Retrieval Augmented Generation (RAG) for University Life

Vlad-Ioan Bîrsan

Computer Science Department, University of Bucharest, Romania

Abstract

This report presents the development and evaluation of a specialized Retrieval-Augmented Generation (RAG) system designed to work as a research assistant for the domain of Anomaly Detection, specifically focusing on "Isolation Forest" algorithms and their variants. We propose a modular architecture that integrates preprocessing pipelines, including Gemini-powered parsing, semantic versus window-based chunking strategies, and vector retrieval via LanceDB. To evaluate the performance of different configurations, we conduct an analysis of dense embedding models, specifically distinguishing between local HuggingFace models (BGE-M3) and cloud-based solutions (Gemini Embedding). Furthermore, the proposed system is compared against two distinct baselines: a classical Information Retrieval system utilizing BM25 with lemmatization, and a 4-bit quantized Mistral-7B Large Language Model. Our experiments demonstrate that the RAG pipeline, particularly when augmented with chunks reranking and strict acceptance criteria, significantly mitigates hallucinations compared to the standalone LLM and overcomes the semantic limitations of keyword-based retrieval. The results prove the necessity of retrieving external knowledge for answering domain-specific queries while maintaining robustness against out-of-scope inputs.

1 Introduction

Large Language Models (LLMs) have changed the way we interact with information. However, they face limitations when dealing with very specialized domains. General-purpose models often hallucinate facts or lack access to the most recent information, making them unreliable for queries that require deep knowledge in a specific area. This project addresses this gap by developing a Retrieval-Augmented Generation (RAG) system designed for the field of Anomaly Detection.

The aim of this project is to create an AI "Research Assistant" capable of answering complex questions about "Isolation Forest" algorithms and their variants. By bounding the LLM's responses to a corpus of scientific PDFs, the goal is to combine the reasoning capabilities of generative AI with the factual accuracy of a search engine. We evaluate this system against two baselines: a classical keyword search (BM25) and a standalone LLM (Mistral-7B), to observe the advantages of the RAG architecture in handling domain-specific data.

The remainder of this report is organized as follows: Section 2 reviews related work in the field of RAG. Section 3 details our proposed Method, covering the dataset characteristics, the preprocessing pipeline, and the modular design of the RAG system. Section 4 describes the two alternative approaches used as baselines (the classical BM25 retrieval system and the standalone Mistral-7B model). Finally, Section 5 outlines potential future work, and Section 6 presents the conclusion.

2 Related Work

To adapt LLMs for specific domains, two approaches are generally used: fine-tuning and Retrieval-Augmented Generation (RAG).

Fine-tuning involves updating the model's internal parameters to learn new information. While useful for learning specific tasks or styles, this method is computationally expensive and the knowledge becomes static after training. As noted by Gao et al. in their survey on RAG [3], LLMs relying only on internal weights often suffer from hallucinations and cannot easily access up-to-date information. In contrast, RAG keeps the model's weights frozen and instead retrieves relevant information from an external source during inference. This non-parametric approach allows for cheaper updates, higher transparency, and verifiable citations.

In the context of scientific research, accuracy is very important. Lála et al. emphasize on this idea with "PaperQA", an RAG system designed to answer questions from scientific literature [6]. They showed that retrieving context from full-text papers significantly reduces hallucinations compared to standard LLMs. Our project builds upon these findings by applying similar RAG techniques specifically to the domain of Anomaly Detection, utilizing a corpus of "Isolation Forest" research papers to ensure that the model answers technical questions with high reliability.

3 Method

Our system is built upon a flexible modular architecture designed to facilitate experimentation. The framework is divided into two loosely coupled components: the preprocessing pipeline and the query pipeline.

The preprocessing pipeline is responsible for the knowledge base construction. It ingests the raw corpus of PDF documents, transforming them into a structured vector index. This component is designed to be independent and is executed only when the source data changes (e.g., adding new papers).

The query pipeline also operates independently, accessing the pre-computed knowledge base to process user inquiries in real-time. Both components are further split into specialized, interchangeable sub-modules (such as parsers, chunkers, and rerankers). This granular modularity allows us to isolate variables—such as the embedding model or chunking strategy— facilitating the evaluation of their impact on the system's performance, as detailed in the following subsections.

3.1 Dataset

The knowledge base for this system is built from a corpus of nine scientific papers, selected to cover the variations of the "Isolation Forest" algorithm. This corpus contains mathematical formulations, algorithmic pseudocode, domain-specific terminology, image plots and diagrams .

The corpus includes the work on Extended Isolation Forest [4], as well as variations such as Functional Isolation Forest [9], Generalized Isolation Forest [7], and Kernel Isolation Forest [8]. It also covers approaches like K-Means-based Isolation Forest [5], alongside probabilistic variants [10], [1] and [2].

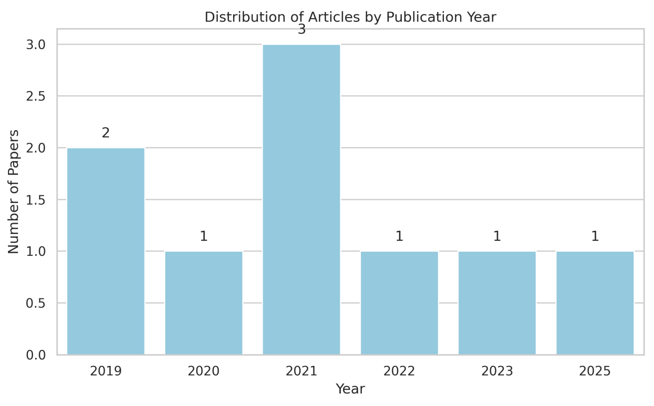

Figure 1: Publishing year distribution of the corpus. The dataset aims to include more recent work, in order to ensure that the RAG system has access to information likely absent from the training data of older LLMs.

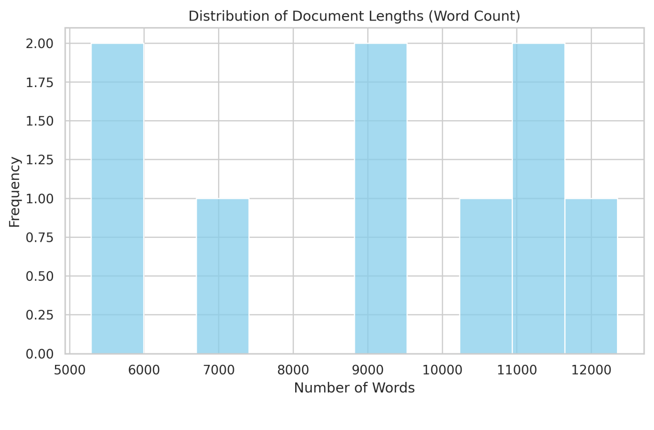

Figure 2: Distribution of document lengths by token count.

Figure 3: Concept word cloud generated from the corpus text.

As illustrated in Figure 1, the dataset is concentrated between 2019 and 2025, providing a fresher knowledge source that aims to don't overlap with the static training data of LLMs. Moreover, the diversity in document length (Figure 2), is reflected in the varying distributions of chunk counts (Figure 4) and sizes (Figure 5).

3.2 Preprocessing

The preprocessing pipeline is responsible for transforming raw PDF documents into a searchable knowledge base. This process is divided into three stages: parsing, chunking, and vector storage, as illustrated in Figure 6.

Scientific papers often present complex layouts (e.g., two-column formats, floating figures) that might confuse standard deterministic PDF libraries. To address this, we utilized "Gemini 3 Pro" accessed via LlamaParse. The model was prompted to transform the documents into clean Markdown format, explicitly removing page headers and footers while replacing images with descriptive captions of their content.

We selected Gemini after a comparative evaluation with Claude 3.5 Sonnet. Claude struggled with document layout analysis, often misordering text from multi-column pages, failing to strip headers/footers, and omitting entire subsections. In contrast, Gemini proved better results in preserving reading order and handling non-standard elements (e.g., text embedded within images, tables), providing a cleaner source for chunking. The specific prompt used for this task is detailed in Appendix A.1.1.

To prepare the text for retrieval, we used two distinct segmentation strategies. To prevent context contamination, each document was processed independently.

1. **Semantic Chunking:** This method segments text based on changes in meaning rather than fixed character counts. The document is first split into sentences. An embedding model then computes the semantic similarity between a buffer of sentences (we used size=1) and the subsequent sentence. A split occurs when this similarity score drops below a threshold determined by a percentile breakpoint. We experimented with breakpoint percentiles of $p \in \{80, 95\}$. A higher percentile results in fewer, larger chunks, while a lower percentile creates more granular segments. We evaluated this strategy using two embed-

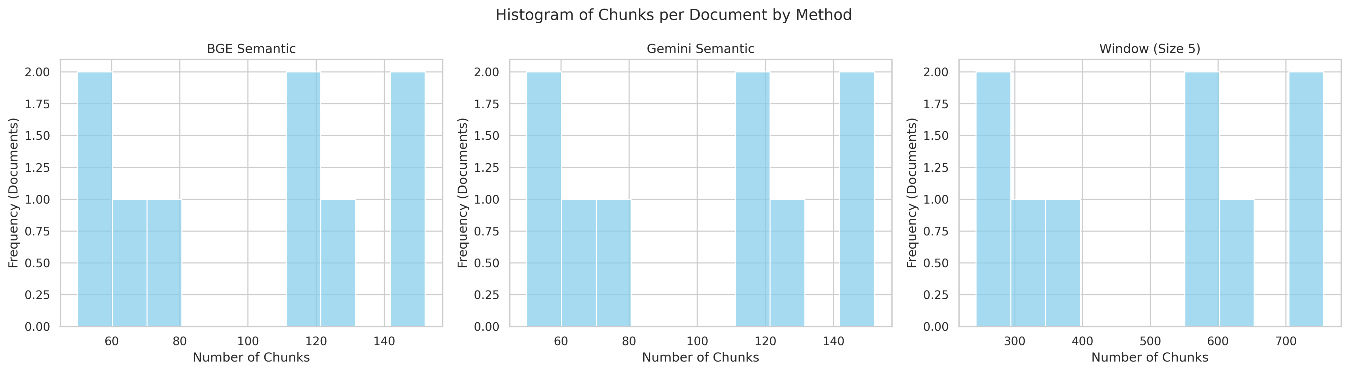

Figure 4: Histogram of chunks generated per document across the three chunking strategies.

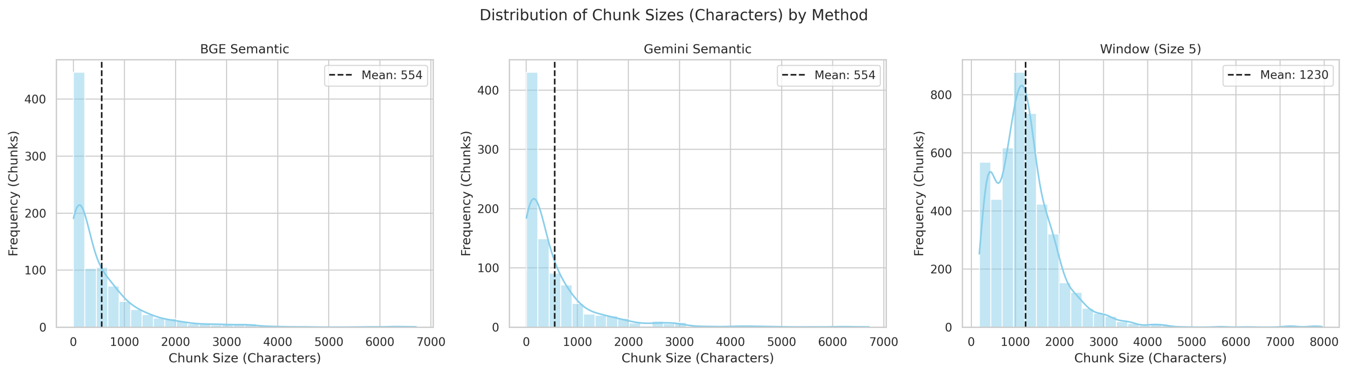

Figure 5: Distribution of chunk sizes (in characters).

ding models: the local BGE-M3 and the API-based Gemini Embedding 001.

2. **Window Chunking:** For comparison, we implemented a sliding window approach. For every sentence in the document, a chunk is created by centering that sentence and including a context window of k sentences to the left and right. We found that a window size of $k = 5$ preserved the context better than $k = 3$. While straightforward, this method introduces significant redundancy and higher storage overhead compared to the non-overlapping semantic chunking.

The generated chunks were indexed using LanceDB, a serverless vector database. We maintained six separate vector stores corresponding to the embedding models used (BGE-M3 and Gemini 001) and to the chunking approaches taken (semantic chunking using Gemini API, semantic segmentation using BGE-M3 model, or window based splitting), this way ensuring compatibility during retrieval. All vectors were normalized for cosine similarity search and each chunk was enriched with metadata—including title, authors, publication year, and file path—to enable the generation of citations in the final RAG response.

3.3 Method and Results

The query processing pipeline is designed as a sequence of modular components that can be easily enabled or disabled to test different configurations. This architecture allows us to isolate the impact of different techniques such as query expansion and chunks re-ranking. Figure 7 provides a visual overview of this workflow.

Upon ingestion, a user's query can optionally pass through rephrasing module. This component utilizes Gemini 3 Pro (via API) to rewrite the input question. The goal is to standardize the writing style and expand on queries that may be too brief or ambiguous, thereby increasing the possibility of a successful semantic match in the vector database. The specific prompt used for rephrasing is detailed in Appendix A.2.1.

The processed query is embedded using the model corresponding to the active experiment (BGE-M3 or Gemini Embedding 001). We perform a cosine similarity search against the vector store to retrieve the top $k = 20$ chunks.

Following retrieval, an optional reranking module can be utilised. This module employs the Gemini LLM to assess the relevance of the retrieved chunks in relation to the specific question. Unlike simple cosine similarity, which only measures vector proximity, the LLM-based reranker understands context. It reorders the chunks by importance and discards those appearing irrelevant. The prompt used for achieving this behavior is listed in Appendix A.2.2.

To prevent hallucinations and ensure the system remains within its domain, we implemented an acceptance criteria module. Before generating a final answer, the system sends the retrieved chunks and the query to the LLM with instructions to classify the request into one of three categories:

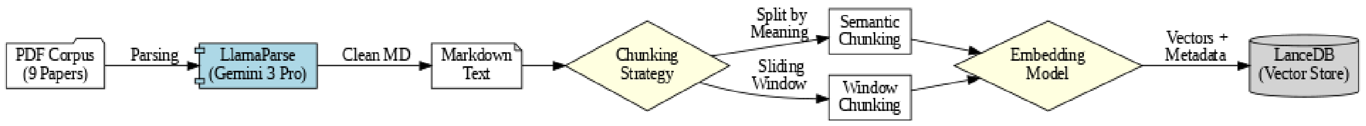

Figure 6: Overview of the preprocessing pipeline: (1) LLM-based parsing transforms PDFs to Markdown, (2) Strategy-based chunking segments the text, and (3) Vectors are stored in LanceDB with metadata.

- ANSWERABLE: The context contains sufficient information.
- NO_DATA: The question is relevant to Anomaly Detection, but the specific answer is missing from our corpus.
- UNRELATED: The question falls outside the scope of the anomaly detection literature (e.g., general knowledge).

This LLM-based classification proved more robust than using arbitrary similarity score thresholds, which are often difficult to tune and require constant updates as the knowledge base grows. If the label is NO_DATA or UNRELATED, the system outputs a reason and terminates the workflow. The prompt for this guardrail is available in Appendix A.2.3.

If the query is considered ANSWERABLE, the top-ranked chunks—along with their metadata (titles, authors, years)—are fed into the final generation module. The Gemini 3 Pro model constructs the answer, citing the specific papers used as context. The system prompt for generation is found in Appendix A.2.4.

To evaluate the system, we created a set of 20 questions (see Appendix B): 14 answerable based on the corpus, 4 relevant but with no data, and 2 completely unrelated.

We evaluated across 24 distinct configurations. This included permutations of the three preprocessing strategies (Semantic BGE, Semantic Gemini, Window) combined with four query pipeline states:

1. Rephrasing + Reranking
2. No Rephrasing + Reranking
3. Rephrasing + No Reranking
4. Baseline (No Rephrasing, No Reranking)

All code, assets, and the full generated answers for every configuration are available in our GitHub repository. Notably, across all 24 experimental setups, the guardrail mechanism achieved 100% accuracy in correctly labeling the queries as ANSWERABLE, NO_DATA, or UNRELATED, proving the reliability of LLM-based acceptance criteria over threshold-based methods.

4 Alternative Approaches

To emphasize the value of the RAG architecture, we also compared our system against two distinct baselines: a standalone Large Language Model (Mistral 7B) representing pure parametric knowledge, and a classical BM25 search engine representing lexical retrieval.

4.1 Standalone LLM (Mistral 7B)

As a baseline for generative capability, we evaluated the Mistral-7B-Instruct-v0.3 model without access to external documents. To make the execution feasible on our hardware, the model was loaded with 4-bit quantization using the `BitsAndBytes` library.

We used a system prompt (Appendix A.3.1) requiring the model to classify queries into ANSWERABLE, NO_DATA, or UNRELATED, mirroring this way the guardrails of our RAG pipeline and allowing for a comparison of reasoning capabilities. However, the standalone model presented poorer performance compared to the RAG system:

- Hallucinations: for questions regarding specific algorithmic details, the model frequently hallucinated incorrect mechanisms. This confirms that the model's knowledge stored in its weights during training did not contain the specific contributions of our 2019-2025 corpus.
- Classification failure: the model failed to distinguish between NO_DATA and UNRELATED scenarios. Without a reference text to check against, it often defaulted to attempting an answer based on general knowledge, failing to admit that it doesn't have knowledge about the topic when required.

4.2 Classical Retrieval (BM25)

As a baseline for retrieval accuracy, we implemented a classical Best Matching 25 (BM25) system using the `rank_bm25` library. Unlike vector-based semantic search, BM25 relies on probabilistic keyword matching.

To ensure a fair comparison, we implemented a text preprocessing pipeline applied identically to both the document corpus and the user queries:

1. Tokenization: breaking text into "atoms".
2. Stop word removal: Removing common English words (e.g., "the", "is", "for") using the NLTK library. this step proved crucial; without it, queries like "What is the best recipe for pizza?" returned false positives primarily based on matching the word "the" or "is".
3. Lemmatization: reducing words to their root forms to handle grammatical variations.

The system was configured to return the top $k = 3$ relevant chunks along with their metadata and a confidence score. While BM25 lacks the semantic understanding to assign explicit labels, the confidence scores provided a strong indicator for classification:

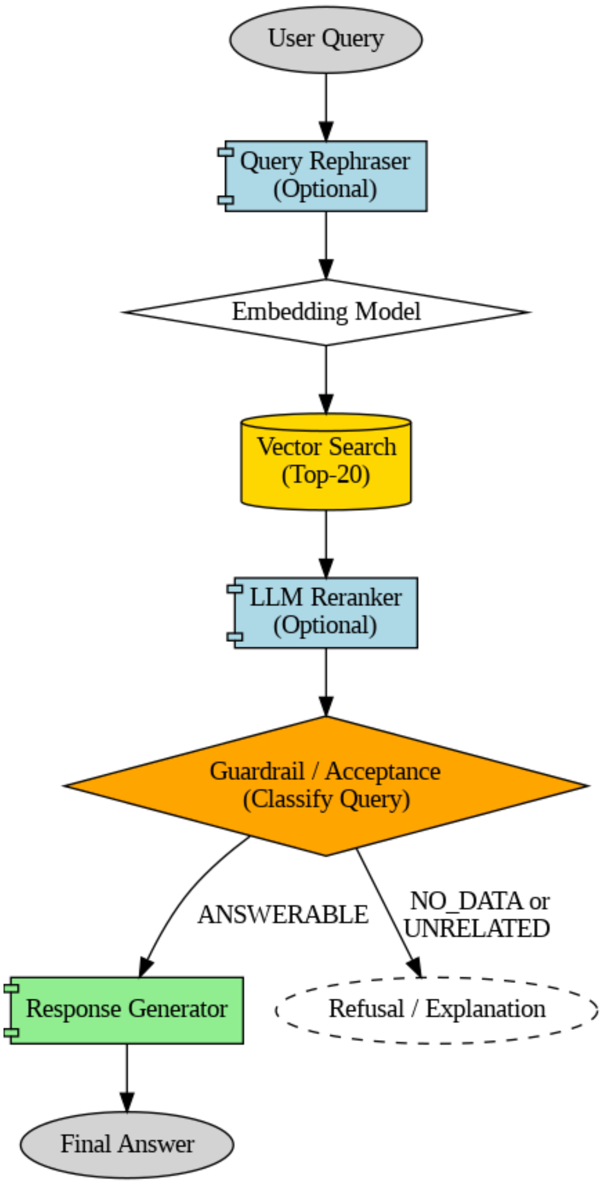

Figure 7: The query processing pipeline. Optional modules (rephrasing, reranking) are highlighted. The guardrail step ensures only answerable queries reach the final answer generator.

- ANSWERABLE: queries yielded high scores due to keywords overlap.
- NO_DATA queries resulted in much lower scores, matching only partial terms.
- UNRELATED queries often resulted in few than three retrieved chunks or very small scores, indicating a clear mismatch.

However, lexical retrieval failed when the query used synonyms not present in the text, or when the query was too short, indicating the value of semantic vector search used by our RAG pipeline.

5 Future Work

While the current system demonstrates the effectiveness of RAG for Anomaly Detection, several paths for improvement remain.

First, we aim to evaluate the pipeline's performance using smaller, open-source LLMs as the generator. Since our experiments relied on the Gemini 3 Pro, it would be useful to explore if the retrieval component of our RAG architecture can augment "weaker" models to perform at a similar level, enabling this way the offline deployment.

Second, we would like to observe the benefits of summarization in the preprocessing step. Instead of embedding raw text chunks, indexing summaries of each section might improve retrieval quality.

Finally, given that the dataset also presents a visual component, a multimodal RAG would be the next step. Instead of relying on text descriptions generated by the parser, we could use multi-modal embeddings to allow the system to retrieve figures directly, allowing the users to ask questions about the visual artifacts in the knowledge base.

6 Conclusion

The modular design allowed us to identify that semantic chunking combined with LLM-based reranking yields high retrieval accuracy. Furthermore, the implementation of an explicit acceptance criteria module proved to be an important component, achieving 100% accuracy in distinguishing between answerable queries, missing data, and unrelated topics. These results prove that for very specialized fields, retrieving external knowledge is an enhancement for the reliability of the assistance.

References

[1] David Cortes. Revisiting randomized choices in isolation forests. *arXiv preprint arXiv:2110.13402*, 2021.

[2] Hichem Dhouib, Alissa Wilms, and Paul Boes. Distribution and volume based scoring for isolation forests. *arXiv preprint arXiv:2309.11450*, 2023.

[3] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1), 2023.

[4] Sahand Hariri, Matias Carrasco Kind, and Robert J Brunner. Extended isolation forest. *IEEE transactions on knowledge and data engineering*, 33(4):1479–1489, 2019.

[5] Paweł Karczmarek, Adam Kiersztyn, Witold Pedrycz, and Ebru Al. K-means-based isolation forest. *Knowledge-based systems*, 195:105659, 2020.

[6] Jakub Lála, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodriques, and Andrew D White. Paperqa: Retrieval-augmented gen-

erative agent for scientific research. *arXiv preprint arXiv:2312.07559*, 2023.

[7] Julien Lesouple, Cédric Baudoin, Marc Spigai, and Jean-Yves Tourneret. Generalized isolation forest for anomaly detection. *Pattern Recognition Letters*, 149:109–119, 2021.

[8] Shutao Li, Kunzhong Zhang, Puhong Duan, and Xudong Kang. Hyperspectral anomaly detection with kernel isolation forest. *IEEE Transactions on Geoscience and Remote Sensing*, 58(1):319–329, 2019.

[9] Guillaume Staerman, Pavlo Mozharovskyi, Stephan Clémençon, and Florence d'Alché Buc. Functional isolation forest, 2019.

[10] Mikhail Tokovarov and Paweł Karczmarek. A probabilistic generalization of isolation forest. *Information Sciences*, 584:433–449, 2022.

A LLMs System Prompts

A.1 Data Preprocessing Prompts

A.1.1 PDF Parsing Prompt

```
The provided document is a scientific research paper.
Your goal is to extract ALL text, tables, and formulas into
Markdown format.
1. Transcribe text STRICTLY VERBATIM. Do not summarize, shorten, or
 rephrase.
2. Do not skip any sections, subsections, or paragraphs, even if
they look dense.
3. Maintain the reading order of the paragraphs and columns.
4. Don't include the figures/images. Instead, provide a description
 of the content of the figure, along with the caption.
5. Exclude the headers and footers of the pages.
```

A.2 Query Processing Prompts

A.2.1 Query Rephraser Prompt

```
You are an AI research assistant. The user is asking a question
about "Isolation Forests" or anomaly detection.
Rephrase the following question to be more specific and optimized
for a vector search engine.
- Keep the core intent.
- Expand technical acronyms (e.g., "IF" -> "Isolation Forest").
- If the query is a simple keyword, turn it into a full sentence.

Original Query: {query}
Rephrased Query:
```

A.2.2 Chunk Reranker Prompt

```
You are a relevance ranking system.
Query: "{query}"

Below are candidate text chunks retrieved for this query.
Rank them by relevance to the query.
Return ONLY the IDs of the top {top_n} most relevant chunks,
separated by commas.
If a chunk is completely irrelevant, exclude it.

Candidates:
{candidates_text}

Result IDs:
```

A.2.3 Acceptance Criteria (Guardrail) Prompt

```
You are the "Gatekeeper" for a Research Assistant about Anomaly
Detection (Isolation Forests).
Your job is to classify the relationship between the USER QUERY and
 the RETRIEVED CONTEXT.

USER QUERY: {query}

RETRIEVED CONTEXT:
{context_text}

Task: Analyze the inputs and output ONE of the following JSON
strings:

1. If the query is completely unrelated to Computer Science/Anomaly
 Detection (e.g. "How to cook pasta", "What is the weather"):
   {{"status": "UNRELATED", "reason": "The user is asking about [
Topic] which is outside the scope of this research assistant."}}

2. If the query IS related to the domain, but the Retrieved Context
 DOES NOT contain the answer:
   {{"status": "NO_DATA", "reason": "The query is relevant, but the
 provided papers do not discuss this specific detail."}}

3. If the Retrieved Context contains the answer:
   {{"status": "ANSWERABLE", "reason": "Context contains sufficient
 information."}}

OUTPUT JSON ONLY:
```

A.2.4 Response Generator Prompt

```
You are a Research Assistant. Answer the question using ONLY the
provided context.

Rules:
1. Cite your sources using the format [Title, Author, Year]
provided in the header of each source.
2. Do not hallucinate information not present in the text.
3. If the context has multiple papers, synthesize them.

Context:
{context_str}

Question: {query}

Answer:
```

A.3 Mistral 7B Prompts

A.3.1 Response Generator Prompt

```
You are a strict Research Assistant specializing ONLY in Anomaly
Detection and Isolation Forests.

YOUR RULES:
1. You answer questions strictly based on technical knowledge of
Isolation Forests.
2. NEGATIVE CONSTRAINT: If the user asks about unrelated topics (
like cooking, weather, sports, or general life advice), you must
REFUSE.
   - YOU MUST NOT provide the requested information "anyway."
   - YOU MUST NOT say "However, here is..."
   - You simply state: "I cannot answer this question as it is
unrelated to Anomaly Detection."
3. If the user asks a technical question you don't know, say "I do
not have sufficient data."
```

B Evaluation Questions

The complete list of 20 evaluation questions used in the experiments:

- **Answerable:**
  1. "What specific artifact does the standard Isolation Forest produce in anomaly score heat maps that Extended Isolation Forest aims to fix?"

2. "How does Extended Isolation Forest fix the bias issues found in the standard algorithm?"
3. "How does Functional Isolation Forest (FIF) project data using a dictionary and scalar products?"
4. "Why are anomalies assumed to be more susceptible to isolation in the kernel space according to the Kernel Isolation Forest paper?"
5. "How does Generalized Isolation Forest (GIF) improve upon Extended Isolation Forest regarding empty branches?"
6. "How does the K-Means Isolation Forest algorithm combine the partition strategy with the K-Means clustering algorithm?"
7. "What are the two hybrid algorithms introduced in the Extended K-Means Isolation Forest paper?"
8. "How does the Probabilistic Generalization of Isolation Forest (PGIF) use segment-cumulated probability?"
9. "How does the Renyi divergence relate to the aggregation functions in distribution-based scoring for Isolation Forests?"
10. "According to the 'Revisiting randomized choices' paper, how does non-uniform random splitting affect the detection of clustered outliers?"
11. "What is the specific application domain (type of images) that the Kernel Isolation Forest is designed to analyze?"
12. "Which benchmark metrics were used to evaluate the Extended K-Means Isolation Forest on the 13 datasets?"
13. "What is the 'visual elbow rule' used for in the context of Functional Isolation Forest experiments?"
14. "What is the main advantage of Generalized Isolation Forest (GIF) over Extended Isolation Forest (EIF) in terms of computation time?"

- **No Data:**
  1. "How does the performance of Isolation Forest compare to an LSTM-based Autoencoder on time-series data?"
  2. "What are the specific latency requirements for deploying Isolation Forest on an Arduino or edge device?"
  3. "How can I implement the Isolation Forest algorithm using the H2O.ai library in R?"
  4. "Does the 'Deep Isolation Forest' variant use Convolutional Neural Networks for feature extraction?"
- **Unrelated:**
  1. "What is the best recipe for pizza?"
  2. "Who won the FIFA World Cup in 2022?"