Evaluating RAG-Fusion with RAGElo: an Automated Elo-based Framework

Zackary Rackauckas[1], Arthur Câmara[2] and Jakub Zavrel[2]

[1]Columbia University, New York, NY, USA
[2]Zeta Alpha, Amsterdam, The Netherlands

Abstract

Challenges in the automated evaluation of Retrieval-Augmented Generation (RAG) Question-answering (QA) systems include hallucination problems in domain-specific knowledge and the lack of gold standard benchmarks for company-internal tasks. This results in difficulties in evaluating RAG variations, like RAG-Fusion (RAGF) in the context of a product QA task at Infineon Technologies. To solve these problems, we propose a comprehensive evaluation framework, which leverages Large Language Models (LLMs) to generate large datasets of synthetic queries based on real user queries and in-domain documents, uses LLM-as-a-judge to rate retrieved documents and answers, evaluates the quality of answers, and ranks different variants of Retrieval-Augmented Generation (RAG) agents with RAGElo's automated Elo-based competition. LLM-as-a-judge rating of a random sample of synthetic queries shows a moderate, positive correlation with domain expert scoring in relevance, accuracy, completeness, and precision. While RAGF outperformed RAG in Elo score, a significance analysis against expert annotations also shows that RAGF significantly outperforms RAG in completeness, but underperforms in precision. In addition, Infineon's RAGF assistant demonstrated slightly higher performance in document relevance based on MRR@5 scores. We find that RAGElo positively aligns with the preferences of human annotators, though due caution is still required. Finally, RAGF's approach leads to more complete answers based on expert annotations and better answers overall based on RAGElo's evaluation criteria.

Keywords

Retrieval-augmented generation, Elo-based evaluation, LLM-as-a-judge, RAG-Fusion

1. Introduction

The text-generating capabilities of LLMs, together with their text understanding abilities, have allowed conversational Question-Answering (QA) systems to experience a considerable leap in performance, with near-human text quality and reasoning capabilities [1]. However, these systems can be prone to hallucinations [2, 3], as they sometimes produce seemingly plausible but factually incorrect answers.

The general inability of such models to identify unanswerable questions [4, 5] can exacerbate hallucinations, especially in enterprise settings. In such scenarios, user questions may require specific domain knowledge to be answered properly. This knowledge

LLM4Eval 2024:The First Workshop on Large Language Models for Evaluation in Information Retrieval, 18 July 2024, Washington, DC

zcr2105@columbia.edu (Z. Rackauckas); camara@zeta-alpha.com (A. Câmara); zavrel@zeta-alpha.com (J. Zavrel)

is usually out-of-domain for most LLMs, but is present in private and confidential internal documents from the company.

One such company is Infineon, a leading manufacturer of semiconductors. Given its wide range of equipment, information about its products is spread across multiple, highly technical documents, including datasheets and selection guides of hundreds of pages. Therefore, an internal retrieval augmented conversational QA system was developed by Infineon for internal users such as account managers, field application engineers, and sales operations specialists. This system allows professionals to ask questions about products from the whole catalog while in the field.

One of the features of Infineon's conversational agent is the usage of *RAG*-Fusion (*RAGF*), a technique for increasing the quality of the generated answers by generating variations of the user question and combining the rankings produced by these variations using rank-fusion methods (i.e., recriprocal rank fusion (RRF) [6]) into a ranking that has both more diverse and higher quality answers.

However, evaluating these systems bring complications common to retrieval augmented agents, especially in enterprise settings, stemming from the lack of comprehensive test datasets. Ideally, such a test set would comprise a large set of real user questions from a query log, paired with "golden answers" provided by experts. The lack of such a test set leads to two main issues. First, evaluation of answers generated by LLMs by traditional n-gram evaluation metrics such as ROUGE [7], BLEU [8], and METEOR [9] is not possible, given the lack of ground truth answers. Second, and as a consequence, evaluating the quality of the answers generated by the LLM systems would require in-domain experts (potentially from within the company) in a process that is both slow and costly [10].

One approach for tackling the lack of an extensive test set is to use synthetic queries generated by LLMs as a proxy of user queries [11]. However, the lack of in-domain knowledge of LLMs makes queries naively generated by these models unreliable and prone to hallucinations, especially when generating queries about specific products and their specifications (c.f., Table 1 for examples of real user's questions submitted to the system).

To solve this, we propose to use a process similar to InPars [12] to create a set of synthetic evaluation queries. We ask LLMs to generate queries based on portions of existing documentation injected into the prompt. To increase similarity to real user queries, we include existing user questions as few-shot examples to the prompt. With this process, we are able to generate a large set of high-quality synthetic queries for evaluating our systems. Figure 2 describes the process of generating synthetic queries and the output of a search agent. Table 2 shows a sample of these queries.

To tackle the second issue, a lack of ground truth "golden answers," we leverage an LLM-as-a-judge process, where a strong LLM is used to evaluate the quality of the answers generated by the RAG agent's LLM [13]. We then follow the practice of judging generated answers in a pairwise fashion [14], prompting the judge LLM to select the better answer between two candidates generated by different RAG pipelines. (c.f. Section 6 with details of our pipelines).

Finally, to mitigate the lack of in-domain knowledge of the judging LLM, we also annotate the relevance of the documents retrieved by the pipelines being evaluated and inject the relevant documents in the context used by the judging LLM. This allows the judging LLM to better assess for hallucinations and completeness and better align the quality of the evaluations to those conducted by experts.

This process is mediated by RAGElo [1], a toolkit for evaluating RAG systems inspired by the Elo ranking system. RAGElo provides an easy-to-use CLI and Python library for using LLMs to evaluate retrieval results and answers produced by RAG pipelines. By combining a retrieval evaluator, a pairwise answer annotator, and an Elo-inspired tournament, RAGElo leverages powerful LLMs to agnostically annotate and rank different RAG pipelines. We notice that, although noisy, the LLM annotations generated by RAGElo are generally well aligned with experts' judgments of relative system quality, allowing for fast experimentation and comparisons between different RAG implementations without the frequent intervention of experts as annotators.

This paper evaluates multiple implementations of Infineon's retrieval augmented conversational agent using RAGElo: a traditional Retrieval-Augmented Generation and a RAG-Fusion implementation. RAG-Fusion generates multiple variations of the user question and combines the rankings produced by these queries into a more diverse set of documents. The documents are then fed into the LLM. We also analyze these same agents under a keyword-based retrieval regimen (i.e., the retriever uses BM25 to retrieve and rank documents), a dense retriever, and a hybrid retriever that combines the ranking generated by the BM25 and the dense retrievers using RRF. Our goal is to answer the following questions:

- Does the evaluation framework proposed by RAGElo align with the preferences of human annotators for answers generated by RAG-based conversational agents?
- Does the RAGF approach of submitting multiple variations of the user question and combining their rankings lead to better answers?

Table 1
Sample of questions submitted by users to the Infineon RAG-Fusion system

User-submitted queries
What is the country of origin of IM72D128, and how does geopolitical exposure affect the market and my SAM for the microphone?
What is the IP rating of mounted IM72D128?
Tell me microphones that have been released since January 2023 based on the datasheet revision history.
We need to confirm whether the IFX waterproof MIC has a sleeping mode and wake-up functions.

Table 2
Sample of synthetic queries for evaluating Infineon's RAG assistant. GPT4 refers to OpenAI's `gpt-4-turbo-2024-04-09` model. Opus, Sonnet and Haiku refer to Anthropic's Claude 3 models `opus-20240229`, `sonnet-20240229` and `haiku-20240307`, respectively.

model	Query
GPT4	What are some typical consumer applications for TLV496x-xTA/B sensors?
GPT4	What specific ISO 26262 readiness is available for the KP253 sensor?
Opus	How small of a form factor can I achieve for a battery-powered air quality device using Infineon's PAS CO2 sensor?
Sonnet	Can Infineon's sensors support bus configurations or daisy-chaining for simplified wiring and reduced complexity in IoT systems?
Haiku	Which TLE4971 current sensor models are available in the TISON-8-6 package?

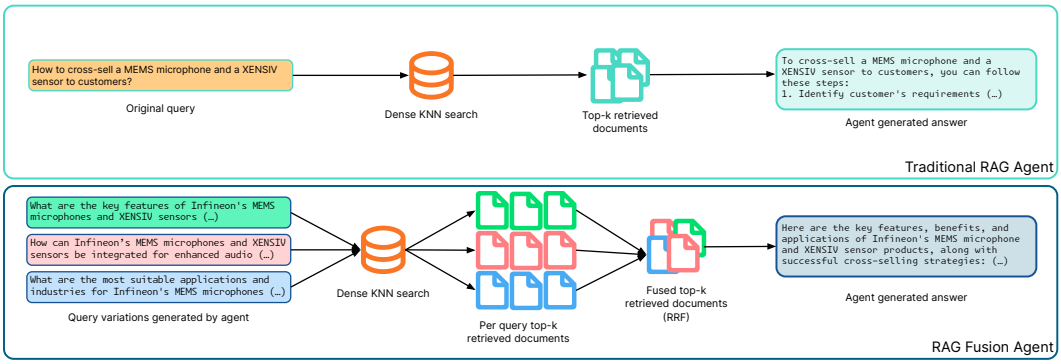

Figure 1: A traditional Retrieval-Augmented Generation pipeline compared to a RAG-Fusion pipeline. While a traditional RAG agent submits only the original query to the search system, a RAGF agent first generates variations of the user query and combines the rankings induced by these queries into a final ranking using RRF. The resulting top-k passages are fed into the LLM for generating the answer to the user's query.

2. Related Work

Several evaluation systems for RAG have been proposed to address flaws in current evaluation methods. For instance, Facts as a Function (FaaF) [15] is an end-to-end factual evaluation algorithm specially created for RAG pipelines. By creating functions from ground truth facts, FaaF focuses on the quality of generation and retrieval by calling LLMs. FaaF has substantially increased efficiency and cost-effectiveness, achieving reduced error rates compared to traditional evaluation methods. The reliance on a set of ground truths does not meet our goal of applying an automated evaluation toolkit to our pipelines. Recently, researchers have moved to eliminate the need for ground truths. This is especially important when automatically evaluating agents that retrieve highly technical documents from a large database, such as the Infineon RAGF conversational agent. RAGE1o eliminates this reliance by using an LLM-as-a-judge, a method studied in

[1] https://github.com/zetaalphavector/ragelo

numerous recent works.

SelfCheckGPT demonstrates the ability to leverage LLMs to detect and rank factuality with zero resources [16]. In addition, it has been demonstrated that GPT3.5 Turbo outperforms ground truth baselines in fact-checking with a "1/2-shot" method [17]. A model built to classify statements as true or false based on the activations of an LLM's hidden layers had up to 83% classification accuracy [18]. This evidence supports RAGElo's usage of LLM-as-a-judge.

Automated evaluation metrics can also be applied to RAG-based agents. BARTScore, an automated metric based on the BART architecture, has also outperformed most metrics on categories including factuality [19, 20]. Besides automated evaluation metrics, several automated evaluation frameworks have been created with a similar goal to RAGElo. Focusing on faithfulness, answer relevance, and content relevance, RAGAS leverages LLM prompting to focus on situations where ground truths and human annotations are not present in a dataset [21]. Prediction-powered inference aims to decrease the number of human annotations needed for machine learning prediction on a dataset of images of galaxies with approximately 300,000 annotations [22]. The ARES toolkit leverages prediction-powered inference to evaluate RAG systems with fewer human annotations. Like RAGElo, ARES automatically evaluates RAG systems using synthetically generated data [23].

ARAGOG highlights Hypothetical Document Embedding (HyDE) and LLM reranking as effective methods for enhancing retrieval precision while also exploring the effectiveness of Sentence Window Retrieval and the potential of the Document Summary Index in improving RAG systems [24].

While the aforementioned frameworks evaluate answers on relevance, faithfulness, and correctness metrics, RAG can also be evaluated on noise and counterfactual robustness, negative rejection, and information integration [25].

In addition to answers, frameworks have also been created to evaluate documents. Corrective Retrieval Augmented Generation (CRAG) builds on RAG by employing a retrieval evaluator to ensure that only the optimal documents are fed into the LLM prompt prior to the answer generation phase [26].

Due to its Elo-based ranking system for answers, its use of LLM-as-a-judge, and its relevance evaluation of the intermediate retrieval steps in a RAG pipeline, RAGElo is a unique evaluation toolkit. In this study, we use it to compare a simple RAG versus a more sophisticated RAGF system on a knowledge-intensive industry-specific domain.

3. Retrieval Augmented QA with rank fusion

While answers generated by traditional retrieval augmented systems are based on a number of documents retrieved from a single query, RAGF introduces additional variation into the retrieval process. Upon receiving a query from the user, a RAGF agent leverages a large language model to generate a set of queries based on the original [27]. Table 3 shows examples of queries generated by the agent based on the query, "How to cross-sell a MEMS microphone and a XENSIV sensor to customers?".

Table 3
Queries Generated from "How to cross-sell a MEMS microphone and a XENSIV sensor to customers?"

LLM-Generated Query
What are the key features of Infineon's MEMS microphones and XENSIV sensors that can be highlighted while cross-selling?
How can Infineon's MEMS microphones and XENSIV sensors be integrated for enhanced audio and motion sensing capabilities in various applications?
What are the most suitable applications and industries for Infineon's MEMS microphones and XENSIV sensors to maximize cross-selling potential?

After generating the variations for the user query, the RAGF agent submits the original and the generated queries to a retrieval system [28] that returns the top-k relevant documents $d, d_1, d_2, \ldots d_k$ from the set of all documents D for each query. The rankings induced by these queries are then combined using recriprocal rank fusion (RRF) [6] into a final, higher-quality set of passages. The intuition behind RAGF is that submitting variations of the same query and combining the final rankings increases the likelihood of relevant passages being injected into the LLM prompt. In contrast, non-relevant passages retrieved by a single query are discarded. Figure 1 describes how RAG and RAGF differ.

$$RRFScore(d \in D) = \sum_{r \in R} \frac{1}{r(d) + k}. \tag{1}$$

4. Development of a synthetic test set

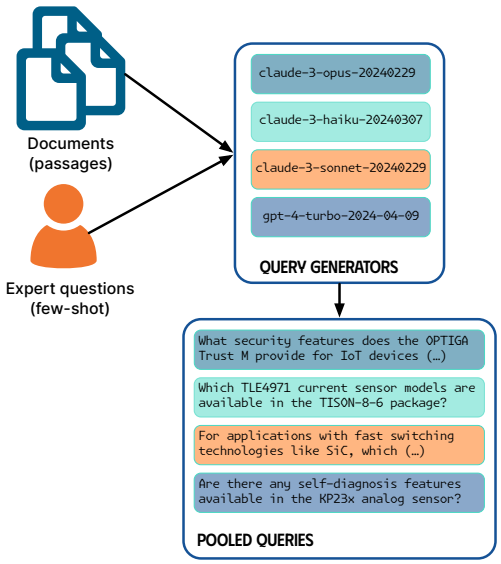

Figure 2: Process for creating synthetic queries. We prompt multiple LLMs to generate queries based on existing documents. We include some existing user queries in the prompt as few-shot examples.

As previously discussed, one of the main issues when evaluating the quality of a QA system in an enterprise setting is that, frequently, companies do not have a large enough existing collection of queries to evaluate such systems' quality. Therefore, in this work, we propose to adopt a strategy previously used by methods for generating synthetic queries for training retrieval systems, such as InPars [12] and Promptagator [29].

Similar to these approaches, we randomly sample passages from documents within our collection and prompt an LLM to generate questions that users may ask about these portions. However, one difference in our approach to generating training queries is the size of these passages. When generating queries for training a retrieval system, we ideally want to keep the passages short to fit in the dense encoder's relatively short context windows. However, when generating queries for evaluating QA systems (including retrieval augmented), we are not bound to the limit of the embedding model used for retrieval. Rather, a longer passage may yield questions that require multiple shorter passages to be answered. Therefore, we submit relatively long passages to the LLMs. Specifically, each passage is extracted from up to ten pages of PDF documents (about 2000 tokens [2])

To keep the questions generated as diverse as possible, we prompt four different LLMs to generate up to ten questions based on the same documents. Our test set collection contains a mix of queries generated by OpenAI's GPT-4 turbo [30] and Anthropic's Claude-3 [31] Opus, Sonnet, and Haiku models [3]. From a set of $N = 840$ queries, we sampled 200 queries across all four models. Half of the queries are selected from GPT-4 generated queries, and the other half from Claude 3 queries. Among the Claude 3 queries, to ensure the quality of the queries and their diversity, we again sample according to each model size. Ultimately, our test set contains 100 queries from GPT-4-turbo, 50 from Claude 3 Opus, 30 from Sonnet, and 20 from Haiku.

Finally, to increase the quality of the generated queries, We asked for an account manager, a sales operations specialist, a marketing representative, and a business development manager to create queries that they would submit to the conversational agent from the perspective of their role. They were instructed to produce queries regarding products from the XENSIV sensor product line, consisting of MEM microphones, radar, current, magnetic, pressure, and environmental sensors. We compiled a list of 23 of these queries to use as a base for experimentation and used them as few-shot examples in the query generation prompt. Figure 2 illustrates our method for generating synthetic queries based on existing user queries and document passages.

5. LLM-as-a-Judge for RAG pipelines

Even with a suitable set of synthetic questions for evaluating our RAG conversational agent, assessing whether a given answer properly answers a question is not trivially done. If a ground-truth "golden answer" is available, one can use traditional syntactic-based

[2]all LLMs used in our experiments had long context windows of 128k or 200k tokens.

[3]We did not use GPT-3.5 or open source models due to their shorter context window at the time of writing.

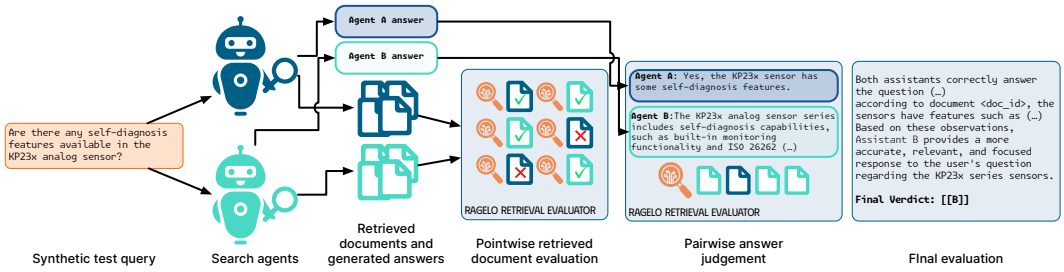

Figure 3: The RAGElo evaluation pipeline. First, documents retrieved by the agents are evaluated pointwise according to their relevance to the user's question. Then, the agents' answers are evaluated pairwise, using the retrieved relevant documents from both agents as reference.

metrics such as BLEU, METEOR or ROUGE [8, 9, 7]. Without such reference answers, one would require human annotators with a considerable understanding of the question's topic to manually assess the quality of the answers produced by each system. However, this is a costly process.

Alternatively, several LLM-as-a-Judge methods have been proposed, where another LLM is asked to evaluate the quality of answers generated by other LLMs. Nevertheless, in an enterprise setting, the answers usually require the LLM to access knowledge not present in their training datasets but rather contained in documents internal to the company. This is usually accomplished using a RAG pipeline like the one described above. Therefore, the judging LLM also needs access to similar knowledge to accurately evaluate the agent's answers' quality.

Therefore, in this work, we rely on RAGElo, an open-source RAG evaluation toolkit that evaluates the answers generated by each agent and the documents retrieved by them. By injecting the annotation of retrieved documents, pooled by the agents being evaluated, on the answer evaluation step, this method allows for the judging LLM to evaluate if the generated answer was able to use all the information available about the question properly and to check for any hallucinations. As the documents used for generating the answers are included in the answer evaluating prompt, an agent that incorrectly cites information from a source or refers to information not present in these documents is likely hallucinating and should have its evaluation adjusted accordingly. As we explore in Section 8, this two-step process results in a high correlation between human expert annotators and the judging LLM, enabling higher reliability and trust when evaluating different RAG pipelines. This process is also illustrated in Figure 3.

5.1. Evaluation aspects

While our main evaluation focuses on the pairwise comparison between the two agents, RAGElo also allows us to evaluate answers pointwise. In this setting, similar to other works [32], we prompt the judging LLM to evaluate the answers according to multiple criteria:

- Relevance: Does the answer address the user's question?

- Accuracy: Is the answer factually correct, based on the documents provided?
- Completeness: Does the answer provide all the information needed to answer the user's question?
- Precision: If the user's question is about a specific product, does the answer provide the answer for that specific product?

6. Retrieval pipelines

We not only experiment with different search agents (i.e., RAG and RAGF. We are also interested in how different retrieval methods may impact the quality of the final answers generated by these agents.

6.1. Retrieval methods

Our corpus consists of passages extracted from the Infineon XENSIV Product Selection Guide, a 117-page document with detailed information on every product in the XENSIV family. This document included technical information about all Infineon XENSIV sensors, consumer and automotive sensor applications, guidance in selecting the correct sensor, and other comprehensive and detailed information about the product line.

The passages are embedded using multilingual-e5-base [33] [4] and indexed using OpenSearch, allowing us to perform both KNN-based vector search, keyword-based search with BM25 [34], and RRF based hybrids thereof.

6.2. QA Systems Implementation

We mainly evaluate two agents: a naive RAG pipeline, where the agent first retrieves top-k passages that are then templated into a prompt, and the Infineon RAG-Fusion (RAGF) agent. Upon receiving a query, a naive RAG agent takes the following actions:

1. Retrieve the top k most relevant passages from the search system.
2. Perform a Chat Completions API call, prompting the LLM with instructions for generating an answer based on the five relevant passages.
3. Process and output the Chat Completions response.

Meanwhile, the Infineon RAGF conversational assistant uses a similar framework and performs the following steps upon receiving a query:

1. Perform a Chat Completions API call to generate four new queries based on the original query using a prompt tailored to the agent's original goal.
2. Retrieve the top k most relevant passages for each query.
3. Using RRF, combines the top-k passages induced by all queries into a final ranking.
4. Perform a Chat Completions API call prompting the LLM with carefully worded instructions for generating an answer based on the top-k fused passages
5. Process and output the Chat Completions response.

[4] https://huggingface.co/intfloat/multilingual-e5-base

7. Experiments

7.1. Comparing LLM-as-a-judge to expert annotators

While LLM-as-a-judge is a theoretically viable algorithm for rating RAG and RAGF answers, we must establish whether the results agree with the annotations of domain experts.

Figure 4 provides a Bland-Altman plot to visually represent the LLM and human judgments' agreement.

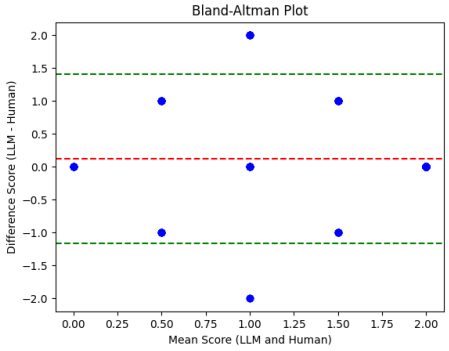

Figure 4: Bland-Altman plot to visualize the comparison between *LLM-as-a-judge* and expert answers.

The bias of approximately 0.12 indicates that, on average, LLM scores were slightly higher than human scores. The limits of agreement ranged from approximately -1.17 to 1.41. demonstrating substantial variability in the difference between LLM and human evaluators.

Next, we compared LLM-as-a-judge to expert annotators with Kendall's τ. Kendalls τ is a nonparametric measure that quantifies the degree of association between two monotonic continuous or ordinal variables by calculating the proportion of concordance and discordance among pairwise ranks, offering valuable insight into their rank correlation [35, 36]. We used the SciPy Stats Kendalltau function to calculate a tau-b score and a p-value for the combined ratings of all columns, flattened into a 1-D array with RAG and RAGF ratings combined [37]. The tau-b value, a nonparametric measure of association, is calculated using the following formula [38]:

$$\tau_b = \frac{(P - Q)}{\sqrt{(P + Q + T) \cdot (P + Q + U)}} \tag{2}$$

P represents the number of concordant pairs, Q represents the number of discordant pairs, T represents the number of ties exclusive to x, and U represents the number of ties exclusive to y.

This test returned $\tau \approx 0.56$, indicating a moderate, positive correlation [39] with a p-value against a null hypothesis of no association of $p < 0.01$ (99.99% confidence level). For comparison, in similar experiments judging human versus LLM judgments, Faggioli et al. found τ values of $\tau = 0.76$ and $\tau = 0.86$[40].

Following the same methodology, we also calculated Spearman's ρ, a similar nonparametric correlation measure. This resulted in $\rho \approx 0.59$ with $p < 0.01$, demonstrating a statistically significant, moderate positive correlation [36].

7.2. RAG vs RAGF

7.2.1. Quality of retrieved documents

We assessed document retrieval quality using Mean Reciprocal Rank@5 (MRR@5), which averages the inverse ranks of the first relevant result within the top five positions across all queries. The formula is given by

$$MRR@5 = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}, \tag{3}$$

where $|Q|$ is the total number of queries and rank_i is considered only if it's within the top five, otherwise it counts as zero [41].

MRR@5 scores were calculated for each agent and each retrieval method considering two categories:

1. MRR@5 score for documents deemed "somewhat relevant" or "very relevant."
2. MRR@5 score for documents deemed "very relevant."

The results can be seen below in Table 4.

Table 4
Mean MRR@5 scores for RAG vs RAG-F. The retrieval method columns indicate if the retrieval component used was vector search only (KNN), keywords only (BM25) or hybrid (KNN and BM25, combined with RRF).

Agent	Retrieval Method	Very Relevant	Somewhat Relevant
RAG	KNN	0.407	0.828
RAG	BM25	0.821	0.955
RAG	Hybrid	0.746	0.949
RAG-F	KNN	0.396	0.810
RAG-F	BM25	0.855	0.970
RAG-F	Hybrid	0.758	0.961

7.2.2. Pairwise evaluation of answers

We then ran RAGE1o games to evaluate end-to-end answer quality of RAG vs RAGF with different base retriever configurations a task that cannot rely on standard Information Retrieval metrics. These RAGE1o results show more victories for RAGF than RAG; For example, when using BM25 as a base retriever, RAGF won 49% of the games, RAG won 14.5%, and RAG and RAGF are tied in 36.5% of the times. The resulting Elo scores for all six variants are shown in table 6, which give a robust ranking of the systems, without reliance

on a gold standard. It is interesting to see, that for both RAGF as well as RAG, BM25 is a strong baseline that is not surpassed by generic embeddings in these experiments.

Next, we compared the RAGE1o outcome to the preference of our Infineon human annotator. We performed two-tailed paired t-tests to compare RAG against RAGF on each category from the Infineon representatives' human evaluations with $\alpha = .05$. As expected, due to its larger variety of retrieved results, RAGF significantly outperforms RAG in completeness at the 95% confidence level with $p \approx 0.01$ However, on the precision of answers, RAG significantly outperformed RAGF at the 95% confidence level with $p \approx 0.04$.

Table 5
RAG vs RAGF Win percentage between pairwise comparison of the agent's answers using GPT-4o as a judge with RAGE1o.

Agent		BM25		KNN		Hybrid		AVG
		RAG	RAGF	RAG	RAGF	RAG	RAGF	
BM25	RAG	—	14.5%	49.5%	52.5%	29.0%	28.5%	**34.8%**
	RAGF	49.0%	—	58.5%	51.5%	53.5%	30.5%	**48.6%**
KNN	RAG	33.0%	27.0%	—	20.0%	26.0%	31.0%	**27.4%**
	RAGF	34.5%	30.0%	37.0%	—	30.5%	32.0%	**32.8%**
Hybrid	RAG	41.5%	21.0%	51.5%	48.0%	—	20.5%	**36.0%**
	RAGF	46.0%	35.0%	49.0%	45.5%	43.5%	—	**44.3%**

Table 6
Elo Ranking for all agents averaged over 500 tournaments.

Agent	Retrieval	Elo score
RAGF	BM25	571.0
RAGF	Hybrid	550.0
RAG	Hybrid	497.0
RAG	BM25	487.0
RAGF	KNN	470.0
RAG	KNN	436.0

8. Discussion

As observed above, we found statistically significant, moderate positive correlations between LLM ratings and human annotations. This indicates a consistent association between the ratings from LLM-as-a-judge and those by Infineon experts. We find that on average, LLM scores are slightly higher than those of human annotators. This means that while relevance judgements on individual queries should not be fully reliable, and IR metrics derived from LLM-as-a-judge should not be equated with regular relevance scores without further calibration, we can still make good use of this approach to rank-order systems. These findings collectively support the validity of our LLM evaluation method, which assesses conversational system outputs based on a combination of relevance, accuracy, completeness, and recall.

The style of evaluation and the different dimensions it takes into account are specified in the prompts given to the LLM in the RAGE1o evaluation, which are provided in Appendix A. Specifically, while the initial LLM-as-a-judge is given specific criteria to focus on only four categories, we instructed RAGE1o's impartial judge LLM to value more than the initial four categories:

> Your evaluation should consider factors such as comprehensiveness, correctness, helpfulness, completeness, accuracy, depth, and level of detail of their responses.

Since RAGF significantly outperformed RAG in the completeness category, the RAGE1o judge LLM likely weighed completeness higher than precision. In addition, based on manual observation of a small random sample of answers, RAGF produced more comprehensive answers and featured higher depth and level of detail due to the multiple query generation. However, games where RAG won were most likely influenced by a significantly more precise answer than that of RAGF. While RAGF values comprehensive answers that offer multiple perspectives to the user, RAG produces shorter answers that answer the original query only. Since completeness is defined as the extent to which a user's question was answered, it can be presumed that RAGF's longer and more comprehensive answers may tend to be more complete. And since precision relates to the agent mentioning the correct product or product family, it can be presumed that RAGF's longer answers have more room to consider other products or product families, leading to reduced answer precision. While the human annotation was done by Infineon experts, different humans may rate answers differently, even if following the same set of criteria.

A larger number of documents or a database of non-technical documents may have led to a different outcome. RAGF can be applied to not only Infineon documents but also any documents database to retrieve. This includes not only enterprise uses but also uses in education, such as mathematics and language learning. The algorithm can be tuned to different use cases by tweaking the internal LLM prompt. For example, the Infineon RAGF bot was prompted to "think like an engineer." However, an educator RAGF bot could be prompted to "think like a teacher." Future work includes exploring other applications of RAGF, especially in education. In addition, we will experiment with different prompts for both LLM-as-a-judge and RAGE1o while using different quantities and types of documents with the same retrieval algorithms.

Based on the calculated MMR@5 scores, we found that the RAGF agent mostly outperforms the RAG agent in ranking both highly relevant and somewhat relevant documents retrieved. This evidence search on multiple query variants produced, on average, slightly more higher-ranked relevant documents than using only the original user query. We also see that using vector search with embeddings is not a silver bullet, as for our test queries, BM25 seriously outperforms it. Since retrieval quality is highly dependent on the quality of the embeddings and their fit to the domain, this outcome will likely be changed by fine-tuning the embeddings, and adding additional intelligent re-rankers, which we leave here for future work, as the evaluation framework would remain the same.

9. Conclusion

Overall, we found that the evaluation framework proposed by RAGElo positively aligns with the preferences of human annotators for RAG and RAGF with due caution due to a moderate correlation and variability of scoring. We found that the RAGF approach leads to better answers most of the time, according to the RAGElo evaluation. According to expert scoring, the RAGF approach significantly outperforms in completeness compared to RAG but significantly underperforms in precision compared to RAG. Based on these results, we cannot confidently assert that RAGF's approach leads to better answers generally. However, the results do support that RAGF's approach leads to more complete answers and a higher proportion of better answers under evaluation by RAGElo.

Since RAGElo is generally applicable to all retrieval-augmented algorithms, in future work, we also intend to test different agents other than RAG and RAGF, including those with different reranking algorithms, different embedding models, and different LLMs. In addition, due to RAGF's underperformance in document relevance, we may also leverage CRAG to reduce this gap. We will also investigate the reflection of human sensitivity in expert ratings, especially whether the LLMs should or can reflect human sensitivities.

Acknowledgments

We thank Brooks Felton from Infineon for his support during this work. We also thank the Infineon sales team for providing valuable feedback.

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, Language Models are Few-Shot Learners, 2020. doi:10.48550/arXiv.2005.14165. arXiv:2005.14165.

[2] Y. Xiao, W. Y. Wang, On Hallucination and Predictive Uncertainty in Conditional Language Generation, in: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, Association for Computational Linguistics, Online, 2021, pp. 2734–2744. doi:10.18653/v1/2021.eacl-main.236.

[3] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, P. Fung, Survey of Hallucination in Natural Language Generation, ACM Computing Surveys 55 (2023) 248:1–248:38. doi:10.1145/3571730.

[4] Z. Yin, Q. Sun, Q. Guo, J. Wu, X. Qiu, X.-J. Huang, Do large language models know what they dont know?, in: Findings of the Association for Computational Linguistics: ACL 2023, 2023, pp. 8653–8665.

[5] A. Amayuelas, L. Pan, W. Chen, W. Wang, Knowledge of knowledge: Exploring known-unknowns uncertainty with large language models, 2023. URL: https://arxiv.org/abs/2305.13712.

[6] G. V. Cormack, C. L. A. Clarke, S. Buettcher, Reciprocal rank fusion outperforms condorcet and individual rank learning methods, in: SIGIR 2019, SIGIR '09, Association for Computing Machinery, New York, NY, USA, 2009, p. 758759. URL: https://doi.org/10.1145/1571941.1572114. doi:10.1145/1571941.1572114.

[7] C.-Y. Lin, ROUGE: A package for automatic evaluation of summaries, in: Text Summarization Branches Out, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 74–81. URL: https://aclanthology.org/W04-1013.

[8] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu, Bleu: a method for automatic evaluation of machine translation, in: ACL 2002, ACL '02, Association for Computational Linguistics, USA, 2002, p. 311318. URL: https://doi.org/10.3115/1073083.1073135. doi:10.3115/1073083.1073135.

[9] A. Lavie, A. Agarwal, Meteor: an automatic metric for mt evaluation with high levels of correlation with human judgments, in: StatMT 2007, StatMT '07, Association for Computational Linguistics, USA, 2007, p. 228231.

[10] Z. Yang, A. Moffat, A. Turpin, Pairwise crowd judgments: Preference, absolute, and ratio, in: Proceedings of the 23rd Australasian Document Computing Symposium, ADCS '18, Association for Computing Machinery, New York, NY, USA, 2018. URL: https://doi.org/10.1145/3291992.3291995. doi:10.1145/3291992.3291995.

[11] N. Arabzadeh, C. L. A. Clarke, A comparison of methods for evaluating generative ir, 2024. URL: https://arxiv.org/abs/2404.04044.

[12] V. Jeronymo, L. Bonifacio, H. Abonizio, M. Fadaee, R. Lotufo, J. Zavrel, R. Nogueira, InPars-v2: Large Language Models as Efficient Dataset Generators for Information Retrieval, 2023. doi:10.48550/arXiv.2301.01820. arXiv:2301.01820.

[13] H. Huang, Y. Qu, J. Liu, M. Yang, T. Zhao, An empirical study of llm-as-a-judge for llm evaluation: Fine-tuned judge models are task-specific classifiers, 2024. URL: https://arxiv.org/abs/2403.02839.

[14] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, I. Stoica, Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL: https://arxiv.org/abs/2306.05685.

[15] V. Katranidis, G. Barany, Faaf: Facts as a function for the evaluation of generated text, 2024. arXiv:2403.03888.

[16] P. Manakul, A. Liusie, M. J. F. Gales, Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models, 2023. arXiv:2303.08896.

[17] T. Zhang, H. Luo, Y.-S. Chuang, W. Fang, L. Gaitskell, T. Hartvigsen, X. Wu, D. Fox, H. Meng, J. Glass, Interpretable unified language checking, 2023. arXiv:2304.03728.

[18] A. Azaria, T. Mitchell, The internal state of an llm knows when it's lying, 2023. arXiv:2304.13734.

[19] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019. arXiv:1910.13461.

[20] W. Yuan, G. Neubig, P. Liu, Bartscore: Evaluating generated text as text generation, in: M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J. W. Vaughan (Eds.), Advances in Neural Information Processing Systems, volume 34, Curran Associates, Inc., 2021, pp. 27263–27277. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/e4d2b6e6fdeca3e60e0f1a62fee3d9dd-Paper.pdf.
[21] S. Es, J. James, L. Espinosa-Anke, S. Schockaert, Ragas: Automated evaluation of retrieval augmented generation, 2023. arXiv:2309.15217.
[22] A. N. Angelopoulos, S. Bates, C. Fannjiang, M. I. Jordan, T. Zrnic, Prediction-powered inference, 2023. arXiv:2301.09633.
[23] J. Saad-Falcon, O. Khattab, C. Potts, M. Zaharia, Ares: An automated evaluation framework for retrieval-augmented generation systems, 2024. arXiv:2311.09476.
[24] M. Eibich, S. Nagpal, A. Fred-Ojala, Aragog: Advanced rag output grading, 2024. arXiv:2404.01037.
[25] J. Chen, H. Lin, X. Han, L. Sun, Benchmarking large language models in retrieval-augmented generation, 2023. arXiv:2309.01431.
[26] S.-Q. Yan, J.-C. Gu, Y. Zhu, Z.-H. Ling, Corrective retrieval augmented generation, 2024. arXiv:2401.15884.
[27] G. Fazlija, Toward Optimising a Retrieval Augmented Generation Pipeline using Large Language Model, Master's thesis, 2024.
[28] Z. Rackauckas, Rag-fusion: A new take on retrieval augmented generation, International Journal on Natural Language Computing 13 (2024) 3747. URL: http://dx.doi.org/10.5121/ijnlc.2024.13103. doi:10.5121/ijnlc.2024.13103.
[29] Z. Dai, V. Y. Zhao, J. Ma, Y. Luan, J. Ni, J. Lu, A. Bakalov, K. Guu, K. B. Hall, M.-W. Chang, Promptagator: Few-shot dense retrieval from 8 examples, 2022. arXiv:2209.11755.
[30] OpenAI, Gpt-4 turbo and gpt-4, 2024. arXiv:gpt-4-turbo-and-gpt-4.
[31] Anthropic, The claude 3 model family: Opus, sonnet, haiku, 2024. arXiv:Model Card Claude 3.pdf.
[32] P. Thomas, S. Spielman, N. Craswell, B. Mitra, Large language models can accurately predict searcher preferences, 2023. arXiv:2309.10621.
[33] L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, F. Wei, Multilingual e5 text embeddings: A technical report, 2024. arXiv:2402.05672.
[34] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, M. Gatford, Okapi at TREC-3, in: D. K. Harman (Ed.), Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994, volume 500-225 of NIST Special Publication, National Institute of Standards and Technology (NIST), 1994, pp. 109–126. URL: http://trec.nist.gov/pubs/trec3/papers/city.ps.gz.
[35] N. D. Edwards, E. de Jong, S. T. Ferguson, Graphing methods for kendall's τ, 2023. arXiv:2308.08466.
[36] S. Perreault, Efficient inference for kendall's tau, 2022. arXiv:2206.04019.
[37] scipy.stats.kendalltau, ???? URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.kendalltau.html#r4cd1899fa369-2.
[38] M. G. KENDALL, The treatment of ties in ranking problems, Biometrika 33 (1945)

239–251. doi:https://doi.org/10.1093/biomet/33.3.239.
[39] P. Schober, C. Boer, L. A. Schwarte, Correlation coefficients: Appropriate use and interpretation, Anesthesia & Analgesia 126 (2018) 1763–1768. doi:https://doi.org/10.1213/ane.0000000000002864.
[40] G. Faggioli, L. Dietz, C. L. A. Clarke, G. Demartini, M. Hagen, C. Hauff, N. Kando, E. Kanoulas, M. Potthast, B. Stein, H. Wachsmuth, Perspectives on large language models for relevance judgment, in: Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR 23, ACM, 2023. URL: http://dx.doi.org/10.1145/3578337.3605136. doi:10.1145/3578337.3605136.
[41] A. Jadon, A. Patil, A comprehensive survey of evaluation techniques for recommendation systems, 2024. arXiv:2312.16015.

A. RAGElo's prompts and configurations

A.1. Retrieval Evaluator

We used the default RAGElo's `ReasonerEvaluator`, which has the following system prompt:

```
You are an expert document annotator. Your job is to evaluate
    whether a document contains relevant information to answer a
     user's question.

Please act as an impartial relevance annotator for a search
    engine. Your goal is to evaluate the relevancy of the
    documents given a user question.

You should write one sentence explaining why the document is
    relevant or not for the user question. A document can be:
- Not relevant: The document is not on topic.
- Somewhat relevant: The document is on topic but does not
    fully answer the user question.
- Very relevant: The document is on topic and answers the user'
    s question.

[user question]
    {query}
[document content]
    {document}
```

A.2. Answer evaluators

For the pointwise evaluator used in Section 5.1, we used the following prompt with RAGElo's `CustomPromptAnswerEvaluator`:

```
You are an impartial judge for evaluating the quality of the
    responses provided by an AI assistant tasked to answer users
    ' questions about the catalogue of IoT sensors produced by
    Infineon.

You will be given the user's question and the answer produced
    by the assistant.
The agent's answer was generated based on a set of documents
    retrieved by a search engine.
You will be provided with the relevant documents retrieved by
    the search engine.
Your task is to evaluate the answer's quality based on the
    response's relevance, accuracy, and completeness.

## Rules for evaluating an answer:
- **Relevance**: Does the answer address the user's question?
- **Accuracy**: Is the answer factually correct, based on the
    documents provided?
- **Completeness**: Does the answer provide all the information
     needed to answer the user's question?
- **Precision**: If the user's question is about a specific
    product, does the answer provide the answer for that
    specific product?

## Steps to evaluate an answer:
1. **Understand the user's intent**: Explain in your own words
    what the user's intent is, given the question.
2. **Check if the answer is correct**: Think step-by-step
    whether the answer correctly answers the user's question.
3. **Evaluate the quality of the answer**: Evaluate the quality
     of the answer based on its relevance, accuracy, and
    completeness.
4. **Assign a score**: Produce a single line JSON object with
    the following keys, each with a single score between 0 and
    2, where 2 is the highest score on that aspect:
- "relevance"
    - 0: The answer is not relevant to the user's question.
    - 1: The answer is partially relevant to the user's
        question.
    - 2: The answer is fully relevant to the user's question.
- "accuracy"
    - 0: The answer is factually incorrect.
    - 1: The answer is partially correct.
```

```
    - 2: The answer is fully correct.
- "completeness"
    - 0: The answer does not provide enough information to
      answer the user's question.
    - 1: The answer only answers some aspects of the user's
      question.
    - 2: The answer fully answers the user's question.
- "precision"
    - 0: The answer does not mention the same product or
      product line as the user's question.
    - 1: The answer mentions a similar product or product line,
       but not the same as the user's question.
    - 2: The answer mentions the exact same product or product
      line as the user's question.

The last line of your answer must be a SINGLE LINE JSON object
   with the keys "relevance", "accuracy", "completeness", and "
   precision", each with a single score between 0 and 2.
[DOCUMENTS RETRIEVED]
{documents}
[User Query]
{query}
[Agent answer]
{answer}
```

For the pairwise evaluation between agents used for the results in Tables 5 and 6, we used `RAGElo`'s `PairwiseAnswerEvaluator` with the following parameters:

```
pairwise_evaluator_config = PairwiseEvaluatorConfig(
    n_games_per_query=15,
    has_citations=False,
    include_raw_documents=True,
    include_annotations=True,
    document_relevance_threshold=2,
    factors="the comprehensiveness, correctness, helpfulness,
       completeness, accuracy, depth, and level of detail of
       their responses. Answers are comprehensive if they show
       the user multiple perspectives in addition to but still
       relevant to the intent of the original question.",
)
```

This generates 15 random games between two agents per query (i.e., all possible unique games for 6 agents) and tells the evaluator that:

- The answers do not include specific citations to any passage (`has_citations=False`)

- Include the full text of the retrieved passages in the evaluation prompt (`include_raw_documents=True`)
- Inject the output of the retrieval evaluator into the prompt (`include_annotations=True`)
- Ignore any passage with a relevance score below 2 (`document_relevance_threshold=2`)
- Consider these factors when selecting the best answer `factors=…`)

These parameters produce the following final prompt used for evaluating the answers:

```
Please act as an impartial judge and evaluate the quality of
    the responses provided by two AI assistants tasked to answer
     the question below based on a set of documents retrieved by
     a search engine.

You should choose the assistant that best answers the user
    question based on a set of reference documents that may or
    may not be relevant.

For each reference document, you will be provided with the text
     of the document as well as reasons why the document is or
    is not relevant.

Your evaluation should consider factors such as
    comprehensiveness, correctness, helpfulness, completeness,
    accuracy, depth, and level of detail of their responses.
    Answers are comprehensive if they show the user multiple
    perspectives in addition to but still relevant to the intent
     of the original question.

Details are only useful if they answer the user's question. If
    an answer contains non-relevant details, it should not be
    preferred over one that only uses relevant information.

Begin your evaluation by explaining why each answer correctly
    answers the user's question. Then, you should compare the
    two responses and provide a short explanation of their
    differences. Avoid any position biases and ensure that the
    order in which the responses were presented does not
    influence your decision. Do not allow the length of the
    responses to influence your evaluation. Be as objective as
    possible.
```

```
After providing your explanation, output your final verdict by
   strictly following this format: "[[A]]" if assistant A is
   better, "[[B]]" if assistant B is better, and "[[C]]" for a
   tie.

[User Question]
{query}

[Reference Documents]
{documents}

[The Start of Assistant A's Answer]
{answer_a}
[The End of Assistant A's Answer]

[The Start of Assistant B's Answer]
{answer_b}
[The End of Assistant B's Answer]
```