How to moderate LLM based chats from hallucinations?

Jakub Rurański
1. Departament of Algoritmics and Software, Silesian University of Technology
Gliwice, Poland
2. TELE-FONIKA Kable S.A
Myślenice, Poland *jakurur124@student.polsl.pl*

Katarzyna Nowak
1. Departament of Algoritmics and Software, Silesian University of Technology
Gliwice, Poland
2. TELE-FONIKA Kable S.A
Myślenice, Poland *katanow108@student.polsl.pl*

Magdalena Góras
1. Departament of Algoritmics and Software, Silesian University of Technology
Gliwice, Poland
2. TELE-FONIKA Kable S.A
Myślenice, Poland *magdgor933@student.polsl.pl*

Karol Dworak
TELE-FONIKA Kable S.A
Myślenice, Poland *karol.dworak@tfkable.com*

Karolina Nurzyńska
1. Departament of Algoritmics and Software, Silesian University of Technology
Gliwice, Poland
2. TELE-FONIKA Kable S.A
Myślenice, Poland *Karolina.Nurzynska@polsl.pl*

Abstract

Chatbots powered by large language models (LLMs) are increasingly prevalent in various domains. Nonetheless, they face challenges such as hallucinations and losing context during extended conversations. This study tackles these issues by proposing a multi-agent strategy for chat architecture where multiple LLMs focus on distinct tasks to enhance the quality of their output. The suggested solution involves a supervisor agent working in conjunction with a document search and review module. We assess the performance of information systems with chatbots designed to respond to sustainability questions in English and handle technical documentation for plant equipment in Polish. A comprehensive analysis of commercial and open-source models revealed that Qwen2.5 v14b's performance is comparable to that of the Gemini family models.

Keywords: large language models, chatbots, prompt engineering, hallucination.

1. Introduction

Chatbots utilising large language models (LLMs) augmented by external document retrieval, known as Retrieval-Augmented Generation (RAG), have significantly transformed how people and businesses leverage technology across various fields. From education to healthcare, marketing, and industry, chatbots enhance efficiency by providing customised learning support [15] and assisting healthcare diagnostics and patient management [18].

In business, chatbots have become essential for automating customer service, analysing preferences, suggesting products, and guiding purchases, enhancing customer experience and efficiency [10]. Industrial sectors are increasingly adopting AI-powered chatbots to manage distributed technical knowledge and documentation, enabling faster decision-making and knowledge sharing [7]. While chatbot applications are extensive, their effectiveness relies on generating responses solely from the provided documents.

A significant challenge is hallucination, where chatbots generate plausible but unconfirmed answers, undermining trust and potentially causing serious consequences, especially in medical applications [12], [18]. Another critical issue is long-context degradation, where models fail to maintain an accurate reference to the provided documents, resulting in responses that are disconnected from the relevant content.

The proposed architecture mitigates traditional RAG limitations by distributing tasks such as query handling, document retrieval, and answer generation among dedicated modules. Moreover, by comparing English and Polish documentation scenarios, we evaluate commercial and open-source LLMs to determine the most effective models for multilingual technical applications.

2. Materials & Methods

Developing intelligent, adaptable, and robust RAG systems remains a significant challenge [16]. Our proposition's primary motivation for using multi-LLM architecture was to overcome the limitations of early LLMs, particularly their struggles with long-context retention and understanding [6]. During our initial testing, we observed that when presented with extensive technical documentation, LLMs frequently hallucinated, lost track of the prompt, or refused to answer. To address these challenges, we implemented a modular, multi-agent approach that distributes key responsibilities across specialised models. The following sections give detailed information concerning the proposed solution.

2.1. Multi-Agent RAG Approach

A common pitfall in traditional LLM-based RAG systems is task overload — expecting a single model to handle user interaction, information retrieval, and structured answer generation. Inspired by human teamwork, we hypothesised that breaking these tasks into distinct roles and assigning them to different models would improve overall performance [25]. Our system consists of three primary modules: supervisor agent (user interaction & question processing), document search and review module, and answer generation node. Each module is tailored to its specific function and optimised using dedicated prompts.

2.2. Supervisor Agent: Managing User Interaction and Query Refinement

To ensure seamless user interaction, we implemented a supervisor agent as the first node in our system. Its core responsibilities include: maintaining conversation context and handling general queries, determining whether a question requires document-based retrieval, and refining ambiguous or incomplete user queries before forwarding them.

Inspired by Adaptive RAG [11], the supervisor agent autonomously requests clarification for ambiguous queries before passing refined questions to the retrieval module.

2.3. Document Search and Review Module

This module aims to retrieve, filter, and structure relevant document fragments to maximise their utility in answer generation. This process involves several key steps:

Multiple Query Reformulation Users often pose questions imprecisely, assuming the chat-

bot understands their implicit context. To mitigate this, we employ a smaller LLM to reformulate the query based on a summarisation of all available documents [26]. This ensures the retrieval process accounts for domain-specific nuances and missing details.

Vector Search and Fragment Retrieval We use a vector database (ChromaDB [4]) for semantic search with the documents preprocessed and divided into fixed-size chunks [22]. Our findings indicate that changing the preprocessing in the technical document case study by using a parsing approach powered by an LLM [21] is more efficient. Subsequently, the user query is embedded with the same model applied during preprocessing. At this stage, we retrieve the top 20 document fragments via hybrid similarity search enhanced with Maximal Marginal Relevance (MMR) [3], a method that promotes diversity while minimising redundancy among retrieved results.

Post-Retrieval Filtering and Context Optimisation first removes duplicate or near-duplicate passages and re-ranks the remaining fragments with a relevance-plus-diversity objective inspired by MMR and its recent chunk-level variant, Chunk-RAG [24]. Next, we apply a long-context reordering strategy to counter the "lost-in-the-middle" effect documented by [17]. Finally, following the hierarchical XML flattening of *LongRefiner* [13], we serialise the selected fragments into a lightweight XML-like format (only ‘<document filename="example.pdf">‘ tags). This structure enhances parsing efficiency and facilitates the answer-generation agent's ability to trace provenance.

LLM Contextual Compressor The retrieved context is reviewed and evaluated by an LLM that rewrites the documents into well-structured source material for the Answer Generation Node [14]. This approach allows us to feed the LLM responsible for accurately answering the user's query with only the relevant information.

2.4. Answer Generation Node

Once the post-filtered context arrives, the final agent selects an answer style—a numbered procedure for shop-floor staff, a troubleshooting checklist, or a brief compliance summary—and generates the reply. Prompt quality proved decisive: starting from a minimal template, we iteratively edited the text and re-ran all evaluation metrics while keeping the rest of the pipeline fixed. The final prompt contains four blocks:

1. **Role block**—frames the model as a domain expert who must rely strictly on the provided documents.
2. **Goal block**— states the desired output.
3. **Guideline block**—lists hard constraints on source-faithfulness, language, style, markdown tables, and handling data gaps.
4. **Few-shot examples**—two Input→Output pairs (long procedural, short informational) that demonstrate the target structure and tone.

2.5. Testing scenarios

The proposed chatbot architecture was evaluated by solving tasks in two scenarios. The first scenario utilised a dataset of 30 documents (1,353 chunks) in English, which collected company data on sustainability procedures and facts. It was also supplied with a set of probable questions and suggested answers. The chatbot's goal was to prepare an answer containing all the information from this set. When questions reflect those from the Q&A set, it should answer with similar content.

In the second scenario, a set of technical documentation, including manuals and other necessary information, was considered to enable new employees to operate the equipment efficiently

in the factory. It contained 101 files (1306 chunks) all in Polish. In this task, a chatbot should serve as a guide, instructing the worker on what to do in all situations within the plant. The answers needed to be precise and correct, as any mistake may result in injury or significant business losses. The datasets of queries and expected responses were developed collaboratively with industry technologists and domain specialists, ensuring high practical relevance and challenging technical specificity. In both cases, the dataset contains 20 examples.

The chatbot was evaluated on two virtual machines. The first, used for testing open-source models, featured 48 vCPUs, 192 GB of RAM, and four NVIDIA L4 GPUs (totalling 96 GB of VRAM). The second, used for testing Gemini models, was a smaller 8 vCPU instance with 32 GB of RAM, as inference was handled externally via the API provider.

2.6. Evaluation Methods

The open-source tool DeepEval [27] was employed to evaluate the quality of responses generated by the LLMs. This modern evaluation environment enables a more precise and context-sensitive assessment of the LLM output. The applied metrics provide a multidimensional evaluation based on factual accuracy, faithfulness to context, contextual relevance, and precision. DeepEval, which follows the LLM-as-a-Judge paradigm [9], proved to be a reliable and efficient tool for automated evaluation in our experiments. The following metrics were used in the evaluation [23]:

Metric Score — a general measure of the relevance and quality of the generated answer in relation to the input query and context. The metric provides both a numerical score and an explanation of the rating.

Correctness Metric Score — based on the G-Eval approach with a chain-of-thought (CoT) strategy, this metric assesses the factual correctness of the output. It considers contradictions with the expected answer, missing key details, and tolerance for generic or subjective language, as long as factual accuracy is maintained.

Contextual Metric Score — evaluates how effectively the model uses contextual information (particularly in RAG systems), focusing on whether the most relevant content was prioritised and integrated into the answer.

Faithfulness Metric Score — measures the degree to which the generated response reflects the source context accurately, aiming to detect hallucinations or misrepresentations.

Contextual Precision Score — similar to the Contextual Metric Score, but emphasising the precision of selected contextual elements (i.e., key informational nodes).

2.7. Selected Model Description

The chosen Gemini family models represent different developmental versions of the Gemini API, available through the Google AI Studio platform:

- **Gemini-1.5 Pro (gemini-1.5-pro-002)** — a stable release with a high input context window (up to 2,097,152 input tokens and 8192 output tokens).
- **Gemini-2.0 Flash-001** — a stable, optimised version with a smaller input limit (1,048,576 input tokens and 8192 output tokens).
- **Gemini-2.0 Pro** — an experimental version, derived from *Gemini 2.0 Pro Experimental*, currently replaced by *Gemini 2.5-exp* [8].

We evaluated multiple open-source models, notably combining Deepseek-R1-32 with Phi-4 for better coherence [19]. Key model characteristics:

- **Deepseek-R1-32** is a reinforcement learning-trained model based on the Qwen-32B architecture. It was developed without traditional supervised fine-tuning, relying instead on large-scale RL-based optimization [1].

- **Phi-4** is a small language model, designed for efficient task-oriented inference. It was integrated with Deepseek to enhance output consistency and factual correctness [19].
- **Qwen2 5_32B** is a large language model developed by Alibaba Cloud, supporting up to 128,000 input tokens and generating outputs up to 8,000 tokens. It is part of the Qwen2.5 family and has been optimised for multilingual and multi-domain tasks [5].
- **Blossom-v5_14B** is a model built on top of Qwen-2, capable of handling context windows up to 32,768 tokens. It was developed by an independent researcher, Azure99, with a focus on long-context comprehension and accuracy [2].
- **Bielik-11B-v2.3-Instruct** is an 11B-parameter language model developed by SpeakLeash and Cyfronet AGH, optimised for Polish NLP tasks. It merges earlier Bielik versions and uses instruction tuning. It scored 8.56 on Polish MT-Bench and 65.71 on the Open PL LLM Leaderboard . [20].

3. Results

A comprehensive evaluation was conducted using an English-language and Polish-language dataset to demonstrate the effectiveness and versatility of our multi-agent RAG architecture. Our evaluation aims not merely to compare standalone model performance, but specifically to assess whether our modular multi-agent architecture improves the performance of open-source models to match or approach that of commercial counterparts. Below results gathered in each case study are presented.

3.1. Evaluation of English-Language Document Processing

We started the experiments with comparative tests conducted on a wide range of language models, including open-source solutions and commercial models provided by Google (Gemini). This allowed us to see the differences in the performance of various accessible solutions and determine their applicability to the described tasks. Tests were repeated seven times, reporting the average results. Tables 1 and 2 contain an average of 20 evaluated questions for the Gemini and open-source models, respectively.

As the results show, it is difficult to determine the winner, as models resulting well in one score (e.g., Gemini-1.5-flash-002, Gemma) are outperformed by others when different criteria are considered (e.g., Gemini-1.5-pro-002, Deepseek-r1-32-phi4). In the case of the Gemini family, the results are very similar; however, more noticeable differences emerge when open-source solutions are taken into consideration. Nevertheless, there are open-source solutions which can easily replace the paid versions. Each group's three best models (marked in bold) were selected for in-depth analysis of the results.

Figure 1 presents all evaluation metrics' mean values and standard deviations for the three selected models. Based on this, we can see that Gemini-1.5 Pro-002 and Gemini-2.0 Flash-001 achieved very similar results across most metrics, particularly in overall accuracy (*Metric Score*) and *Faithfulness Metric Score*. Both models demonstrated similar levels of stability, with relatively low and comparable standard deviations, suggesting consistent performance across

Table 1. Comparison of Gemini models

Model	Metric Score	Correctness	Contextual	Faithfulness	Precision
Gemini-1.5-flash-002	0.96	0.85	0.62	0.99	0.48
Gemini-1.5-pro-002	0.96	0.85	0.44	0.91	0.88
Gemini-2.0-flash-001	0.96	0.83	0.42	0.94	0.90
Gemini-2.0-pro	0.96	0.83	0.42	0.94	0.90
Gemini-1.5-pro-001	0.92	0.85	0.69	0.98	0.46

Table 2. Comparison of open-source models

Model	Metric Score	Correctness	Contextual	Faithfulness	Precision
Gemma	1.00	0.83	0	1.00	0
Nemotron-mini	0.99	0.80	0	0.99	0
Deepseek-r1-32-phi4	0.96	0.82	0.32	0.98	0.90
Jais-adaptive-7b	0.96	0.75	0	1.00	0
Hermes3	0.95	0.88	0	1.00	0
Blossom-v5_14b	0.95	0.77	0.23	0.94	0.90
Qwen2 5_32b	0.95	0.74	0.32	0.92	0.85
Phi4-64k	0.92	0.82	0.36	0.99	0.32
Deepseek-coder-v2-32k	0.82	0.78	0.35	1.00	0.47
Mistral	0.71	0.59	0.65	1.00	0.57
Zephyr	0.69	0.61	0.63	1.00	0.48
Llama3-chatqa-70b	0.66	0.58	0.36	0.94	0.89
Llama-3.2	0.62	0.49	0	1.00	0
Llama3-chatqa-8b	0.58	0.50	0.34	0.98	0.93
Gemma2_27b	0.54	0.48	0.31	0.91	0.92
Llama3 1.8b-64k	0.45	0.33	0.56	1.00	0.53

queries. The Gemini-2.0 Pro, although promising, scored lower across all metrics and exhibited a noticeably higher variance, indicating less stability and potential issues with reliability in this version. Detailed analysis of individual test queries (see Figure 2) reveals that neither Gemini-1.5 Pro nor Gemini-2.0 Flash-001 consistently outperforms the other. Performance varies de-

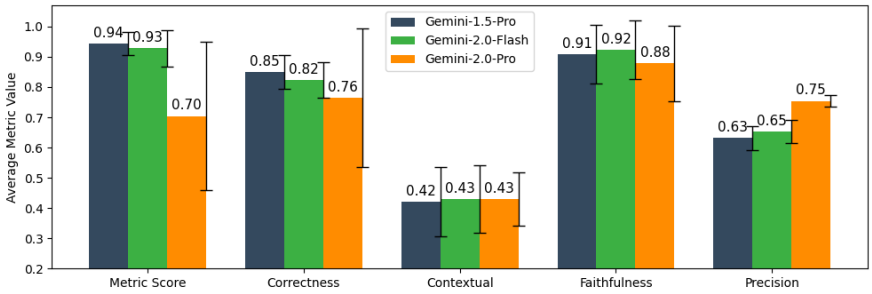

Fig. 1. Comparison of average metric values for Gemini models with standard deviations

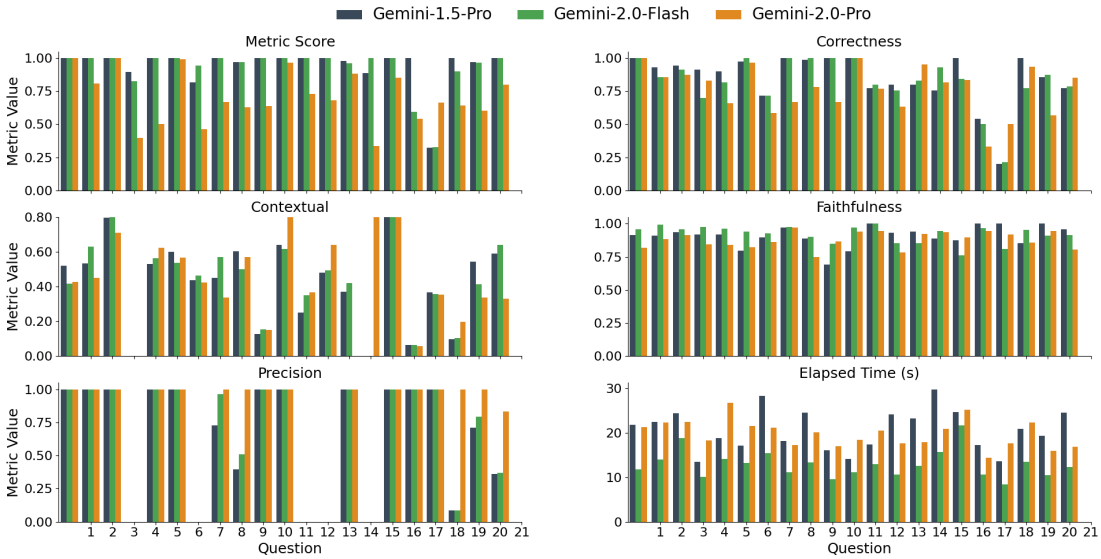

Fig. 2. Metric values per test query for Gemini models

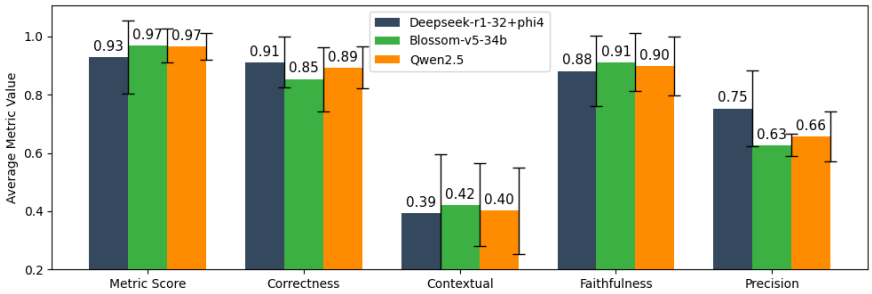

Fig. 3. Average metric scores for open-source models with standard deviations

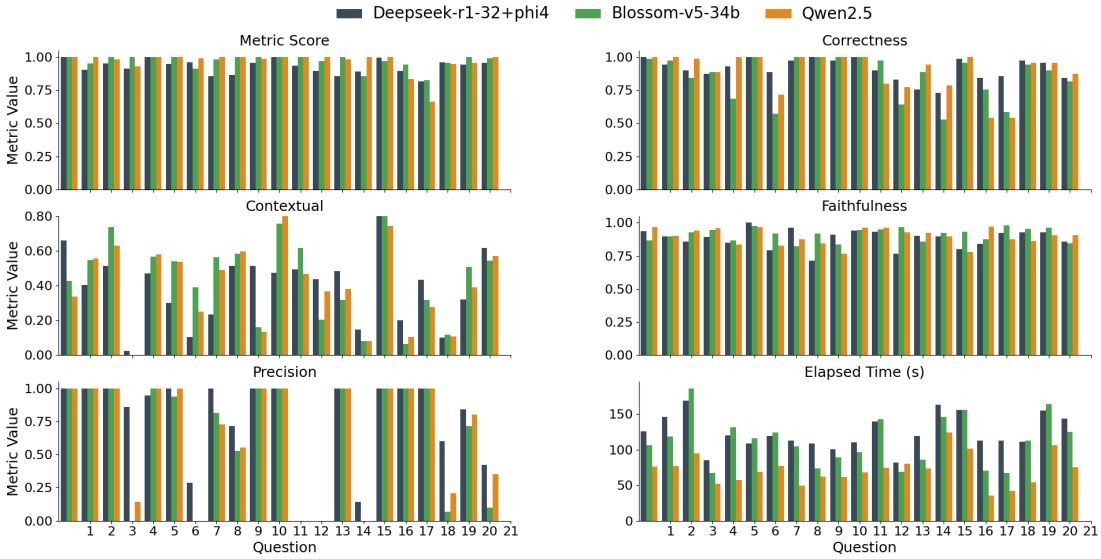

Fig. 4. Evaluation metric results per query for open-source models

pending on the query, confirming that both models are generally comparable, with strengths in different scenarios.

Figure 3 presents a comparison of the average values and standard deviations of all evaluation metrics for the three selected open-source models. All of them demonstrated a comparable overall performance — the average *Metric Score* and *Correctness Metric Score* exceeded 0.90 in all cases. One key difference lies in stability — the combination **Deepseek-R1-32 + Phi4** exhibited noticeably higher standard deviations than the other two models, suggesting a greater variance in performance depending on the query. Despite this, Deepseek + Phi4 sometimes produced highly accurate and relevant responses, indicating potential that could be unlocked through further tuning and integration work.

It is important to note that the system architecture and prompts were initially optimised specifically for Gemini models through iterative refinement, adjusting prompts and evaluating their impact. Future research should explore creating dedicated configurations tailored explicitly to individual open-source models. Such tailored optimisations could substantially enhance their evaluation scores and overall system performance.

The detailed per-query results presented in Figure 4 confirm the following observations: The Deepseek-R1-32 + Phi4 combination showed substantial variability — it performed very well for specific queries and considerably worse for others, reflecting inconsistent behaviour under a unified prompt strategy. In the case of Blossom-v5 and Qwen2.5, their performance appears comparable. However, the results vary depending on the specific query. Regarding processing speed, Qwen2.5 stands out as the fastest of the three, which may be a crucial advantage in latency-sensitive applications.

Table 3. Average response time for each model

Model	Average Time (s)	Standard Deviation
Gemini-2.0 Pro	19.81	2.39
Gemini-1.5 Pro	20.68	1.94
Gemini-2.0 Flash	12.93	1.79
Deepseek-R1-32 + Phi4	123.83	23.98
Qwen2.5	71.85	10.383
Blossom-v5	111.94	23.59

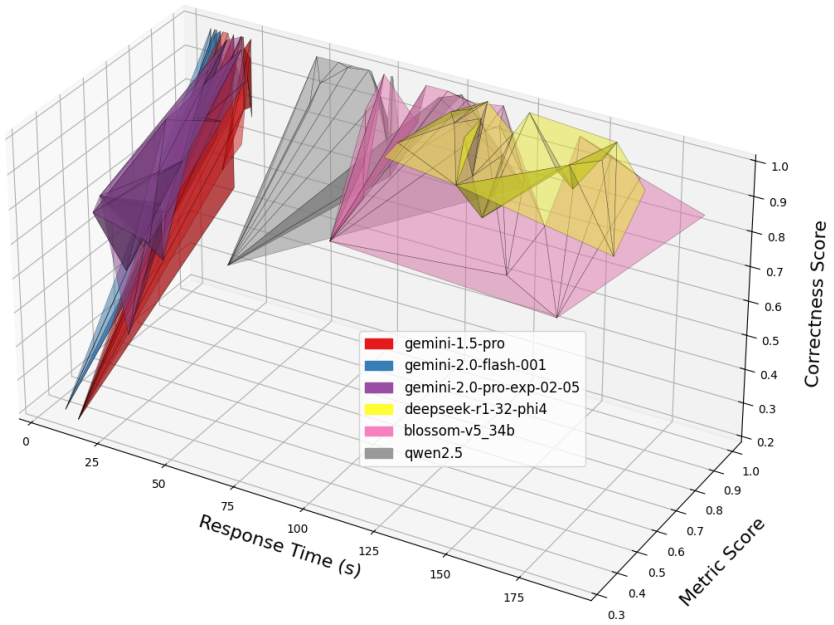

Fig. 5. Model comparison by response time, general relevance (Metric Score), and factual correctness (Correctness Score). We can see that the Gemini models give rapid answers, but their relevance and correctness vary more when compared to the open-source models. Yet, the open-source model's performance is slow.

The final analysis included a comparison of all six models — both commercial and open-source — in terms of key evaluation metrics and average response time. Table 3 summarises the average time required to generate a response for each model. These results clearly show that open-source models — particularly Blossom-v5 and Deepseek-R1-32 with Phi4 — have significantly longer response times than Gemini models. The exception is Qwen2.5, which achieved times similar to the commercial models, making it the most efficient among the tested open-source models.

Figure 5 gives more detailed insight into the differences between the compared models, where the three most variable metrics were considered (Metric Score, Correctness Metric Score and Response Time) and plotted for each test sample separately. The visualisation reveals that the top-performing models with balanced response time and high-quality output are: **Gemini-1.5 Pro**, **Gemini-2.0 Flash**, and **Qwen2.5**. Notably, upon manual inspection of generated answers, Qwen2.5 demonstrates comparable coherence and factual accuracy to the Gemini models.

Thus, the final choice of model depends on user priorities — if response speed and commercial support are critical, Gemini models may be more attractive. However, for those seeking a high-performing open-source alternative — Qwen2.5 proves to be a strong and viable option.

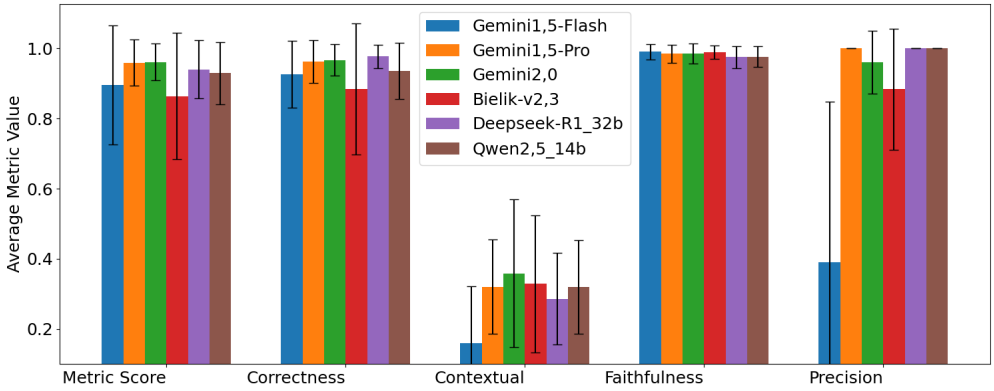

Fig. 6. Overall performance comparison across all evaluated models

3.2. Evaluation of Polish-Language Document Processing

In this case study, the primary objective was to investigate whether our modular RAG system could enhance the performance of open-source LLMs to levels comparable to Google's state-of-the-art Gemini models, particularly given the inherent challenges associated with processing technical content in Polish. The comparative analysis concentrated on the best models defined in the previous task and included Bielik v2.3 11B, a model optimised explicitly for Polish-language content. Gemini models were tested six times, and open-source models three times, due to computational constraints.

The results of this evaluation are summarised in Figure 6. Notably, our multi-agent RAG architecture successfully narrowed the performance gap typically observed between open-source and commercial models. Gemini 1.5 Pro and Gemini 2.0 Flash achieved superior consistency, indicated by lower standard deviations, confirming their reliability in technical Polish-language applications. Notably, the open-source model Deepseek R1 32B achieved accuracy metrics on par with those of Gemini models, yielding the highest correctness and contextual precision scores among open-source options. Bielik v2.3, despite its smaller parameter size, showed remarkable performance considering its targeted linguistic optimisation for Polish. However, it exhibited significant variability in response time, highlighting potential stability issues.

Our research revealed that optimising prompts and interactions explicitly for the Polish language significantly improved the performance of open-source models, as shown in Figure 7. This linguistic adaptation was particularly beneficial for Bielik, enabling it to effectively leverage its strengths in processing Polish-language contexts. Additionally, our multi-step retrieval and refinement process, combined with specialised multilingual embeddings, substantially mitigated the typical weaknesses of generic embedding models in handling Polish text.

Although open-source models exhibited competitive accuracy, they generally required longer response times due to computational limitations in the testing environment. Figure 8 illustrates the trade-offs between response speed and output quality. Qwen2.5 stood out by balancing

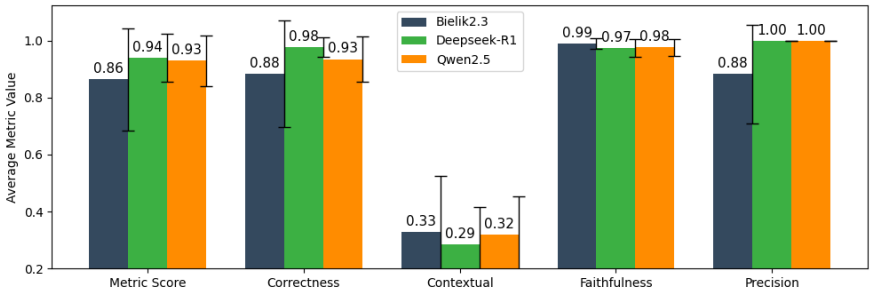

Fig. 7. Average metric scores for open-source models after Polish-language optimisation

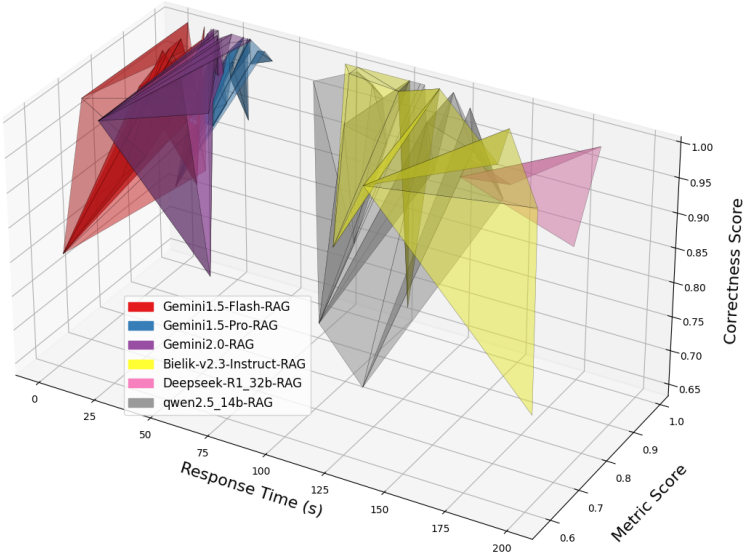

Fig. 8. Comparison of response time versus answer quality and correctness across models. A few erratic outliers were excluded for clarity, in the case of the Bielik model, which exhibited unusually high response times (>400s) in select instances. This visualisation underscores that the Gaminis models give rapid answers when compared to open-source data. While the relevance and correctness are comparable in the evaluated models.

speed and quality effectively, showing promise as a viable alternative to commercial models in resource-sensitive deployments.

Overall, the Polish-language case study underscores the robustness and adaptability of our modular RAG architecture. By carefully orchestrating distinct roles among specialised agents, our approach significantly elevated the performance of open-source LLMs, affirming their potential as cost-effective alternatives to proprietary solutions, especially in multilingual technical applications.

4. Conclusion

The presented multi-agent RAG system demonstrated significant advancements in addressing key limitations of traditional single-model chatbot solutions, particularly in managing technical and multilingual documentation. Through the modular distribution of responsibilities — user interaction management, context retrieval optimisation, and structured answer generation — our approach effectively mitigated common issues such as model hallucinations and context loss.

We significantly improved system robustness and answer accuracy by structuring our RAG pipeline into specialised, interconnected modules. Our evaluations highlight the importance of structured input processing, query refinement, and context optimisation in enhancing RAG performance. These findings underscore the importance of task-specific large language model (LLM) orchestration when designing AI-driven knowledge assistants, particularly in technical and multilingual domains.

The performance of various LLMs was evaluated in two distinct case studies that considered English and Polish texts. We compared the commercial (Google Gemini) with open-source models and concluded that the differences between Gemini family networks are negligible in the presented task settings. More differences were visible within open-source solutions, whose performance is usually not as good as that of commercial solutions, except for Qwen2.5 v14b whose performance is comparable.

Future research could benefit from enhancing model stability, particularly by addressing the response time variability observed in smaller, specialised models like Bielik. Further optimi-

sation of multilingual embeddings and exploration of hybrid architectures combining various open-source models may also provide avenues for continued improvement in RAG system performance and applicability.

References

[1] AI, D.: Deepseek-r1 distill qwen 32b. https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B (2024), accessed: 2025-03-28

[2] Azure99: Blossom-v5_14b model listing. https://llm.extractum.io/list/?mtr=Azure99 (2024), accessed: 2025-03-28

[3] Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98). pp. 335–336. Association for Computing Machinery, Melbourne, Australia (1998), https://doi.org/10.1145/290941.291025

[4] Chroma: Chroma – the ai-native open-source embedding database (2024), https://trychroma.com/, accessed: 2025-05-27

[5] Cloud, A.: Qwen2.5 on ollama. https://ollama.com/library/qwen2.5 (2024), accessed: 2025-03-28

[6] Cui, H., Shamsi, Z., Cheon, G., Ma, X., Li, S., Tikhanovskaya, M., Norgaard, P., Mudur, N., Plomecka, M., Raccuglia, P., Bahri, Y., Albert, V.V., Srinivasan, P., Pan, H., Faist, P., Rohr, B., Statt, M.J., Morris, D., Purves, D., Kleeman, E., Alcantara, R., Abraham, M., Mohammad, M., VanLee, E.P., Jiang, C., Dorfman, E., Kim, E.A., Brenner, M.P., Jain, V., Ponda, S., Venugopalan, S.: Curie: Evaluating llms on multitask scientific long context understanding and reasoning (2025), https://arxiv.org/abs/2503.13517

[7] Freire, S.K., Wang, C., Foosherian, M., Wellsandt, S., Ruiz-Arenas, S., Niforatos, E.: Knowledge sharing in manufacturing using large language models: User evaluation and model benchmarking (2024), https://arxiv.org/abs/2401.05200

[8] Google: Gemini api model reference (2024), https://ai.google.dev/gemini-api/docs/models?hl=pl, accessed: 2025-03-28

[9] Gu, J., Jiang, X., Shi, Z., Tan, H., Zhai, X., Xu, C., Li, W., Shen, Y., Ma, S., Liu, H., Wang, S., Zhang, K., Wang, Y., Gao, W., Ni, L., Guo, J.: A survey on llm-as-a-judge (2025), https://arxiv.org/abs/2411.15594

[10] Huseynov, F.: Chatbots in digital marketing. In: Advances in Marketing, Customer Relationship Management, and E-Services Book Series, pp. 46–72. Publisher Name (if known, replace this) (2023), https://doi.org/10.4018/978-1-6684-7735-9.ch003

[11] Jeong, S., Baek, J., Cho, S., Hwang, S.J., Park, J.C.: Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity (2024), https://arxiv.org/abs/2403.14403

[12] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y.J., Madotto, A., Fung, P.: Survey of hallucination in natural language generation. ACM Computing Surveys 55(12), pp. 1–38 (Mar 2023), http://dx.doi.org/10.1145/3571730

[13] Jin, J., Li, X., Dong, G., Zhang, Y., Zhu, Y., Wu, Y., Li, Z., Ye, Q., Dou, Z.: Hierarchical document refinement for long-context retrieval-augmented generation (2025), https://arxiv.org/abs/2505.10413

[14] Joren, H., Zhang, J., Ferng, C.S., Juan, D.C., Taly, A., Rashtchian, C.: Sufficient context: A new lens on retrieval augmented generation systems (2024), https://arxiv.org/abs/2411.06037

[15] Labadze, L., Grigolia, M., Machaidze, L.: Role of ai chatbots in education: systematic literature review. s41239-023-00426-1.pdf (2023), https://doi.org/10.1186/s41239-023-00426-1

[16] Li, S., Stenzel, L., Eickhoff, C., Bahrainian, S.A.: Enhancing retrieval-augmented generation: A study of best practices (2025), https://arxiv.org/abs/2501.07391

[17] Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P.: Lost in the middle: How language models use long contexts (2023), https://arxiv.org/abs/2307.03172

[18] M, L., Y, M., D, L., T, S., K, E., B, L.: Roles, users, benefits, and limitations of chatbots in health care: Rapid review. Journal of Medical Internet Research 26, pp. e56930 (2024), https://www.jmir.org/2024/1/e56930

[19] Microsoft: Phi cookbook. https://github.com/microsoft/PhiCookBook (2024), accessed: 2025-03-28

[20] Ociepa, K., Łukasz Flis, Wróbel, K., Gwoździej, A., Kinas, R.: Bielik 7b v0.1: A polish language model – development, insights, and evaluation (2024), https://arxiv.org/abs/2410.18565

[21] Perez, A., Vizcaino, X.: Advanced ingestion process powered by llm parsing for rag system (2024), https://arxiv.org/abs/2412.15262

[22] Qu, R., Tu, R., Bao, F.: Is semantic chunking worth the computational cost? (2024), https://arxiv.org/abs/2410.13070

[23] Sardana, A.: Real-time evaluation models for rag: Who detects hallucinations best? (2025), https://arxiv.org/abs/2503.21157

[24] Singh, I.S., Aggarwal, R., Allahverdiyev, I., Taha, M., Akalin, A., Zhu, K., O'Brien, S.: Chunkrag: Novel llm-chunk filtering method for rag systems (2025), https://arxiv.org/abs/2410.19572

[25] Tran, K.T., Dao, D., Nguyen, M.D., Pham, Q.V., O'Sullivan, B., Nguyen, H.D.: Multi-agent collaboration mechanisms: A survey of llms (2025), https://arxiv.org/abs/2501.06322

[26] Wang, X., Wang, Z., Gao, X., Zhang, F., Wu, Y., Xu, Z., Shi, T., Wang, Z., Li, S., Qian, Q., Yin, R., Lv, C., Zheng, X., Huang, X.: Searching for best practices in retrieval-augmented generation. In: Al-Onaizan, Y., Bansal, M., Chen, Y.N. (eds.) Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. pp. 17716–17736. Association for Computational Linguistics, Miami, Florida, USA (Nov 2024), https://aclanthology.org/2024.emnlp-main.981/

[27] Yang, Y., Li, Z., Dong, Q., Xia, H., Sui, Z.: Can large multimodal models uncover deep semantics behind images? (2024), https://arxiv.org/abs/2402.11281