engineering proceedings

MDPI

Proceeding Paper

Comparative Evaluation of Baseline and Retrieval-Augmented Generative AI Chatbot Architectures for Customer Support Use Cases †

Gergő László Novák * and András Aschenbrenner

Department of Artificial Intelligence and HPC, Faculty of Informatics and Electrical Engineering, Széchenyi István University, 9026 Győr, Hungary; aschenbrenner.andras@sze.hu
* Correspondence: novak.gergo.laszlo@sze.hu
† Presented at the Sustainable Mobility and Transportation Symposium 2026, Győr, Hungary, 8 October 2026.

Abstract

Generative artificial intelligence-based chatbot systems are increasingly used to support information provision and customer service processes; however, their reliability strongly depends on the underlying architecture and the availability of domain-specific knowledge. This study compares a zero-shot baseline chatbot relying solely on parametric model knowledge with a Retrieval-Augmented Generation architecture connected to an external document-based knowledge base. The experimental system was implemented using the n8n automation platform, GPT-4o-mini, OpenAI embeddings, and a ChromaDB vector database. The architectures were evaluated through information retrieval metrics, including Recall@k, Precision@k, and Mean Reciprocal Rank, as well as automated response quality assessment focusing on relevance, contextual accuracy, and hallucination. The results indicate that RAG improves factual grounding and reduces hallucination in document-based information systems, while baseline models may provide more fluent responses. The findings support architecture selection for reliable generative chatbot deployment.

Keywords: generative artificial intelligence; chatbot architecture; retrieval-augmented generation; customer support; information retrieval; hallucination; vector database

1. Introduction

Generative artificial intelligence has become a practical component of modern information provision and customer support systems [1]. Large language models can generate coherent and context-sensitive responses. They are also efficient in adapting to a user's phrasing and reducing the cognitive effort needed for searching through documents. This ability is especially valuable in support use cases, where users expect immediate answers to questions about policies, processes and technical or administrative information. However, this generative flexibility can also raise serious reliability issues. A model may produce plausible statements that are not based on a genuine source [1–3].

The architectural design of a chatbot directly affects its factual reliability. A simple baseline model generates its answers based on the model's parametric memory encoded in its weights. This approach is simply implementable and can return fluent responses with low latency, but its knowledge is static and difficult to verify. In contrast, Retrieval-Augmented Generation (RAG) combines the abilities of an LLM with an external, non-parametric knowledge source [4,5]. In a RAG system, relevant document segments are retrieved from an indexed corpus, which is then forwarded to the model as context before generation. This

Check for updates

Academic Editors: Gábor Dogossy, Szabolcs Kocsis-Szürke, Dániel Csikor, Dániel Serfőző, Miklós Kuczmann, András Lajos Nagy, László Lendvai and Ambrus Zelei

Published: 5 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

makes the response more strongly dependent on the available evidence and can reduce hallucination in knowledge-intensive tasks [2–4].

This distinction is especially important in document-based customer support scenarios. In such environments, relevant knowledge may frequently be narrow, field-specific and prone to changes. General LLM knowledge might be insufficient, deprecated or inconsistent with the authoritative documentation. Furthermore, the complete retraining of the model after every slight change in information is costly and might even be unjustified. This is especially true if the primary task is efficient information retrieval rather than stylistic adaptation [3,4]. RAG offers a compromise by allowing modification of the knowledge base without needing to retrain the model.

Existing studies and comparisons often evaluate RAG in a wide, knowledge-heavy environment or focus on such component-level developments as information retrieval, reranking or evaluation metrics [3]. Less emphasis is put on compact support scenarios where the knowledge base corpus might be smaller, but factual accuracy is critical. In these cases, the central problem is not general searching but a compromise between a fluent LLM and a model constrained in its retrieval capabilities, which might answer more cautiously.

This study presents the comparative evaluation of two generative AI chatbot architectures. The first one is a zero-shot baseline architecture. The second architecture is RAG connected to an external document-based knowledge base. The aim of the study is to examine how the integration of a retrieval component influences information retrieval performance, response relevance, contextual accuracy and hallucination behavior. The paper contributes a compact experimental framework for comparing the mentioned workflows under identical model and question conditions. The emphasis during the comparison was put on measurable trade-offs between simplicity, fluency, factual grounding and scalability. This study differs from broader benchmark studies as it focuses on a small document-based knowledge base, while interpreting information retrieval metrics alongside response-level behavior. The most important finding is that efficient retrieval reduces hallucination. Despite that, automated response quality evaluators favor the fluency of standard LLMs.

2. Materials and Methods

The experiment was designed as a controlled comparison between the two architectures. The same generator model and validation set were used. The first architecture was a zero-shot baseline LLM that generated answers without external context. The second architecture was a RAG system that retrieved relevant chunks of the external knowledge base using them as context for response generation. Both workflows were implemented on the n8n automation platform for seamless repeatable execution and data export. It also made integration with external services effortless, which proved useful for vector database operations.

2.1. The Corpus

The source of the specialized knowledge base was a concise document corpus. The original material consisted of 80 pages of unstructured text and mixed informational content. The corpus was intentionally treated as a narrow knowledge base. With this approach, the experiment could focus on the architecture and give a realistic picture of small corpus support systems. Providing a statistically representative benchmark across all domains was not intended.

2.2. Validation and Dataset

For the evaluation of both information retrieval and response generation, a validation dataset was created from 40 manually selected questions. The questions were of four distinct

https://doi.org/10.3390/engproc2026157049

categories: factual, procedural, complex and non-answerable. With this structure, short factual questions, explanation of multi-step procedures, and synthesis of information from multiple chunks were covered. It also made testing answer refusal possible when no context can be provided from the external knowledge base. Table 1 summarizes the composition of the validation dataset.

Table 1. The composition of the validation set.

Question Type	Number of Questions	Evaluation Purpose
Factual	16	Specific facts, names, values, locations, and dates
Procedural	12	Multi-step instructions and administrative processes
Complex	8	Synthesis of information from multiple document segments
Non-answerable	4	Detection of missing evidence and controlled refusal

2.3. Corpus Preprocessing and Chunking

During preprocessing, the base text was segmented into chunks of 250–500 words utilizing a recursive, character-based strategy. To maintain semantic continuity throughout chunk boundaries, an overlap of 10–15% was used. Every chunk was stored alongside a unique chunk ID, a chunk title, and a section-level index. Storing the metadata was vital to enable the generated answer and the retrieved information to be evaluated separately.

The chosen chunk size was a compromise between retaining context and retrieval precision. Smaller chunks might improve specificity but can separate an answer from its context. Larger chunks help retain context but can cause noise and unnecessary tokens. The above-mentioned overlap reduced boundary loss without inducing excessive duplication. The value of top-k was fixed at three to maintain comparability across questions.

2.4. Baseline Architecture

Baseline architecture meant the simplest generative configuration. The validation questions were loaded from a CSV file on Google Drive, which was converted to JSON objects through n8n. Then, each question was forwarded to a basic LLM chain connected to a GPT-4o-mini model. No external context, document retrieval or vector database query was used. The output was merged with the metadata of the original question, then exported in a CSV file for later evaluation.

This architecture was chosen to represent a low-complexity support chatbot that relies solely on its internal parametric memory. Its expected architectural advantages are simple deployment, lower operational cost, shorter execution path, and fluent language generation. Runtime latency and operational cost were not measured in this study; thus, the mentioned advantages are expectations concluded from the architecture's simpler workflow rather than empirical results. Its main limitation is the absence of grounding in the authoritative document corpus, which may lead to misinterpretation or hallucination when the requested information is domain-specific or not present in the model's training data.

2.5. Retrieval-Augmented Generation Architecture

The RAG architecture extended the baseline workflow with an external non-parametric memory. The pre-processed chunks were imported from a JSONL file and stored in ChromaDB. OpenAI's text-embedding-3-small model was used to embed both the document

chunks and the incoming user questions in the same vector space. For each validation question the top three most relevant chunks were retrieved. Those were then forwarded to gpt-4o-mini alongside the original questions.

A parallel metadata extraction branch had to be made, as the default question answer chain did not make retrieving source identifiers possible. The branch queried the vector storage and collected the chunk identifiers used for evaluation. Then, the generated answer, the original question's metadata and the retrieved chunk identifiers were merged and exported. That approach made separately evaluating the retriever component possible.

Figure 1 presents the experimental workflow. The baseline path sends the question directly to the generator, whereas the RAG path first performs embedding-based retrieval and uses the selected document chunks as grounding context. The same generator model was used in both paths to isolate the effect of the retrieval component.

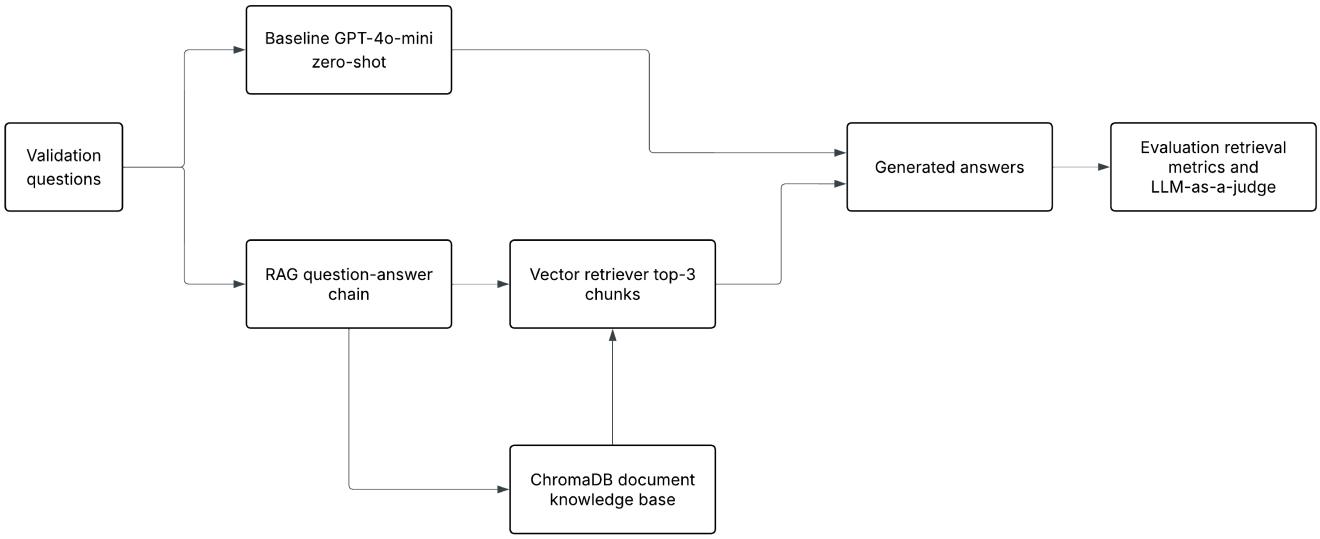


Figure 1. Experimental workflow used to compare the baseline and RAG chatbot architectures.

2.6. Evaluation Metrics

During evaluation, both the retrieval process and the generation quality were taken into consideration. Retrieval quality was measured by Recall@k, Precision@k and Mean Reciprocal Rank (MRR). Recall@k measures whether relevant chunks appear among the first k retrieved results. Precision@k measures the proportion of retrieved chunks that are relevant. MRR focuses on the rank of the first relevant result. It is especially useful if the response depends on whether the required evidence appears early in the context window [6]. A manual relevance mapping was created, on which the evaluation was based. A chunk was marked relevant if it contained a direct answer for a factual question, contained a crucial step for a procedural question or contained a useful piece of information for answering a complex question. The questions were made for the document corpus; thus, in the case of non-answerable questions, no chunk was marked as relevant.

The generated answers were assessed by an automated "LLM-as-a-judge" procedure, using the gpt-4o-mini model, which was also used for generation. Model temperature was set to 0 for smaller sampling variability. The evaluator prompt contains the original question, the baseline answer, the RAG answer and evaluation guidelines that explain the three dimensions being assessed: relevance, contextual accuracy and hallucination [7]. In each dimension, a value of 1 indicated an advantage for RAG, a −1 an advantage for baseline and 0 indicated a tie. Exported values are processed with the help of a Python 3.14 script to calculate aggregate scores. This provides a scalable annotation. Alongside the

aggregate scores, the decisions were grouped by question category to provide a breakdown of the scores by category using the same method.

The evaluation method also accounted for biases. Automated evaluators might favor fluent, detailed answers, even if those answers lack factual grounding. This so-called verbosity bias can penalize the concise answers or the rejection of answers in the RAG architecture, even though these are beneficial in customer support scenarios [8,9].

3. Results

The retrieval component of the RAG architecture performed strongly on the validation set. Table 2 presents the evaluation results for the top three retrieved chunks. The Recall@3 value of 0.9468 indicates that the system was able to retrieve the required context in almost all answerable cases. The MRR value of 0.9306 indicates that most of the time, the first relevant segment was ranked in either the first or second position. The lower Precision@3 value is attributable to the metric's mathematical characteristics. The retriever component had to return with exactly three chunks, even when there was only one relevant chunk.

Table 2. Retrieval evaluation results for RAG architecture.

Metric	Value	Interpretation
Recall@3	0.9468	The relevant context was retrieved in nearly all cases
Precision@3	0.3981	Lower value caused by fixed retrieval of three chunks
Mean Reciprocal Rank	0.9306	The first relevant chunk was usually highly ranked

The results of the retrieval component evaluation of the RAG pipeline demonstrate its ability to select relevant evidence, particularly in factual and procedural scenarios. The underlying behavior seems to differ among different question types. In case of factual questions, the retriever usually identified only one highly relevant chunk, containing the required value, name, place, or date. This behavior explains the high MRR value, as the chunk containing the relevant information was ranked first or second place most of the time. In cases of procedural questions, multiple chunks could contain useful information, as the answer frequently required the synthesis of information from multiple chunks. In these scenarios, the retrieved context sometimes contained adjacent or supporting fragments, leading to a lower Precision@3 value, but helped procedural continuity.

Information retrieval for complex questions became more difficult, as generating a precise answer required the synthesis of information from multiple chunks. The fixed top 3 setting was a compromise. Although the probability of the retrieval was increased, it introduced a secondary, more general context. Thus, the lower Precision@3 value should not be seen as a direct retrieval fault. It primarily reflects the architectural design decision that three chunks must be retrieved, even if only one question would be sufficient.

An important operational difference is that RAG architecture makes the selection of relevant information unambiguous and measurable. The baseline model in the absence of the retrieval step is not capable of separating corpus-based information from general parametric information. In contrast, RAG enables tracing errors to retrieval faults, the document's lack of coverage or generational faults. This makes the system more transparent and easier to debug; however, it increases its complexity at the same time.

The automated response quality comparison produced more mixed results. As shown in Table 3, the baseline model received a slight advantage in relevance and a larger advantage in contextual accuracy according to the automated judge. However, the RAG model showed a clear advantage in the hallucination dimension. This indicates that the baseline model

generates more fluent and verbose responses, while RAG is more conservative and grounds its answers in the document corpus.

Table 3. Automated comparison of generated responses. Positive values indicate RAG advantage; negative values indicate baseline advantage.

Evaluation Dimension	Aggregate Value	Observed Tendency
Relevance	−0.1	Slight baseline advantage
Contextual accuracy	−0.5	Baseline favored by the automated judge
Hallucination	+0.5	RAG advantage due to stronger grounding

The per-category breakdown of the scores, shown in Table 4, provides a more nuanced image of the aggregate results. In factual and procedural questions, direct and stepwise information provision capability was tested. Answering complex questions required the synthesis of information from multiple chunks. The non-answerable category was introduced to test answer refusal behavior. Considering the size of the last category, the values should be interpreted qualitatively rather than quantitatively.

Table 4. Automated comparison of generated responses per question category. Positive values indicate RAG advantage; negative values indicate baseline advantage.

Metric	Factual	Procedural	Complex	Non-Answerable
Relevance	2	−4	2	−4
Contextual accuracy	−6	−10	0	−4
Hallucination	6	10	0	4

The manual review of the output provided an important explanation for these values. The baseline model generally provided longer, more verbose answers with broader contextual framing. In cases of non-answerable or domain-specific questions, this behavior increased the risk of hallucination, as the model tried to provide an answer without any factual grounding from the document corpus.

In contrast, the RAG model showed the opposite tendency. With clear, retrieved evidence, the system provided answers that were grounded in the knowledge base. When evidence was incomplete, it usually gave short answers or denied answering at all. Although this behavior was scored negatively by the automated judge, it is the desirable behavior in support systems where factual precision is more important than stylistic completeness.

Thus, the baseline optimized for fluency and completeness, while RAG optimized for evidence-bound reliability.

4. Discussion

The results confirm that the decision on chatbot architecture has to be aligned with reliability requirements. The baseline LLM architecture is operationally simple and can generate coherent answers without additional infrastructure. It may be sufficient when the task is providing general information, informal help and language transformation. However, in document-based support scenarios, the verifiability of the baseline is limited. It is unable to distinguish valid information from the document corpus from general knowledge derived from its own parametric memory.

The RAG architecture mitigates this limit by introducing a retrieval step. The high Recall@3 and MRR values suggest that the vector-based retrieval procedure could identify relevant chunks efficiently. This supports selecting RAG for compact domain-specific corpora where factual grounding is crucial. The lower Precision@3 value does not mean poor retrieval quality. The architectural design choice of the top three forces the retrieval of three chunks even if one would be sufficient. In practical deployments, this could be improved by using a dynamic top-k approach, hybrid search, reranking or confidence-based context filtering [10–12].

The generation results highlight a known evaluation challenge. An automated evaluator may favor fluent and verbose responses, even if they are not factually grounded [8,9]. This introduces a risk when comparing RAG and baseline models, as a well-designed RAG answers more cautiously and deliberately. In particular, in support scenarios, a clear "I don't know" answer can be more valuable than a fluent but not factually grounded response. Therefore, reducing hallucination and the quality of answer refusal should be treated as a primary requirement.

From a system design point of view, RAG means more complexity. It includes document chunking, embedding, vector storage, retrieval configuration, managing metadata and tracing retrieval errors. These steps increase development effort and can cause operational latency. Nonetheless, this architecture provides better scalability, as its knowledge base is updatable without retraining the whole model. Fine-tuning remains useful if the goal is to train a model for a certain style or terminology, but less flexible when it comes to a regularly updated knowledge base [3,4].

The experiment also suggests that information retrieval and response quality should not be evaluated in isolation. If a relevant chunk is missing from the context, a grounded answer cannot be expected. In contrast, even correct information retrieval cannot guarantee a valid final answer if the generator cannot process the information properly. Future evaluation methodologies should integrate information retrieval and factuality metrics, as well as metrics assessing refusal behavior [11,12].

5. Limitations and Future Research

There are limitations to the research. First, the validation set comprised 40 questions and was designed for a compact, document-based knowledge base. This enabled manual assessment of the relevance metrics yet limited statistical generalization. A larger evaluation set would be needed to verify whether the same patterns hold across several customer support domains, larger corpora, and a more diverse set of questions.

Second, the assessment of response quality was based on a so-called "LLM-as-a-judge" method. Although this method is scalable, it is not neutral: the results suggest bias towards verbose and more fluent responses [8,9]. Future research should augment automated assessment with human annotation for factuality, appropriateness of response refusal and source alignment.

Third, the system was tested in a single domain and with one generator model, embedding model, and vector database configuration. Future research should examine hybrid sparse–dense retrieval, dynamic top-k selection, reranking, confidence thresholds, uncertainty estimation, multilingual queries, and rejection-based fine-tuning.

Therefore, the results should be interpreted as configuration-dependent. With different corpus structures, more heterogeneous or simply larger corpora, alternative generator backbones, different embedding models, reranking methods or query engines may change both retrieval quality and response-level behavior. Consequently, the findings are better off treated as evidence of the trade-offs observed under a specific, small corpus domain, rather than a universal ranking of zero-shot and RAG architectures.

https://doi.org/10.3390/engproc2026157049

6. Conclusions

This study compared a zero-shot baseline generative chatbot architecture with RAG architecture for document-based customer support use cases. The RAG system achieved strong retrieval performance, with Recall@3 of 0.9468 and MRR of 0.9306, indicating that the relevant document context was usually retrieved and highly ranked. The automated response evaluator favored the baseline model regarding relevance and contextual accuracy. However, manual inspection and interpretation highlighted that this was partly caused by verbosity bias and the baseline model's tendency to provide confident answers, even when it lacks grounding.

The main practical implication is that RAG is the preferred architecture when factuality is the most crucial factor. Zero-shot models still prove useful for general-purpose scenarios and for fluency, although they carry a higher risk of hallucination.

All things considered, RAG architecture modifies system operation by shifting from unconstrained generation to factually grounded information synthesis. In some cases, this attribute might reduce stylistic richness but increase reliability in strict, fact-bound use cases. Future research should extend evaluation with larger datasets, human annotation, hybrid retrieval, reranking, uncertainty estimation and multilingual testing.

Author Contributions: Conceptualization, G.L.N. and A.A.; methodology, G.L.N.; software, G.L.N.; validation, G.L.N.; formal analysis, G.L.N.; investigation, G.L.N.; data curation, G.L.N.; writing—original draft preparation, G.L.N.; writing—review and editing, G.L.N. and A.A.; visualization, G.L.N.; supervision, A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was founded by the European Union under the Horizon Europe Marie Skłodowska-Curie Actions Staff Exchanges program through the project Overcoming Digital Divide in Europe and Southeast Asia (ODDEA) [Grant Agreement No. 101086381]. Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or the European Research Executive Agency (REA). Neither the European Union nor the granting authority can be held responsible for them.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available from the corresponding author upon reasonable request.

Acknowledgments: During the preparation of this manuscript, the authors used ChatGPT (GPT-5.5 Pro model, OpenAI (San Francisco, CA, USA), 2026) and Grammarly solely for language editing, grammatical corrections, and improving the clarity and readability of the English text. All suggested revisions were reviewed by the authors, who take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Caldarini, G.; Jaf, S.; McGarry, K. A Literature Survey of Recent Advances in Chatbots. *Information* **2022**, *13*, 41. [CrossRef]
2. Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y.J.; Chen, D.; Dai, W.; et al. Survey of Hallucination in Natural Language Generation. *ACM Comput. Surv.* **2023**, *55*, 248. [CrossRef]
3. Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, H. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv* **2024**, arXiv:2312.10997.
4. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.; Rocktäschel, T.; et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 9459–9474.
5. Barnett, S.; Kurniawan, S.; Thudumu, S.; Brannelly, Z.; Abdelrazek, M. Seven Failure Points When Engineering a Retrieval Augmented Generation System. In Proceedings of the 2024 IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI (CAIN 2024), Lisbon, Portugal, 14–15 April 2024; pp. 194–199. [CrossRef]

6. Manning, C.D.; Raghavan, P.; Schütze, H. *Introduction to Information Retrieval*; Cambridge University Press: Cambridge, UK, 2008.
7. Es, S.; James, J.; Espinosa-Anke, L.; Schockaert, S. RAGAS: Automated Evaluation of Retrieval Augmented Generation. *arXiv* **2023**, arXiv:2309.15217.
8. Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E.P.; et al. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In Proceedings of the 37th International Conference on Neural Information Processing Systems, New Orleans, LA, USA, 10–16 December 2023; pp. 46595–46623.
9. Stureborg, R.; Alikaniotis, D.; Suhara, Y. Large Language Models are Inconsistent and Biased Evaluators. *arXiv* **2024**, arXiv:2405.01724.
10. Zhang, H.; Chen, X.; Li, Y.; Wang, Z. Efficient and Effective Retrieval of Dense-Sparse Hybrid Vectors Using Graph-Based Approximate Nearest Neighbor Search. *arXiv* **2024**, arXiv:2410.20381.
11. Robertson, S.; Zaragoza, H. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* **2009**, *4*, 1–174. [CrossRef]
12. Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; Yih, W.-t. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 6769–6781. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.