

Development of an Interactive Search System for Standardised Documents in Legal Metrology and Comparison with a Retrieval-Augmented Generation Approach

Zihao Wang¹, Marko Esche¹, Martin Nischwitz¹, Hans Grönniger²

¹Physikalisch-Technische Bundesanstalt, Department 8.5 „Metrological IT“, Abbestr. 2-10, 10587 Berlin, Germany, Tel.: 030 3481 7975, E-Mail: marko.esche@ptb.de

²Ostfalia University of Applied Sciences, Faculty of Computer Science/IT, Am Exer 2, 38302 Wolfenbüttel, Germany, Tel.: 05331 939 31190, E-Mail: h.groenniger@ostfalia.de

Abstract

Modern measurement systems are becoming increasingly diverse and complex, with software playing an ever more important role. Ensuring the compliance of software protection features with legal requirements has thus become a core task in legal metrology. Focusing on technical solutions harmonised in WELMEC Guide 7.3, this paper develops and evaluates two types of document content retrieval systems: a custom interactive search system based on automated scripts, database, and a search engine, which supports structured and full-text search, fuzzy matching, and relevance ranking; and a Retrieval-Augmented Generation (RAG) approach that integrates dense vector retrieval with large language models, enabling natural language queries. Experimental results demonstrate that the interactive search system achieves stable and accurate performance in locating target entries and handling complex queries, whereas the RAG system, which relies heavily on model quality and prompt design, often struggles to maintain accuracy and stability. This paper confirms the practical value of the interactive search system in supporting software examination and provides a reference for the applicability of the RAG approach.

Keywords: Document Content Retrieval, Interactive Search System, Retrieval-Augmented Generation (RAG), Large Language Model (LLM), Legal Metrology

Introduction

The Physikalisch-Technische Bundesanstalt (PTB) is the National Metrology Institute of Germany and, among other tasks, is responsible for the evaluation of software in electronic measuring instruments, e.g., taximeters, electricity meters, heat meters, based on the guides (hereafter referred to as “Guide”) published by the European Cooperation in Legal Metrology (WELMEC). Among these, the Software Guide 7.2:2025 [1] serves as a key reference both for software examiners and for manufacturers of measuring instruments when preparing their documentation for conformity assessment.

Following the current revision of Guide 7.3, the new version of Guide 7.3 primarily covers the acceptable solutions transferred and adapted from the central requirement document Guide 7.2, with each solution having four keyword attributes. A typical use case involves developers and examiners locating and consulting all relevant acceptable solutions based on specific keywords. However, the guide is avail-

able only in paper format or as a PDF file, contributing to inefficient information retrieval.

To further advance the digital transformation of standards, the International Electrotechnical Commission (IEC) and the International Organization for Standardization (ISO) jointly introduced SMART Standards, which stands for Standards that are Machine Applicable, Readable, and Transferable. This concept has been widely adopted by multiple organisations and initiatives. Its most fundamental requirement is to use machine-readable formats, e.g., XML or JSON, to store document content, enabling automated processing and facilitating subsequent applications [2].

Among the vast collection of standardised documents, a significant portion serves only for content retrieval, without requiring integration or interaction with machine systems. How to efficiently retrieve the content of such documents has become an important research topic. Although custom-built database search systems can satisfy the retrieval needs of different standardised documents, they suffer from high development and maintenance costs

and limited flexibility when adapting to varying content structure or query requirements. To reduce the cost of manual processing, there is a growing need to leverage AI technology to enable more intelligent document content retrieval.

Dense retrieval methods that combine text embeddings and vector-based queries have been widely applied in information retrieval tasks [3, 4]. However, relying solely on semantic matching makes it difficult to model logical relationships between contexts, which imposes limitations when querying structured information. Recent advances in large language models (LLMs) for natural language understanding, reasoning, and generation [5] have spurred the development of Retrieval-Augmented Generation (RAG) methods [6, 7]. These methods provide the retrieved relevant passages as context to LLMs, enabling them to answer natural language queries about documents. This not only improves the accuracy of results [8, 9] but also reduces the risk of producing outputs that appear plausible but do not align with the provided source inputs or factual information, i.e., hallucination [10].

This paper focuses on transforming key tabular content from the new Guide 7.3 into structured, machine-readable data through automated scripts, and developing an interactive web application to support efficient search and content access.

It also aims to evaluate the performance of RAG systems when applied to this document under different configurations of the retrieval and generation components. Experiments will be conducted using a set of quality metrics to assess the reliability of the interactive search system and the RAG systems. The goal is to summarise the applicability of the two systems in this specific use case.

The remainder of this paper is structured as follows. Section 2 introduces the components of the interactive search system. Section 3 describes the construction of the RAG system. Section 4 covers the experimental setup, evaluation metrics and the experimental results in tabular form. Section 5 visualises and analyses the results, and discusses the limitations of the current experiments. Finally, Section 6 concludes the paper and outlines possible directions for future work.

Interactive Search System

The developed system employs a frontend and backend separated architecture and is primarily composed of four modules: frontend, backend, database, and search engine.

The frontend is built using React to construct the web user interface, responsible for receiv-

ing user queries, performing format validation, and displaying search results. The backend, based on the Flask framework, provides a RESTful endpoint and parses query parameters to construct the corresponding database queries. MongoDB is used to store all acceptable solutions from Guide 7.3 and exchanges data with the backend, while Typesense, as a search engine, supports full-text search functionalities and directly responds to requests from the frontend.

All modules are run as Docker containers and are managed collectively via Docker Compose. Figure 1 illustrates the data communication among the containers. The init-typesense container, which is responsible for initialising the Typesense data, is executed only once.

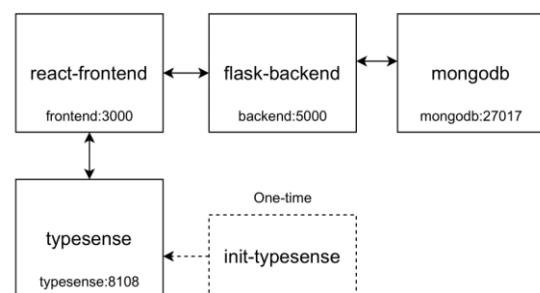


Fig. 1: Communication among docker containers

As shown in the table in Figure 2, each acceptable solution contains seven sections: "Name", "Description", "Variants", "Conditions", "Keywords", "Required Documentation", and "Functional Checks". There are four types of keywords: "Assets", "Security properties", "Attack vectors" and "Status". Each type can contain one or more specific values, with "all" representing the complete set of values for that category.

WELMEC Guide 7.3

Acceptable Solutions

Assets	Security properties	Attack vectors	Status
software			
Full-text Search			
bootloader			

Found 3 results.

Solution: bootloader secured with a password

Solution: bootloader secured by trusted environment

Assets: software

Security properties: integrity, availability, authenticity

Attack vectors: ID1

Status: active

Solution: bootloader secured with a password

Assets: software

Security properties: integrity, availability, authenticity

Attack vectors: ID1

Status: active

Solution: secure boot

Assets: software

Security properties: integrity, availability, authenticity

Attack vectors: ID1

Status: active

Risk Class B	Risk Class C	Risk Class D
General		
- The boot configuration is secured by a password, see 4.4.2.1. - The integrity of the legally relevant parts of the operating system is checked by the bootloader by means of a reference value, see 4.1.1. - The integrity of the bootloader is physically protected by means of a hardware seal, see 4.4.3.1.		
Variants		
• •		
Conditions		
• •		
Keywords		
- Assets: software - Security properties: integrity, availability, authenticity - Attack vectors: ID1 - Status: active		
Required Documentation		
Description of the boot order and the interactions between boot configuration, bootloader and operating system.		
Functional Checks		
- Check that modification of the boot configuration is not possible without entering a password. - Check that booting of a modified boot configuration is not possible.		

Fig. 2: User interface of the interactive search system

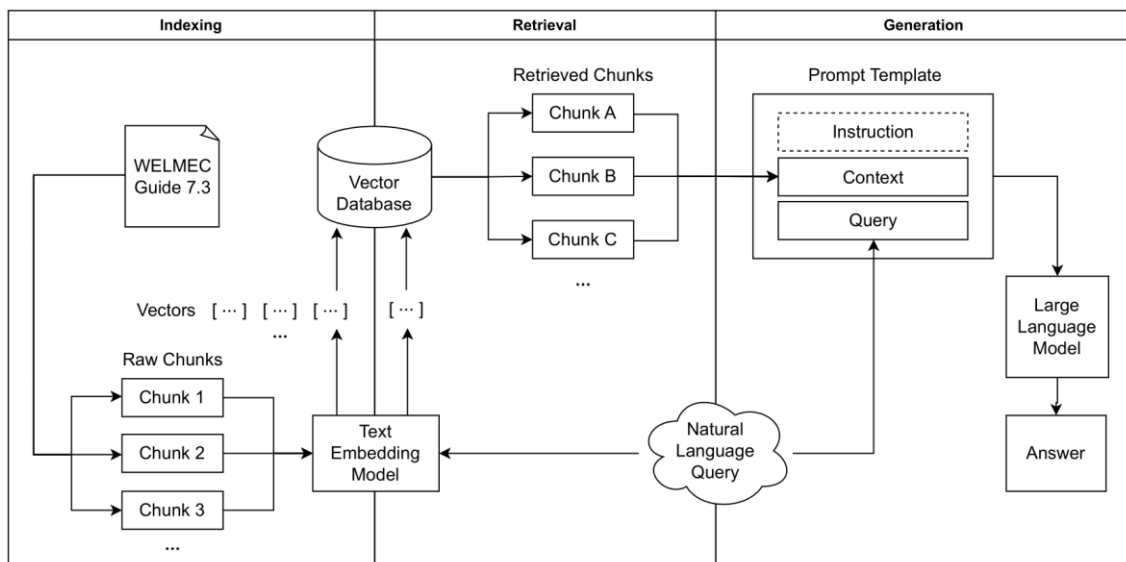


Fig. 3: Workflow of the RAG system

Four keyword input fields are provided at the top, corresponding to the four types of keywords in the “Keywords” section of an acceptable solution, along with a full-text search input field. The lower part is divided into two columns: the left side displays all retrieved results in the form of clickable cards, while the right side renders the title and table content of the selected acceptable solution in its original format, thereby ensuring that the presentation remains consistent with the official document.

Retrieval-Augmented Generation System

Based on the LlamaIndex framework, a basic RAG pipeline is implemented using exclusively a dense retrieval strategy. The overall workflow of the system is illustrated in Figure 3.

It begins with data ingestion, where the input DOCX document is parsed according to headings and tables, enabling semantic chunking of the Guide 7.3. All chunks are converted into dense vectors using a text embedding model and stored in the Chroma vector database. These stored vectors can be reused without being regenerated each time.

To optimise retrieval results, two parameters are configured: `similarity_top_k`, which limits the number of chunks returned from the matching results; and `similarity_cutoff`, which filters out results with low similarity scores.

For the LLM, the temperature parameter controls the randomness of the generated responses, with lower values leading to more stable outputs. Additionally, a prompt template can be provided to improve the quality of the generated content. The basic version, represented by `basic_prompt`, includes the retrieved content along with the user query.

Comparative experiments with various parameters and configurations will be conducted in the next section to evaluate the performance of different RAG systems.

Experiments and Results

In order to better analyse the performance of dense retrieval in identifying relevant texts and the further optimisation of results by LLMs, the retrieval and generation components of the RAG system are evaluated separately. OpenLLMetry, an open-source project for tracking and monitoring LLM applications, is employed here to observe the text chunks obtained during the retrieval stage.

The test cases are divided into two sets: Test Set 1 and Test Set 2. Test Set 1 contains 20 queries of varying complexity, using keyword values that actually exist in acceptable solutions. Test Set 2 contains 10 queries with non-existent values to test the system’s reliability when queries do not match any acceptable solutions. The following Experiments 1–4 are conducted on Test Set 1, while Experiment 5 is conducted on Test Set 2.

Experiment Set 1 (E1) provides a comprehensive performance comparison between the interactive search system and the RAG systems, using Test Set 1. In this experiment, OpenAI’s text embedding models, `text-embedding-3-small` and `text-embedding-3-large`, are used to generate dense vectors of different dimensions. To ensure retrieval completeness even in extreme cases, the `similarity_top_k` parameter is set higher than the total number of acceptable solutions (here set to 100), while `similarity_cutoff` is set to 0.2 to prevent relevant results from being prematurely excluded. The generation component uses

two general-purpose models, gpt-4omini-2024-07-18 and gpt-4.1-2025-04-14, as well as a reasoning model, o4-mini-2025-04-16, with the temperature parameter set to 0 to ensure stable responses.

To further enhance the quality of the generated outputs, a custom prompt template is designed. This template includes task analysis, definitions of the acceptable solution, and instructions on how to handle situations where no relevant information is found, with the aim of mitigating model hallucinations. It is noteworthy that the keywords fields may contain the value "all". To ensure the LLM accurately matches these fields when processing specific queries, the prompt explicitly specifies all corresponding values.

An excessively low similarity_cutoff value may place a high burden on the LLM and may include irrelevant results, while a value that is too high may unnecessarily exclude relevant results. To illustrate the effect of this parameter, different similarity_cutoff values (0.25, 0.30, and 0.35) are used in Experiment Set 2 (E2), with text-embedding-3-small employed for text embeddings.

If the temperature is not constrained, the LLM generates more diverse outputs, which are potentially less reliable or may even contain fabricated information. Experiment Set 3 (E3) quantifies this phenomenon by testing temperature values of 0.3, 0.6, 0.9 and 1.2, while the remaining configurations are set as follows: similarity_cutoff of 0.2, custom prompt, text-embedding-3-small and o4-mini-2025-04-16.

To demonstrate the impact of prompt quality on three different LLMs, Experiment Set 4 (E4) compares results using the custom prompt versus using only the retrieved text chunks and the user query as the prompt. Other configurations are: similarity_cutoff of 0.2, temperature of 0, and textembedding-3-small.

Finally, Experiment Set 5 (E5) examines whether the interactive search system and the RAG systems using three different LLMs return no results when queried for non-existent values. The settings are: similarity_cutoff of 0.2, temperature of 0, custom prompt, and text-embedding-3-small.

The evaluation is conducted using five traditional quality metrics, namely Precision (P), Recall (R), F1 Score (F1), Fallout (F), and Miss Rate (M). In addition, the Hallucination Rate (H) is calculated to evaluate the generation component of the RAG systems. Hallucination is calculated as the ratio of fabricated items generated by the LLM to the total number of generated items, representing the probability that the LLM produces incorrect information.

Table 1 presents the results of the RAG system using two different text embedding models and three different LLMs in E1. The results of the interactive search system are also listed in the table for comparison.

Tables 2 shows how the retrieval performance varies with similarity_cutoff values of 0.25, 0.30, and 0.35 in E2.

Table 3 presents the results of E3, illustrating how different temperature settings affect the generation performance of the RAG system.

Tab. 1: Results of different RAG systems and the interactive search system (E1)

Model		Retrieval					Generation					
		P (%)	R (%)	F1 (%)	F (%)	M (%)	P (%)	R (%)	F1 (%)	F (%)	M (%)	H (%)
embedding-small	4o-mini	43.33	100.00	55.91	100.00	0.00	54.54	21.03	24.33	8.73	78.97	0.00
	4.1						78.28	51.69	57.02	4.90	43.31	0.00
	o4-mini						100.00	94.43	96.02	0.00	5.57	0.00
embedding-large	4o-mini	43.33	100.00	55.91	100.00	0.00	68.06	25.30	33.90	3.56	74.70	0.00
	4.1						86.70	45.44	54.04	5.63	54.56	0.00
	o4-mini						98.83	95.82	96.12	2.69	4.18	0.00
Interactive Search System		95.00	95.00	95.00	0.00	5.00	-	-	-	-	-	-

Tab. 2: Results of different similarity_cutoff thresholds (E2)

similarity_cutoff	Retrieval				
	P (%)	R (%)	F1 (%)	F (%)	M (%)
0.25	43.46	93.71	55.28	96.00	6.29
0.30	45.25	58.39	43.59	61.88	41.61
0.35	54.82	22.87	25.44	12.44	77.13

Tab. 3: Results of different temperature values (E3)

temperature	Generation					
	P (%)	R (%)	F1 (%)	F (%)	M (%)	H (%)
0.3	94.81	85.21	86.57	3.19	14.79	0.00
0.6	97.62	94.17	93.90	6.50	5.83	0.00
0.9	100.00	88.04	90.27	0.00	11.96	0.00
1.2	100.00	95.19	95.84	0.00	4.81	0.00

Tab. 4: Results of different LLMs using basic prompt (E4)

LLM	Generation					
	P (%)	R (%)	F1 (%)	F (%)	M (%)	H (%)
4o-mini	48.09	22.30	26.03	13.32	77.70	0.00
4.1	74.50	30.76	39.79	0.22	69.24	0.00
o4-mini	70.00	24.79	33.40	0.00	75.21	0.00

Table 4 shows the generation performance of three different LLMs with only the retrieved text chunks and the user query as the prompt in E4. Table 5 records the Fallout and Hallucination Rate of RAG systems using three different LLMs when handling queries with non-existent values in E5, while for the interactive search system, only the Fallout is recorded.

Discussion

Figure 4 presents a comparison between the output quality of the different RAG systems and the performance of the interactive search system. Among them, RAG systems using the reasoning model o4-mini could better understand the task instructions in the custom prompt and performed well, yet they still failed to return all relevant acceptable solutions exclusively.

It is noteworthy that the interactive search system did not achieve a 100% F1 score and a 0% Miss Rate. This resulted from its reliance on boolean matching, which leads to no results being returned for queries containing spelling errors. In contrast, high-quality LLMs can comprehend instructions regarding typo handling and therefore demonstrate an advantage in responding to such queries.

With the similarity_cutoff set to 0.2, the size of the text embedding model did not affect the

Tab. 5: Results of queries with non-existent values (E5)

LLM	F (%)	H (%)
4o-mini	3.61	0.00
4.1	3.33	0.00
o4-mini	3.06	0.00
Interactive Search System	0.00	-

results, and the retrievers in each RAG system returned all acceptable solutions. The only difference was that the similarity scores obtained for the same query varied across vector databases, causing the acceptable solutions within the LLMs' context to be ranked differently. However, this mainly affected the outputs of non-reasoning models, while the reasoning model was hardly influenced, and the similarity score remains an uncontrollable variable with no research significance.

Table 1 shows that, under this configuration, the retrieval component of the RAG system obtained identical results across all five metrics. Due to an excessively low cutoff value, the retriever passed all acceptable solutions to the LLM, which achieved the highest recall but led to very poor precision, resulting in an F1 score of only around 50%. Figure 5 illustrates the changes after increasing the similarity_cutoff value: the accuracy of the results does not improve significantly, while recall drops sharply, causing more relevant acceptable solutions to be missed.

Through analysis of the test results, it was found that the keywords specified in user queries did not effectively constrain the retrieval stage. For example, for the query "List the names of all acceptable solutions where the 'Assets' field contains 'software'.", acceptable solutions whose assets field contains the

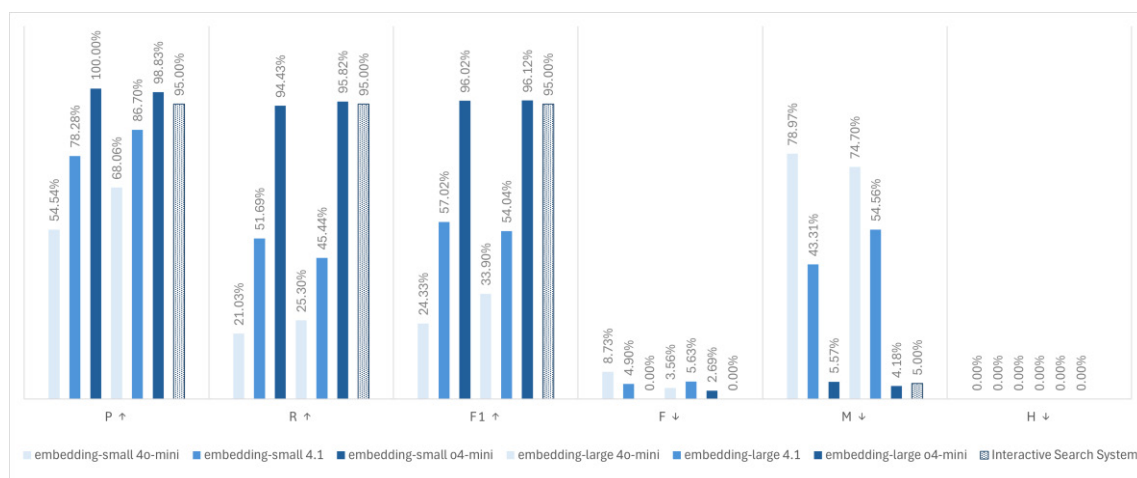


Fig. 4: A comprehensive performance comparison between RAG systems using different models and the interactive search system (E1)

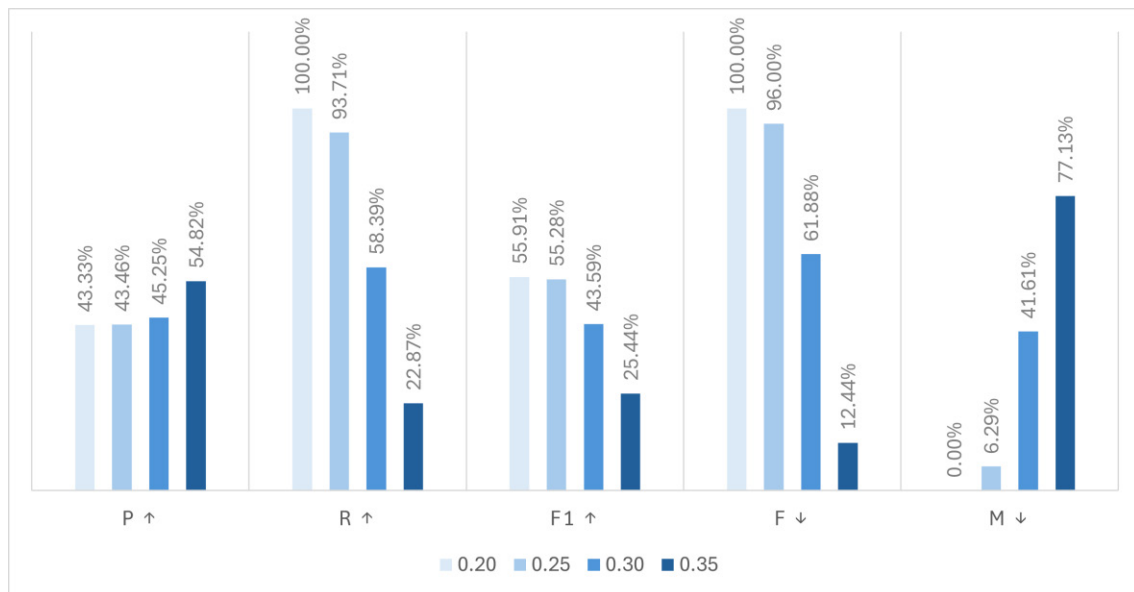


Fig. 5: Comparison of the retrieval performance of the RAG system with varying similarity_cutoff thresholds (E2)

term “software” do not necessarily obtain higher scores in similarity comparisons. The reason is that, apart from the “Keywords” section, semantic information in other sections negatively affects vector index construction and matching. This implies that increasing the similarity_cutoff value may lead to the loss of relevant documents during retrieval, thereby reducing overall usability.

Figure 6 shows the impact of the temperature parameter on the performance of the LLM. As the temperature value increases, the F1 score fluctuates without showing a clear trend, and this also applies to the four other metrics. Notably, the Hallucination Rate remains at 0%.

the LLMs successfully followed the instructions in the custom prompt and completed tasks without producing fabricated answers. In general, when using high-quality LLMs with a well-designed prompt, the setting of the temperature has a negligible effect on the performance. Figure 7 illustrates the impact of prompt quality on the performance of different LLMs. When no explicit instructions or special-case handling were provided, all three LLMs performed poorly. Unexpectedly, even without explicitly instructing the model not to fabricate information, no hallucinations occurred in the task of returning title texts.

When more detailed descriptions were added, the RAG system using low-quality model 4o-

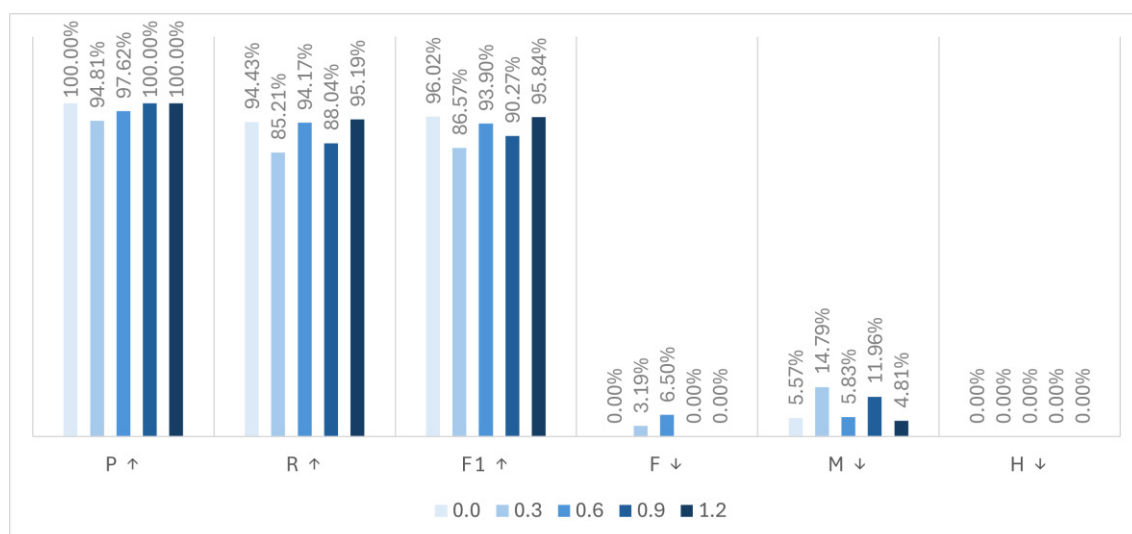


Fig. 6: Comparison of the generation performance of the RAG system with varying temperature values (E3)

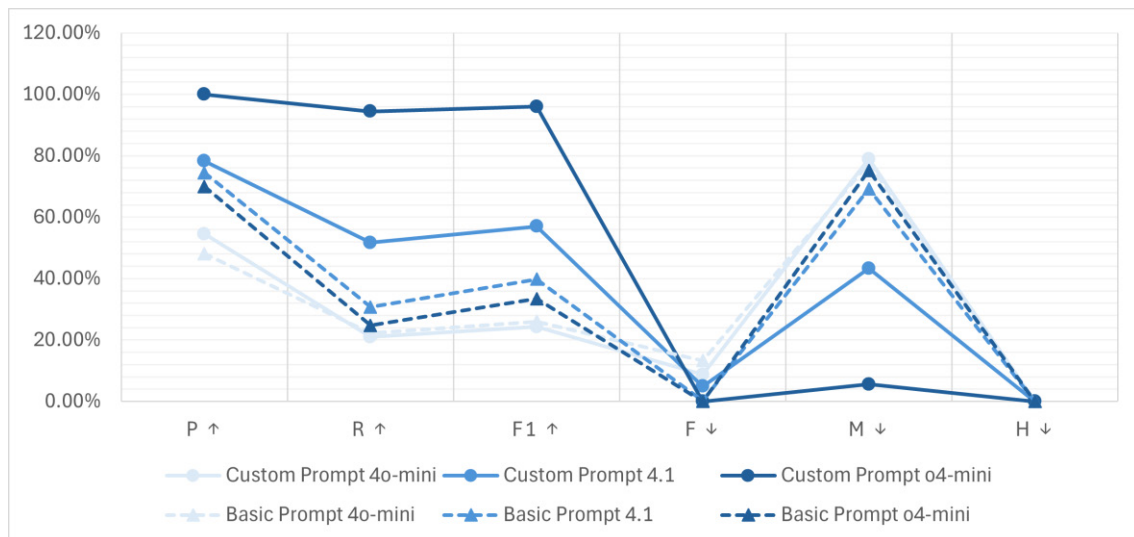


Fig. 7: Comparison of the generation performance of RAG systems using different LLMs under basic prompt (E4)

mini maintained almost the same level of performance. In contrast, the RAG system built on the reasoning model o4-mini nearly tripled its F1 score, with a significant drop in the Miss Rate. This improvement is mainly attributed to translating statements containing “all” into their corresponding values in the prompt, allowing a large number of relevant acceptable solutions to be retrieved. Despite some improvement observed with the 4.1 model, its overall performance remains insufficient for practical use. As shown in Figure 8, in the final set of experiments, when faced with keywords that do not exist in the documents, all RAG systems were able to maintain a Hallucination Rate of 0%, thereby ensuring a certain level of reliability. However, even the best among the three tested LLMs could not definitively determine that there are no matching documents in every test

case. Therefore, in such a scenario, RAG systems cannot guarantee complete correctness. The main limitation of this test is that there are only 36 acceptable solutions in Guide 7.3. Even if all of them are passed to the LLM, it does not exceed the context window limit. When the number increases, using a relatively low similarity_cutoff value may result in LLM failure or forced truncation of excess text. Future work should test documents containing more of these acceptable solutions to further evaluate the system’s usability. Moreover, each test case under each experimental configuration was executed a single time through the RAG pipeline and the results were recorded. Especially for RAG systems with higher temperature, the results may show certain variability. Future work could conduct multiple runs and average the results to obtain more reliable conclusions.

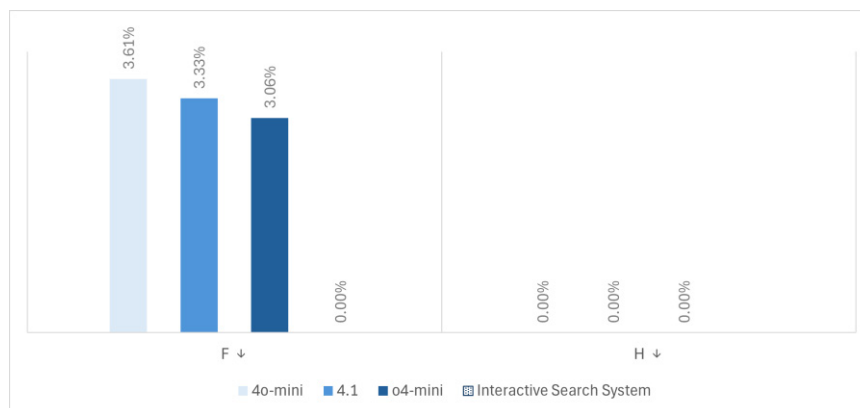


Fig. 8: Comparison of the performance of RAG systems and interactive search system on queries with non-existent values (E5)

Conclusion and Outlook

In this paper, an interactive search system was developed, which supports both structured and full-text searches of acceptable solutions contained in WELMEC Guide 7.3, with results displayed on the web interface in their original format. The system automates the data processing workflow through multiple scripts for data extraction, database storage, and search engine import, thereby minimising manual work and associated costs. In addition, it supports complex queries, fuzzy matching, result highlighting, and relevance ranking, effectively improving document retrieval efficiency and user query experience.

This paper also implemented a RAG approach that combines dense vector retrieval with LLMs. This method does not require a data model and allows generating answers directly from natural language queries on the document content. It reduces development cost and can be easily applied to a wide range of documents.

Experimental results show that the interactive search system performs excellently in all tests, except in cases involving spelling errors. It accurately locates target acceptable solutions and maintains high precision even in complex query scenarios, demonstrating its applicability for efficient content retrieval of standardised documents. In contrast, the RAG systems using general-purpose LLMs failed to meet practical requirements across various parameters and configurations; even with a reasoning LLM, a well-designed prompt is necessary to achieve basic usability, yet complete accuracy and stability could not be guaranteed, which presents difficulties for practical applications.

Some limitations still exist. Firstly, the retrieval component currently relies solely on dense retrieval, and the inclusion of irrelevant text in vector construction has reduced retrieval performance. Future work could consider constructing vectors only from the “Keywords” section of acceptable solutions, adopting sparse retrieval methods (e.g., BM25), or employing a hybrid retrieval strategy. Secondly, in the current prompt template, retrieved acceptable solutions are ranked in descending order of similarity score, meaning those with lower similarity scores are positioned closer to the user query. According to [11], placing the target document near the query improves accuracy, and the study also suggests that adding irrelevant documents may further enhance system performance. These findings provide valuable insights for future work.

References

- [1] “WELMEC 7.2 Software Guide,” European Cooperation in Legal Metrology, WELMEC Secretariat, Braunschweig, Standard, Sep. 2025
- [2] “Scenarios for Digitizing Standardization and Standards,” DIN e. V. and DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE, Berlin and Frankfurt am Main, White Paper, June 2021
- [3] J. Herzig, T. Müller, S. Krichene, and J. Eisen-schlos, “Open domain question answering over tables via dense retrieval,” in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 512–519, doi: 10.18653/v1/2021.naacl-main.43
- [4] A. Asai, X. Yu, J. Kasai, and H. Hajishirzi, “One question answering model for many languages with cross-lingual dense passage retrieval,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 7547–7560, 2021
- [5] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, et al., “A survey of large language models,” *arXiv preprint arXiv:2303.18223*, 2023, doi: 10.48550/arXiv.2303.18223
- [6] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel et al., “Retrieval-augmented generation for knowledge-intensive nlp tasks,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020
- [7] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, and H. Wang, “Retrieval-augmented generation for large language models: A survey,” *arXiv preprint arXiv:2312.10997v5*, 2024, doi: 10.48550/arXiv.2312.10997
- [8] C. Melton, A. Sorokine, and S. Peterson, “Evaluating Retrieval Augmented Generative Models for Document Queries in Transportation Safety,” *arXiv preprint arXiv:2504.07022*, 2025, doi: 10.48550/arXiv.2504.07022
- [9] Y. H. Ke, L. Jin, K. Elangovan, H. R. Abdullah, N. Liu, A. T. H. Sia, C. R. Soh, J. Y. M. Tung, J. C. L. Ong, C.-F. Kuo et al., “Retrieval augmented generation for 10 large language models and its generalizability in assessing medical fitness,” *npj Digital Medicine*, vol. 8, no. 1, p. 187, 2025, doi: 10.1038/s41746-025-01519-z
- [10] H. Ye, T. Liu, A. Zhang, W. Hua, and W. Jia, “Cognitive mirage: A review of hallucinations in large language models,” *arXiv preprint arXiv:2309.06794*, 2023, doi: 10.48550/arXiv.2309.06794
- [11] F. Cuconasu, G. Trappolini, F. Siciliano, S. Filice, C. Campagnano, Y. Maarek, N. Tonello, and F. Silvestri, “The power of noise: Redefining retrieval for RAG systems,” in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024, pp. 719–729, doi: 10.1145/3626772.3657834