv.2307.15217. arXiv: 2307.15217 [cs]. Pre-published.

[65] Lisana Berberi et al. "Machine Learning Operations Landscape: Platforms and Tools". In: *Artificial Intelligence Review* 58.6 (2025). ISSN: 0269-2821. DOI: 10.1007/s10462-025-11164-3.

[66] Zackary Rackauckas, Arthur Câmara, and Jakub Zavrel. *Evaluating RAG-Fusion with RAGElo: An Automated Elo-based Framework*. Oct. 8, 2024. DOI: 10.48550/arXiv.2406.14783. arXiv: 2406.14783 [cs]. Pre-published.

[67] Kishore Papineni et al. "BLEU: A Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*. The 40th Annual Meeting. Philadelphia, Pennsylvania: Association for Computational Linguistics, 2001, p. 311. DOI: 10.3115/1073083.1073135.

[68] Soki Kawashima, Shun Shiramatsu, and Takeshi Mizumoto. "Development of RAG System for Digital Transformation of Local Government and Considering Optimal Document-Segmentation Methods". In: *Proceedings of Ninth International Congress on Information and Communication Technology*. Ed. by Xin-She Yang et al. Vol. 1004. Singapore: Springer Nature Singapore, 2024, pp. 603–617. ISBN: 978-981-97-3304-0. DOI: 10.1007/978-981-97-3305-7_48.

[69] Yusuke Fukui et al. "Evaluation of a Retrieval-Augmented Generation System Using a Japanese Institutional Nuclear Medicine Manual and Large Language Model-Automated Scoring". In: *Radiological Physics and Technology* 18.3 (2025), pp. 861–876. ISSN: 1865-0333. DOI: 10.1007/s12194-025-00941-y.

Appendices

Appendix A
Literature Review Supplementals

A.1 Concept Matrix

Papers in the literature review were in the concept matrix mapped to the following concepts:

1. Benchmark datasets
2. Evaluation dataset
3. Training datasets
4. Automated test frameworks
5. Testing challenges
6. Hallucinations
7. Evaluation
8. Automated evaluation
9. Crowd-sourced evaluations
10. Manual evaluation
11. Metric
12. Human alignment
13. Medical care
14. Legal challenges

15. Ethical challenges
16. User expectations
17. Requirements specification
18. Explainable AI
19. Natural language
20. Translation
21. Technical implementation
22. QA Systems
23. RAG
24. Industry
25. Vulnerabilities
26. Reinforcement Learning
27. MLOps

Due to its size, the complete concept matrix is not reproduced in this thesis. The full concept matrix is available as an open dataset at Zenodo (DOI: 10.5281/zenodo.18164715) [27].

Table A.1: How each concept from the concept mapping maps to an overall area of interest. Some concepts were used for metadata purposes, and are not mapped to any area of interest.

Concept	Area of interest
Benchmark datasets	Question dataset
Evaluation dataset	Question dataset
Training datasets	Question dataset
Automated test frameworks	Automated test framework
Testing challenges	Testing challenges
Hallucinations	Testing challenges
Evaluation	Evaluation method
Automated evaluation	Evaluation method
Crowd-sourced evaluations	Evaluation method
Manual evaluation	Evaluation method
Metric	Evaluation metric
Human alignment	Human alignment
Medical care	*not mapped*
Legal challenges	*not mapped*
Ethical challenges	*not mapped*
User expectations	Testing challenges
Requirements specification	Testing challenges
Explainable AI	Evaluation method
Natural language	*not mapped*
Translation	*not mapped*
Technical implementation	*not mapped*
QA Systems	*not mapped*
RAG	*not mapped*
Industry	*not mapped*
Vulnerabilities	*not mapped*
Reinforcement Learning	*not mapped*
MLOps	*not mapped*

Table A.2: The venues from where papers were taken in the literature review (excluding preprints).

Venue	Number of papers
Annual Meeting of the Association for Computational Linguistics	1
ACM Computing Surveys	3
ACM International Conference on Information and Knowledge Management	1
ACM Conference on Fairness, Accountability and Transparency	1
International Conference on Human Agent Interaction	1
ACM Transactions on Software Engineering Methodology	1
ACM International Conference on Web Search and Data Mining	1
CEUR Workshop Proceedings	1
Engineering Applications of Artificial Intelligence	1
International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management	1
International Conference on Advances in Computation, Communication and Information Technology	1
IEEE Conference on Artificial Intelligence	1
IEEE International Conference on Software Testing, Verification and Validation Workshops	1
IEEE Swiss Conference on Data Science	1
IEEE Transactions on Software Engineering	1
Intensive Care Medicine	1
The ITEA Journal of Test and Evaluation	1
Journal of Web Engineering	1
Knowledge and Information Systems	1
Lecture Notes in Computer Science	1
MDPI Big Data and Cognitive Computing	1
MDPI Electronics	1
Natural Language Engineering	1
Natural Language Processing and Chinese Computing	1
Conference on Neural Information Processing Systems	1
Product-Focused Software Process Improvement. Industry-, Workshop-, and Doctoral Symposium Papers	1
Radiological Physics and Technology	1
Social Science Research Network	1
Transactions of the Association for Computational Linguistics	2

Table A.3: The inclusion, and if applicable, exclusion criteria, used for selecting which papers to include in each iteration of the literature review.

Topic	Paper inclusion criteria	Paper exclusion criteria
Benchmark datasets	Covering datasets used for benchmarking either novel question answering architectures, or benchmarking novel evaluation methods themselves.	Datasets that are not primarily used for benchmarking purposes.
Evaluation dataset	Covering datasets used for evaluating specific implementations of a question answering system.	
Training datasets	Covering datasets used for training or fine-tuning language models.	Datasets that are not used in training an artificial intelligence system.
Automated test frameworks	Covering test frameworks that are entirely or mostly automated.	Test frameworks that include significant manual work, or testing methodologies that cannot be placed neatly in a framework.
Testing challenges	Covering challenges related to testing question answering systems, challenges being defined broadly.	Testing challenges that are not unique for question answering systems.
Hallucinations	Covering hallucinations or performance issues of large language models or question answering systems.	
Evaluation	Covering the systematic evaluation of question answering systems against accuracy requirements.	Papers covering evaluation of other kinds of AI systems, or processes that do not primarily focus on accuracy.
Automated evaluation	Covering automated evaluation processes, or how evaluation processes generally can be automated.	
Crowd-sourced evaluations	Covering how crowd-sourced feedback can be used for evaluation purposes.	
Manual evaluation	Covering manual ways of evaluation.	

Table A.3: The inclusion, and if applicable, exclusion criteria, used for selecting which papers to include in each iteration of the literature review.

Topic	Paper inclusion criteria	Paper exclusion criteria
Metric	Covering quantitative ways of measuring question answering system performance with regard to accuracy.	Covering metrics or judgments that are qualitative in nature.
Human alignment	Covering how well automated metrics relate to human judgments.	
Medical care	Question answering system applied in medical care settings.	
Legal challenges	Legal challenges of using or deploying question answering systems.	Legal challenges that are related to other activities, such as training or testing.
Ethical challenges	Covering direct ethical challenges that arise when using or deploying question answering systems.	Ethical challenges that relate to high resource consumption of artificial intelligence systems.
User expectations	Covering the expectations users have when using question answering systems.	Covering user expectations on artificial intelligence, generally.
Requirements specification	Covering how requirements are specified, decomposed, and followed up on when testing question answering systems against accuracy requirements.	Covering primarily other kinds of requirements, specifically non-functional such as latency or resource consumption.
Explainable AI	Covering transparency and understanding of artificial intelligence processes.	
Natural language	Covering natural language processing applications.	
Translation	Covering translation applications of natural language processing systems.	
Technical implementation	Covering technical implementation details of a question answering system.	
RAG	Relating specifically to the RAG architecture.	Relating to other question answering architectures.

Table A.3: The inclusion, and if applicable, exclusion criteria, used for selecting which papers to include in each iteration of the literature review.

Topic	Paper inclusion criteria	Paper exclusion criteria
Industry	Relating to industry applications, whether actual systems or research-focused.	
Vulnerabilities	Covering vulnerabilities, malicious intent, or jailbreaking of artificial intelligence systems.	
Reinforcement Learning	Covering the use of feedback or metrics to improve the performance of a system.	
MLOps	Covering the deployment, maintenance, or operations of a machine learning system.	

Appendix B

Interview and Workshop Materials

B.1 Participant Profiles

System architect. An experienced system architect professional who has hands-on experience in designing and implementing large-scale enterprise systems. The participant is familiar with software testing strategies, although not necessarily those that apply to generative AI-based question answering systems. This profile provided insights into how software developers perceive and approach quality assurance, testing strategies, and the perceived applicability of the identified methods and metrics in real-world development.

Customer success manager. A customer success manager responsible for gathering and translating customer requirements into product features. This profile offered a customer-facing perspective on what end users value in terms of system quality, usability, and reliability, as well as of the acceptance and adoption of experimental features in large enterprise systems. These experiences helped contextualize how customer expectations align with the identified quality dimensions.

Developers (workshop participants). Three developers participated in the initial workshop alongside the System architect and Customer success manager. These participants provided complementary technical perspectives on the presented testing methods and metrics and contributed to identifying key discussion themes for the follow-up interviews.

B.2 Interview Guide and Questions

Here, we outline the questions that served as a guide for conducting the two individual interview sessions.

B.2.1 System Architect

- Is it important to test generative AI-based question answering systems? Why or why not?
- Which quality dimensions do you focus on when testing? Do you consider accuracy to be especially important?
- Which type of testing (among those we have reviewed) seems most interesting or useful to you? Is there any method you are skeptical about?
- How would you make use of the results from these metrics? Does a lack of trust in the metrics limit how useful they would be to you?
- How do you view the challenges of creating good test cases? How would you deal with those difficulties? Do you consider this a barrier for adoption of the outlined testing strategies?
- Why do you think there is a gap between the intention to test AI systems and the fact that most organizations don't? Is this assumption correct, in your experience?

B.2.2 Customer Success Manager

- Is it important to test generative AI-based question answering systems?
- Which quality dimensions do you consider the most important?
- What requirements do customers have for generative AI-based question answering systems? How do these requirements differ between customers?
- Are such systems typically viewed as a fun or interesting gimmick, or do customers see them as tools that could eventually replace manual processes? Or as a complement to them?
- How do you think about different ways of communicating quality to users? How do you ensure that users have trust in the systems they use?

B.3 Thematic Analysis

A thematic analysis was performed to identify and analyze patterns in the data collected during the Q&A session and individual interviews. Passages in the detailed notes taken during the workshop and individual interviews were annotated using the free open-source tool *Taguette* [29]. The following themes were identified.

- quality assurance
- business requirements
- continuous improvement

- framing a question answering product
- system learning and evolution
- metrics and methods
- perception of accuracy and reliability
- question dataset
- stakeholder alignment
- system role
- test automation
- testing challenges
- trust and transparency
- user expectations

EXAMENSARBETE Systematic test strategies for generative question answering systems
STUDENTER Jacob Bentzer, Henrik Vester
HANDLEDARE Emelie Engström (LTH)
EXAMINATOR Elizabeth Bjarnason (LTH)

När AI svarar på frågor: en verktygslåda för att testa och kvalitetssäkra

POPULÄRVETENSKAPLIG SAMMANFATTNING **Jacob Bentzer, Henrik Vester**

I takt med att AI-modeller används i fler och fler situationer ökar kraven på kvalitet i svaren. Detta arbete presenterar en strukturerad modell för hur man kan lägga upp testning av AI-system som svarar på frågor, och hur man kan automatisera sådan testning.

Testning av olika sorters programvara och system har förekommit ända sedan de första datorprogrammen tillkom på 1950-talet. Moderna AI-system, exempelvis ChatGPT, skiljer sig dock på många sätt från vanliga datorprogram. Det är som bekant inte säkert att man får samma svar varje gång en fråga ställs, och ett rätt svar kan se ut på många olika sätt. Därför är det svårt, även för en människa, att utröna om ett visst svar är acceptabelt eller inte. Därför kräver testning av AI-system nya metoder och angreppssätt.

I vårt examensarbete har vi utvecklat en taxonomi av de AI-testmetoder som finns, och satt dem i ett större sammanhang. Det finns många olika aspekter av vad som gör ett svar faktamässigt eller inte, som vi har delat in i fem områden: svarsrelevans (att svaret besvarar frågan), rättrogenhet (att AI-systemet håller sig till ämnet), korrekthet (att svaret överensstämmer med känd fakta), referenshantering (att systemet klarar av att hänvisa till varifrån svaret kommer) och källrelevans (att systemet klarar av att skilja på relevanta och icke-relevanta källor). För var och en av dessa områden har vi utforskat vilka metoder som finns tillgängliga för att testa dessa, och vilka mätetal som kan användas för att kvantifiera hur väl ett system uppfyller dessa krav. Metoderna skiljer sig åt i hur väl de kan automatiseras, och vad som krävs av oss människor – till exempel måste man i vissa fall skapa en slags testdatabas av exempelfrågor med kända svar.

Fråga	Referensdokument	Referenssvar
Varför hjälper inte antibiotika mot förkylning?	Folkhälsomyndigheten, 1177	Förkylning orsakas av virus.

Vi har också djupdykt i tre stycken fallstudier som använder metoderna vi har beskrivit, för att kunna se hur väl metoderna fungerar i praktiken. Där ser vi att det generellt sett fungerar bra att automatisera vissa delar av testningen, särskilt de delar som rör källhantering. Vi ser dock att det i andra fall fortfarande krävs orimligt mycket arbete av den som testar att manuellt lägga in referenssvar, och att det är ett problem att mätetalen som fås ut av vissa metoder inte överensstämmer särskilt väl med hur en människa hade betygssatt svaren. Testningen så som den ser ut i praktiken är också ofta onödigt snäv, och utelämnar ofta relevanta aspekter som till exempel svarsrelevans.