Developing a Retrieval-Augmented Generation (RAG) Pipeline for Domain-Specific Question Answering on O-RAN Specification Documents

Sri Sai Naga Venkata Adithya Swarna
sswar011@ucr.edu
University of California, Riverside

Abstract

This project creates a Retrieval-Augmented Generation (RAG) system that is designed to retrieve, extract, and generate answers from O-RAN specification documents. Due to the size and quantity of the O-RAN standards, manual retrieval of information is time-consuming and not effective. The proposed system processes over 115 O-RAN documents, converts them into embeddings, and retrieves the required information.

The core of this architecture is vector-based retrieval, where documents are preprocessed, chunked, embedded, and indexed into a ChromaDB vector store. During querying by the user, the most relevant document chunks are queried before being input to the Llama 2-7B language model to produce the response.

Of the most significant characteristics of this system is the ability to extract data from uploaded as well as stored documents. Issues still persist in retrieving data smoothly from both sources together, as the system now prioritizes stored or uploaded documents but not both simultaneously.

1 Introduction

The Open Radio Access Network (O-RAN) specifications define an interoperable and modular mobile network architecture to build a more vendor-neutral and flexible radio access network. However, the O-RAN specifications are lengthy, consisting of hundreds of pages in multiple documents. It is time-consuming and inefficient to get technical information from these documents manually. The engineers and researchers who use O-RAN need an efficient way to retrieve relevant information quickly without wasting time digging through large PDFs.

To address this challenge, I developed a RAG pipeline that enables natural language O-RAN specification query and response to be provided based on accurate document-based answers. This one involves vector-based retrieval using ChromaDB and generation of responses through Llama 2-7B to validate that answers derive from actual O-RAN documents and not strictly from the LLM's own knowledge. Further, the system allows for retrieval of data from stored and newly uploaded documents, enhancing the flexibility of dynamic document updating.

The overall objective of this project is to enable technical document retrieval automation via efficient searching over cached and freshly uploaded O-RAN specifications. Unlike traditional keyword-based searches, which can fail to capture semantic connections between words, this system utilizes embeddings and vector similarity search to fetch the most contextually suitable information. Once the contextually suitable document chunks are found, they are passed on to Llama 2-7B, which generates a structured and context-aware response.

For effective retrieval, the system provides metadata filtering and ranking functionality that identifies pertinent sections of documents by query intention. Dynamic chunking is also used to divide large documents into useful sections within token limits of the model to prevent noise in responses and improve retrieval effectiveness.

Some of the developmental challenges faced were: One of the largest problems was improving retrieval ranking because early retrieval had extraneous chunks, degrading answer generated precision. Metadata filtering was introduced to improve search results, but optimization can still be done. The second problem was handling ambiguous questions, e.g., distinguishing close phrases like "Near-RT RIC" and "Non-RT RIC." The system is now able to retrieve helpful chunks, but LLM sometimes incorrectly reads their context. In addition, retrieving information from both the uploaded and stored documents simultaneously is still an area for improvement because the system has a bias towards using one over the other.

2 Overview of Solution

Retrieval of informative content from O-RAN technical documents requires a RAG pipeline that integrates document preprocessing, embedding-based retrieval, and an answering language model. The goal was to enable users to query pre-stored O-RAN documents as well as newly-uploaded documents while maintaining accuracy and efficiency.

System Architecture The system consists of two primary phases: Document Ingestion and Retrieval Generation.

Document Ingestion (Preprocessing Storage)

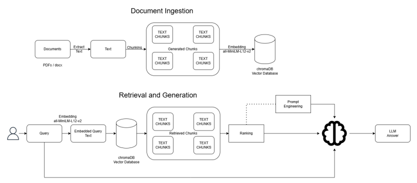


Figure 1. Complete Pipeline

- The system text is imported from O-RAN PDF and DOCX files.
- Dynamic chunking is applied instead of fixed-size chunking to make sure each chunk carries contextual meaning.
- Text chunks are embedded with the all-MiniLM-L12-v2 model to convert them into numerical vectors.
- Embeddings, along with metadata (document title, token length, source file, etc.), are queried into ChromaDB to facilitate fast retrieval.

Retrieval Generation (Query Processing LLM Response)

- When the user inputs a query, it is projected into an embedding using the same all-MiniLM-L12-v2 model.
- The query embedding is used to perform a similarity search in ChromaDB and retrieve the most relevant document chunks.
- Ranking is performed on retrieved chunks before passing them to the Llama-2-7B model to produce answers.
- The LLM generates an answer based on retrieved material only, reducing hallucinations.

Use of Tokens As both embedding and language models process text through tokens, token handling was essential in implementation: Embeddings (MiniLM-L12-v2): Each chunk was capped at 384 tokens for compatibility. Llama-2-7B Model: The model can handle 4096 tokens, but only a subset of fetched chunks can be utilized due to limitations of prompt length. Chunking: Document text was dynamically chunked to achieve the balance between retrieval precision and query efficiency, preventing chunks from being either too small (they become meaningless) or too big (they exceed token constraints). This structured organization allowed the system to deal with stored documents as well as newly inserted files, preventing retrieval from becoming rigid and inflexible for real-time O-RAN knowledge querying.

The system workflow is divided into two primary pipelines, as illustrated in the following diagrams:

2.1 Literature Survey

Key papers reviewed during the project provided foundational knowledge for system design. These include:

1. **"Sherlock on Specs: Building LTE Conformance Tests through Automated Reasoning"** – Highlighted the importance of automated reasoning in processing technical documents.
2. **"Extracting Procedural Knowledge from Technical Documentation:"** – Motivated embedding-based retrieval for procedural documents.
3. **"Large Language Models for Networking: Opportunities and Challenges"** – Highlighted the need for domain-specific LLM fine-tuning."

Key Insight: The literature emphasized modular architectures and the combination of retrieval with generative models, directly influencing the system's pipelines.

3 Detailed implementation

This section gives a detailed description of the core components of the system, such as document loading, preprocessing, and chunking, embedding, vector storage, retrieval, and response generation. The implementation is done in a structured pipeline manner to achieve scalability and efficiency when dealing with large amounts of O-RAN documentation.

3.1 Document Loading, Preprocessing, and Chunking

Text extraction, cleaning, and segmentation of O-RAN documents for vector-based retrieval is the first process in the RAG pipeline. The process keeps pertinent sections of the documents in a structured format without losing semantic meaning.

Text Extraction and Preprocessing The O-RAN documents are PDF and DOCX. The system extracts text and metadata from the documents and does text cleaning to improve retrieval accuracy.

PDFs are processed by a library that extracts structured text as well as formatting. Title, author, and date created are also extracted as metadata. DOCX files are processed differently using a document parsing library, but since they lack embedded metadata, only file-level attributes are noted.

To improve the quality of the extracted text, unnecessary whitespace, duplicate line breaks, and sequences of duplicate characters are removed. Footnotes, headers, and legal disclaimers not relevant to retrieval are filtered out as well.

Adaptive Token-Based Chunking After text extraction, documents are split into semantic chunks for efficient retrieval. Instead of using fixed-size chunking, the system applies token-based dynamic chunking using a special tokenization method. This ensures that each chunk is semantically complete while being within model token limitations.

This method allows the system to preserve sentence structure and guarantees embedding model and LLM compatibility without resorting to arbitrary text breaks. Dynamically adjusted chunks are calculated based on document structure with a limit of 512 tokens per chunk. There is a 100-token overlap between chunks to ensure continuity so that no vital information will be lost when breaking sections.

Earlier incarnations of the system used simple character-based chunking, which sometimes led to fragmented key phrases. By introducing token-aware chunking, retrieval relevance was significantly improved, and LLM responses were contextually more accurate.

3.2 Embedding Generation

Once documents are chunked, each chunk is converted into a numerical vector representation to enable semantic search. Contrary to keyword searches, embeddings allow the system to retrieve data based on contextual similarity rather than word matches.

The system utilizes a lightweight transformer-based embedding model that is designed for high-quality and efficient vector generation. The model maintains semantic relationships between words such that retrieved chunks are contextually relevant to user queries.

Each chunk is calculated and transformed into a dense vector, which can then be utilized in nearest-neighbor queries. Embeddings are also stored with metadata for rapid recall.

Batching is also used for increased efficiency in creating embeddings for multiple chunks at once. Vector normalization techniques are also employed for increasing similarity search performance. The embedding process was also optimized to queries specific to O-RAN to enhance the quality of information being retrieved.

3.3 Vector Store Management

After embeddings are created, they are stored in an efficient vector database, initially started with FAISS-based system. The database natively allows for fast similarity search and filtering by metadata.

Unlike FAISS, which required reindexing whenever new documents were added, the new system supports real-time document updates natively. It also allows filtering by metadata such as document title, section, and publication year, enabling users to filter search results by specific attributes.

The system uses hybrid search capability, combining semantic search (through embeddings) with keyword-based filtering for better retrieval accuracy. A unique identifier is assigned to each chunk, such that the retrieved content is traceable back to its source document.

Metadata filtering allows searchers to restrict queries to specific types of documents or give precedence to more recent publications. This improves relevance by ensuring that returned documents will be aligned with query intent.

The transition from FAISS to the current vector database resolved inherent scalability issues, enabling the better handling of stored documents and new documents being ingested. Dynamic document ingestion is now supported without compromising retrieval efficiency.

3.4 Retrieval and Question Answering

Once document embeddings are saved, the next step is to retrieve the most important information for a query. Semantic retrieval is performed by the system where the same model is used for document chunks to embed the queries. The query embedding is matched against stored embeddings in the vector database to find the most relevant document chunks.

Retrieval is optimized by a ranking mechanism that prefers the highest-matching chunks. To avoid irrelevant responses, metadata filtering is employed to restrict retrieval to specific document subsections or document types. The retrieved chunks are then passed to the language model for response generation.

To inhibit hallucinations, the LLM is specifically engineered to generate responses based solely on the retrieved content. Without any appropriate chunks, the system does not respond rather than generating hypothesized responses. Other techniques such as dynamic construction of the prompt make sure that the query and retrieved context are phrased in a way as to maximize response coherence.

The single biggest retrieval problem is handling vague queries. Words in O-RAN documentation such as "Near-RT RIC" and "Non-RT RIC" are the same in their context but are used with other meanings. Approximate matching algorithms are used in the system to improve the retrieval accuracy when ambiguous words are input. Improvement needs to be achieved, however, to ensure all ambiguous words are correctly interpreted every time.

3.5 UI-Based Real-Time Document Processing

Besides supporting pre-stored documents, the system also supports a UI-based document upload facility through which the users are allowed to load new documents dynamically. This allows real-time fetching from existing as well as newly uploaded documents.

If the user submits a document using the UI, it is treated separately from pre-stored documents. The uploaded document is text-extracted, chunked, embedded, and indexed similar to stored documents. The current retrieval model prioritizes the uploaded or stored documents individually rather than both jointly. To provide support for this feature is critical in achieving a complete integrated retrieval system.

The UI provides an interactive interface by which users are able to enter natural language queries. The system identifies the most suitable information from the vector database before feeding it into the LLM for response generation.

One of the largest flaws in the current UI is that it is not able to blend retrieval from uploaded and stored documents seamlessly. When entering new documents, the system is able to favor a new document or an existing one, as opposed to seamlessly combining the two sources. Enhancements in the future will involve simplifying retrieval logic to connect access to each document cohesively.

3.6 Dynamic Prompt Engineering for LLM Responses

In order to enhance the quality of answers generated by LLM, the system applies structured prompt engineering before forwarding retrieved document fragments to the LLM. Instead of forwarding the query and retrieved information as is, the system constructs a well-crafted prompt with specific instructions.

The structured prompt includes:

- Contextual guidance to obtain responses purely from retrieved information.
- Exclusion constraints to prevent hallucinations and unnecessary information.
- Query restructuring for increased retrieval relevance.

In development, there was a test where the LLM dynamically generated prompts and ranked retrieved chunks. The goal was to optimize the retrieved context before response generation. This introduced a great deal of latency from multiple inference calls, however, and proved infeasible despite optimization of retrieval accuracy. Instead, simpler ranking by embedding similarity and filtering by metadata were employed.

Prompt engineering is an area of future work, since varied query types require varied amounts of retrieval and formatted input representations.

3.7 Response Generation

Once relevant document chunks are retrieved and structured into a formatted prompt, the final step involves generating responses using the LLM. The model is guided to produce structured, fact-based answers while avoiding speculation or hallucinations.

The system utilizes Llama-2-7B, a large language model that generates human-like text based on the provided context. The retrieved chunks are passed in a structured format to ensure responses remain grounded in the retrieved content.

To optimize response generation:

- Token limits are managed to prevent overly long responses.
- Retrieval ranking ensures only the most relevant document chunks are included in the prompt.
- Truncation strategies prioritize key information when necessary.

A key challenge was maintaining concise and factual responses. The model sometimes expanded beyond the retrieved content, introducing information not present in the original documents. To address this, strict prompt constraints were implemented, reinforcing that responses must be generated solely from retrieved text.

Further improvements are required to balance completeness and accuracy, especially for queries needing multi-document synthesis. Fine-tuning the model with domain-specific data is a potential next step to enhance response quality.

4 Evaluation

The system performance was evaluated on the basis of retrieval accuracy, response quality, and overall efficiency, and four general areas of assessment were made

4.1 Retrieval Effectiveness

- The system was tested on factual, comparative, and summarization questions.
- Factual questions registered high similarity values (0.7+), indicating effective retrieval.
- Summarization questions registered slightly lesser values due to some loss of important information.
- Ambiguous queries (e.g., "Near-RT RIC" vs. "Non-RT RIC") showed lower accuracy, suggesting the need for improved retrieval ranking.

4.2 LLM Response Quality

- Target responses were matched against reference answers using BLEU and ROUGE scores.
- Factual accuracy was good, but the LLM would occasionally overgenerate beyond the retrieved information, reducing similarity scores.
- Elaborative responses were penalized against short reference answers, diminishing BLEU/ROUGE scores while being factually accurate.

4.3 Efficiency Scalability

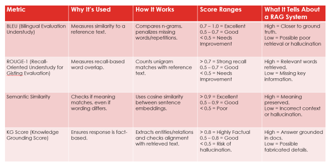
- Replacement of FAISS by ChromaDB significantly enhanced retrieval speed and indexing efficiency.
- Batch processing of the embeddings eased document ingestion and reduced processing time for big data.
- Efforts in retrieval ranking improved response time but not appreciably accuracy, and the work needed additional fine-tuning.

4.4 Hallucination Prevention

- The LLM was restricted to generating responses only from retrieved text.
- Most responses were fact-driven with a KG score of 0.6 and demonstrated room for further reduction of hallucinations.

Key Insights

- Strengths: Good factual query retrieval, improved efficiency with ChromaDB, and template-based response generation.

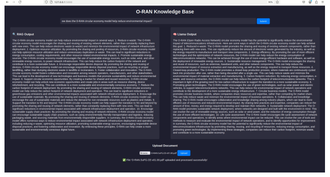


Metric	Why It's Used	How It Works	Score Ranges	What It Tells About a RAG System
BLEU (Bilingual Evaluation Understudy)	Measures similarity to a reference text.	Compares n-grams, penalizes missing words/repetitions.	0.7 – 1.0 = Excellent 0.5 – 0.7 = Good < 0.5 = Needs Improvement	High = Closer to ground truth. Low = Possible poor retrieval or hallucination

ROUGE-1 (Recall-Oriented Understudy for Gisting Evaluation)	Measures recall-based word overlap.	Counts unigram matches with reference text.	> 0.7 = Strong recall
			0.5 – 0.7 = Good
			< 0.5 = Needs Improvement
				High = Relevant words retrieved.
				Low = Missing key information.

Semantic Similarity	Checks if meaning matches, even if wording differs.	Uses cosine similarity between sentence embeddings.	> 0.9 = Excellent 0.5 – 0.9 = Good < 0.5 = Poor	High = Meaning preserved. Low = Incorrect context or hallucination.
KG Score (Knowledge Grounding Score)	Ensures response is fact-based.	Extracts entities/relations and checks alignment with retrieved text.	> 0.8 = Highly Factual 0.5 – 0.8 = Good < 0.5 = Risk of hallucination.	High = Answer grounded in docs. Low = Possible fabricated details.

Figure 2. Evaluation Metrics



127.0.0.1:5000

O-RAN Knowledge Base

ow does the O-RAN circular economy model help reduce environmental impact?

🔍 RAG Output

The O-RAN circular economy model can help reduce environmental impact in several ways: 1. Reduce e-waste: The O-RAN circular economy model encourages the sharing and reusing of existing network elements, rather than constantly replacing them with new ones. This can help reduce electronic waste (e-waste) and minimize the environmental impact of network infrastructure deployment. 2. Optimize resource utilization: By promoting the sharing and pooling of resources, O-RAN circular economy model can help optimize resource utilization and reduce unnecessary duplication or waste. This can lead to significant reductions in energy consumption, cooling requirements, and other resource-related environmental impacts. 3. Promote sustainable energy sources: The O-RAN circular economy model encourages the use of sustainable energy sources, such as solar, wind, and other renewable energy sources, to power network infrastructure. This can help reduce the carbon footprint of the network and contribute to a more sustainable future. 4. Encourage responsible device disposal: By promoting the sharing and reusing of devices, O-RAN circular economy model can encourage responsible device disposal practices, such as recycling or proper landfilling, rather than dumping electronic waste in landfills or incinerators. 5. Foster collaboration and innovation: The O-RAN circular economy model fosters collaboration and innovation among network operators, manufacturers, and other stakeholders. This can lead to the development of new technologies and business models that promote sustainability and reduce environmental impact. 6. Support circular economic opportunities: By creating a market for shared and reused resources, O-RAN circular economy model can help unlock new economic opportunities and create jobs in areas such as resource sharing, device refurbishment, and recycling. This can contribute to a more sustainable and environmentally conscious economy. 7. Reduce the carbon footprint of network deployment: By promoting the sharing and reusing of network elements, O-RAN circular economy model can help reduce the carbon footprint of network deployment and operation. This can lead to significant reductions in greenhouse gas emissions and other environmental impacts associated with network infrastructure deployment. 8. Encourage the use of sustainable materials: By promoting the sharing and reusing of network elements, O-RAN circular economy model can encourage the use of sustainable materials and reduce the demand for non-renewable resources such as plastics and metals. 9. Support the transition to 5G and beyond: The O-RAN circular economy model can help support the transition to 5G and beyond by promoting the sharing and reusing of network elements, rather than constantly replacing them with new ones. This can lead to significant reductions in environmental impact associated with network infrastructure deployment and operation. 10. Encourage sustainable supply chain practices: By promoting the sharing and reusing of network elements, O-RAN circular economy model can encourage sustainable supply chain practices, such as using environmentally friendly transportation and logistics, reducing packaging waste, and sourcing materials from environmentally responsible suppliers. In summary, the O-RAN circular economy model offers numerous ways to reduce environmental impact associated with network infrastructure deployment and operation, including reducing e-waste, optimizing resource utilization, promoting sustainable energy sources, encouraging responsible device disposal practices, and fostering collaboration and innovation. By embracing these principles, O-RAN can help create a more sustainable and environmentally conscious digital future.

🧠 Llama Output

The O-RAN (Open Radio Access Network) circular economy model has the potential to significantly reduce the environmental impact of telecommunications infrastructure. Here are some ways in which the O-RAN circular economy model can help achieve this goal: 1. Reduced e-waste: The O-RAN model promotes the sharing and reusing of existing network components, rather than replacing them with new ones. This can significantly reduce the amount of electronic waste generated by the industry, as well as the energy required to manufacture and transport new components. 2. Energy efficiency: By promoting the use of more efficient technologies and the optimization of network resources, the O-RAN model can help reduce the energy consumption of telecommunications networks. This can be achieved through the use of more energy-efficient hardware and software, as well as the deployment of renewable energy sources. 3. Sustainable resource management: The O-RAN model encourages the sharing and reuse of resources, such as antennas, baseband units, and other network components. This can help reduce the environmental impact of resource extraction and manufacturing, as well as the energy required to transport these resources. 4. Closed-loop production: The O-RAN model promotes a closed-loop production process, where materials are continuously cycled back into production after use, rather than being discarded after a single use. This can help reduce waste and minimize the environmental impact of material extraction and manufacturing. 5. Carbon footprint reduction: By reducing energy consumption, e-waste, and resource extraction, the O-RAN model can help reduce the carbon footprint of telecommunications networks. This is important in light of the growing need for low-carbon infrastructure to support the transition to a more sustainable economy. 6. Promoting green technologies: The O-RAN model encourages the use of green technologies, such as solar power and electric vehicles, to support telecommunications networks. This can help reduce the environmental impact of network operations and contribute to the development of a more sustainable energy infrastructure. 7. Circular business models: The O-RAN model promotes circular business models, where companies share resources and expertise, rather than competing for market share. This can help reduce waste and minimize the environmental impact of business operations. 8. Collaboration and knowledge sharing: The O-RAN model encourages collaboration and knowledge sharing among industry players, which can lead to more efficient use of resources and reduced environmental impact. By sharing best practices and expertise, companies can reduce the amount of time, money, and energy required to develop and maintain their networks. 9. Sustainable network deployment: The O-RAN model promotes sustainable network deployment, where networks are designed and built with the environment in mind. This can involve the use of renewable energy sources, such as solar or wind power, and the reduction of energy consumption through the use of more efficient technologies. 10. Life cycle assessment: The O-RAN model encourages life cycle assessment of network components and operations, to identify areas where environmental impact can be reduced. This can involve the use of tools and methods to measure and evaluate the environmental impact of network operations, and to identify opportunities for improvement. In summary, the O-RAN circular economy model has the potential to significantly reduce the environmental impact of telecommunications infrastructure by promoting sharing, reusing, and recycling of resources, reducing energy consumption, and promoting green technologies. By implementing these strategies, companies can reduce their carbon footprint, minimize waste, and contribute to a more sustainable economy.

Upload Document

O-RAN.SuFG.CE-v01.00.pdf

✅ File 'O-RAN.SuFG.CE-v01.00.pdf' uploaded and processed successfully!

Figure 3. Output Check 1

- Weaknesses: Improved retrieval ranking, improved query ambiguity processing, and smooth combination of stored and uploaded retrieval.

This testing confirms the system's ability to improve retrieval based on documents and identify the key areas where further improvement in ranking and response generation is needed.

5 Major Roadblocks and Lessons Learned

The development of the RAG system for O-RAN documents encountered several challenges, each providing key insights for future improvements.

5.1 Retrieval Challenges

- Ambiguous Query Handling: The system struggled to differentiate between closely related technical terms, leading to bad retrieval. While metadata filtering assisted, ranking logic refinement at a more granular level is needed.
- Ranking Optimization Trade-offs: Ranking optimization with additional processing added response time without significantly improving accuracy. Precision vs. speed trade-off remains a significant challenge.
- Unifying Stored and Uploaded Document Retrieval: The current system prioritizes either uploaded or stored documents but does not integrate them successfully. A hybrid retrieval approach needs to be used for enabling the dynamic addition of documents.

5.2 LLM-Generated Responses

- Hallucination Control: While retrieval constraints improved factuality, the LLM sometimes expanded on topics beyond retrieved content. Additional strict prompt constraints were helpful, but intermittent overgeneralization remains an issue.
- Prompt Optimization and Speed: Using the LLM to generate prompts and rank retrieved chunks improved some responses but introduced latency. Due to the added computational overhead, a less complex ranking process was chosen for efficiency.

Key Lessons Learned

- Precision vs. Speed: More complex retrieval mechanisms don't always result in better responses. Efficiency and accuracy must be finely balanced.
- Fine-Tuned Retrieval is Central: Aggressive filtering of metadata and structuring of chunks contribute more to retrieval accuracy than complex ranking mechanisms.
- LLM Guidance is Necessary: Strongly constraining response generation mitigates hallucinations but still demands a certain level of interpretability for open questions.

These observations and challenges direct the next phase of improvement for retrieval ranking, query processing, and general system responsiveness.

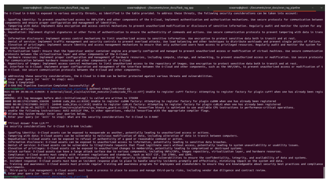
6 Artifacts

The project is a modular RAG pipeline with document processing, embedding generation, retrieval, and response generation modules. The core codebase has procedures for text extraction and chunking from O-RAN PDFs and DOCX documents, embedding generation using a transformer-based model, vector storage in ChromaDB, retrieving relevant document chunks, and LLM-based response generation. A UI-based system with real-time document upload and interactive querying is also provided. More than 115 O-RAN documents were ingested, transformed into structured JSON format, and indexed for easy retrieval. Fast similarity-based searches are facilitated by the system's embedding store and ChromaDB indexes, which support stored and uploaded documents.

The architecture is a structured flow: document ingestion pre-extracts and preprocesses text, vector storage stores embeddings for retrieval, query processing fetches relevant document chunks, and the LLM produces structured responses. The evaluation examined retrieval effectiveness, response accuracy, and system performance, reflecting strengths in fact-based query retrieval and areas for improvement in ranking and retrieving from stored and uploaded documents. These artifacts are a scalable platform for further research in retrieval ranking, response structure, and LLM fine-tuning.

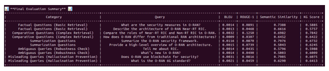
Visual Artifacts UI Screenshots: Show document upload, query input, and response generation. Terminal Outputs: Query input, and response generation. Evaluation Scores: Include similarity scores, retrieval accuracy, and response quality metrics.

GitHub Repository: The complete source code for the project, including source documents can be found at: Link



```
sswarna@ghost: ~/Documents/oran_docs/flask_rag_app    sswarna@ghost: ~/Documents/oran_docs/flask_rag_app    sswarna@ghost: ~/Documents/oran_docs/oran_rag_pipeline
he O-Cloud in O-RAN is exposed to various security threats, as identified in the table provided. To address these threats, the following security considerations can be taken into account:

. Spoofing identity: To prevent unauthorized access to VNFs/CNFs and other components of the O-Cloud, implement authentication and authorization mechanisms. Use secure protocols for communication between
 components and ensure proper configuration and management of identities.
. Tampering with data: Implement data encryption and access control mechanisms to prevent unauthorized modification or disclosure of sensitive information. Regularly audit and monitor the system for any
uspicious activity.
. Repudiation: Implement digital signatures or other forms of authentication to ensure the authenticity of commands and actions. Use secure communication protocols to prevent tampering with data in trans
t.
. Information disclosure: Implement access control mechanisms to limit unauthorized access to sensitive information. Use encryption to protect sensitive data both in transit and at rest.
. Denial of service: Implement traffic management and security controls to prevent overload attacks. Use load balancing and other techniques to distribute workloads and prevent single points of failure.
. Elevation of privileges: Implement secure identity and access management mechanisms to ensure that only authorized users have access to privileged resources. Regularly audit and monitor the system for
ny suspicious activity.
. Virtualization layer: Ensure that the hypervisor and/or container engine are properly configured and managed to prevent unauthorized access or modification of virtual machines. Use secure communication
 protocols between the virtualization layer and other components of the O-Cloud.
. Hardware resources: Implement proper configuration and management of hardware resources, including compute, storage, and networking, to prevent unauthorized access or modification. Use secure protocols
 for communication between hardware resources and other components of the O-Cloud.
. Repository of images: Implement access control mechanisms to limit unauthorized access to the repository of images. Use encryption to protect sensitive data both in transit and at rest.
0. Interface to O-Cloud: Implement proper configuration and management of the interface between the O-Cloud and other components, such as the VNF/CNFs, to prevent unauthorized access or modification of t
e O-Cloud. Use secure communication protocols between the O-Cloud and other components.

y addressing these security considerations, the O-Cloud in O-RAN can be better protected against various threats and vulnerabilities.
 Enter your query (or 'exit' to stop): exit
✅ Step 5 Completed!
 O-RAN RAG Pipeline Execution Completed Successfully! 🚀
swarna@ghost:~/Documents/oran_docs/oran_rag_pipeline$ python3 step5_retrieval.py
025-03-09 20:06:45.319659: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:477] Unable to register cuFFT factory: Attempting to register factory for plugin cuFFT when one has already been regis
tered
ARNING: All log messages before absl::InitializeLog() is called are written to STDERR
0000 00:00:1741570005.336430  164948 cuda_dnn.cc:8310] Unable to register cuDNN factory: Attempting to register factory for plugin cuDNN when one has already been registered
0000 00:00:1741570005.341571  164948 cuda_blas.cc:1418] Unable to register cuBLAS factory: Attempting to register factory for plugin cuBLAS when one has already been registered
025-03-09 20:06:45.362727: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
o enable the following instructions: AVX2 AVX512F FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
 Ollama RAG System Ready! Enter your queries below.
 Enter your query (or 'exit' to stop): What are the security considerations for O-Cloud in O-RAN?

 **Final Answer from LLM:**
he security considerations for O-Cloud in O-RAN include:

. Spoofing identity: O-Cloud assets can be exposed to masquerade as another, potentially leading to unauthorized access or actions.
. Tampering with data: O-Cloud assets can be vulnerable to malicious modification of data, including alteration of data in transit between computers.
. Repudiation: O-Cloud assets can be exposed to refusal of authenticity of a good or reasonable command or action.
. Information disclosure: O-Cloud assets can be exposed to unauthorized access to information, potentially leading to sensitive data breaches.
. Denial of service: O-Cloud assets can be vulnerable to illegitimate requests that flood legitimate users without access, potentially leading to system unavailability or usability issues.
. Elevation of privileges: O-Cloud assets can be exposed to unauthorized changes in membership, potentially leading to compromised or destroyed systems.
. Attack surface: O-Cloud assets can have a large attack surface due to various components, including VNFs/CNFs, images repository, virtualization layer, and hardware resources.
. Compliance: O-Cloud assets must comply with relevant regulations and standards, such as NIST CSF, ISO 27001, and GDPR.
. Continuous monitoring: O-Cloud assets must be continuously monitored for security incidents and vulnerabilities to ensure the confidentiality, integrity, and availability of data and systems.
0. Incident response: O-Cloud assets must have an incident response plan in place to handle security incidents promptly and effectively, minimizing impact on the system and data.
1. Training and awareness: O-Cloud assets must provide regular training and awareness programs for employees and stakeholders to ensure they are knowledgeable about security best practices and compliance
 requirements.
2. Third-party risk management: O-Cloud assets must have a process in place to assess and manage third-party risks, including vendor due diligence and contract review.
 Enter your query (or 'exit' to stop): exit
swarna@ghost:~/Documents/oran_docs/oran_rag_pipeline$
```

Figure 4. Terminal Output



```
📊 **Final Evaluation Summary** 📊
+----------------------------------------------+--------------------------------------------------------+--------+---------+---------------------+----------+
|                   Category                   |                         Query                          |  BLEU  | ROUGE-1 | Semantic Similarity | KG Score |
+----------------------------------------------+--------------------------------------------------------+--------+---------+---------------------+----------+
|     Factual Questions (Basic Retrieval)      |        What are the security measures in O-RAN?        | 0.0014 |  0.0891 |       0.7388        |  0.5885  |
|     Factual Questions (Basic Retrieval)      |     Describe the architecture of O-RAN Near-RT RIC.    | 0.0013 |  0.0468 |       0.6616        |  0.5757  |
|   Comparative Questions (Complex Retrieval)  | Compare the roles of Near-RT RIC and Non-RT RIC in O-RAN. | 0.0032 |  0.1250 |       0.6982        |  0.7842  |
|   Comparative Questions (Complex Retrieval)  | How does O-RAN differ from traditional RAN architectures? | 0.0009 |  0.0307 |       0.6452        |  0.6422  |
|           Summarization Questions            |        Summarize the O-RAN security framework.         | 0.0116 |  0.0870 |       0.7976        |  0.6732  |
|           Summarization Questions            |  Provide a high-level overview of O-RAN architecture.  | 0.0015 |  0.0739 |       0.5043        |  0.6245  |
|     Ambiguous Queries (Robustness Check)     |                   Tell me about RIC.                   | 0.0014 |  0.0435 |       0.5796        |  0.5988  |
|     Ambiguous Queries (Robustness Check)     |                  What's new in O-RAN?                  | 0.0014 |  0.0498 |       0.5003        |  0.6009  |
| Misleading Queries (Hallucination Prevention) |        Does O-RAN use blockchain for security?         | 0.0031 |  0.0816 |       0.7905        |  0.6075  |
| Misleading Queries (Hallucination Prevention) |             What is the O-RAN 6G standard?             | 0.0021 |  0.0459 |       0.6290        |  0.6413  |
+----------------------------------------------+--------------------------------------------------------+--------+---------+---------------------+----------+
```

Figure 5. Evaluation Metrics Output

7 Conclusion and Future Work

This project successfully developed a Retrieval-Augmented Generation (RAG) system for querying O-RAN documents by integrating vector-based retrieval with LLM-powered response generation. The system efficiently processes, embeds, and indexes over 115 O-RAN documents using ChromaDB, enabling hybrid retrieval that combines semantic search, keyword matching, and metadata filtering. Adaptive token-based chunking ensures contextual integrity, while structured prompt engineering helps constrain LLM responses to factual information, reducing hallucinations. Additionally, a UI system was implemented to facilitate real-time document uploads and interactive querying, allowing users to retrieve information from both pre-stored and newly added documents.

During development, LLM-driven adaptive prompt generation and chunk ranking were explored to refine retrieval and improve answer structuring. However, these methods significantly slowed down system performance, leading to their removal. Future improvements will focus on optimizing prompt engineering strategies to enhance response quality without added latency. The system currently retrieves from either stored or uploaded documents separately, necessitating a unified retrieval mechanism. Implementing Knowledge-Augmented Generation (KAG) can refine retrieval accuracy by structuring retrieved content before passing it to the LLM. Agent-based retrieval ranking using reinforcement learning could dynamically adjust retrieval prioritization based on query complexity. Further enhancements will include fine-tuning Llama-2-7B for O-RAN-specific terminology, real-time indexing updates, and improving ambiguous query handling. These refinements will help transform the system into an adaptive, intelligent document assistant capable of real-time technical knowledge retrieval.

References

[1] S. Agarwal, S. Atreja, and V. Agarwal. 2020. Extracting Procedural Knowledge from Technical Documents. *ArXiv preprint* (2020). arXiv:2010.10156 [cs.CL] https://arxiv.org/abs/2010.10156

[2] L. Bariah, H. Zou, Q. Zhao, B. Mouhouche, F. Bader, and M. Debbah. 2023. Understanding Telecom Language Through Large Language Models. *ArXiv preprint* (2023). arXiv:2306.07933 [cs.CL] https://arxiv.org/abs/2306.07933

[3] Y. Chen, D. Tang, Y. Yao, M. Zha, X. Wang, X. Liu, H. Tang, and B. Liu. 2023. Sherlock on Specs: Building LTE Conformance Tests through Automated Reasoning. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA, 3529–3545. https://www.usenix.org/conference/usenixsecurity23/presentation/chen-yi

[4] Y. Chen, D. Tang, Y. Yao, M. Zha, X. Wang, X. Liu, H. Tang, and D. Zhao. 2022. Seeing the Forest for the Trees: Understanding Security Hazards in the 3GPP Ecosystem through Intelligent Analysis on Change Requests. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 17–34. https://www.usenix.org/conference/usenixsecurity22/presentation/chen-yi

[5] Y. Chen, L. Xing, Y. Qin, X. Liao, X. Wang, K. Chen, and W. Zou. 2019. Devils in the Guidance: Predicting Logic Vulnerabilities in Payment Syndication Services through Automated Documentation Analysis. In *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, Santa Clara, CA, 747–764. https://www.usenix.org/conference/usenixsecurity19/presentation/chen-yi

[6] Y. Chen, Y. Yao, X. Wang, D. Xu, C. Yue, X. Liu, K. Chen, H. Tang, and B. Liu. 2021. Bookworm Game: Automatic Discovery of LTE Vulnerabilities Through Documentation Analysis. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1197–1214. https://doi.org/10.1109/SP40001.2021.00104

[7] Y. Huang, H. Du, X. Zhang, D. Niyato, J. Kang, Z. Xiong, S. Wang, and T. Huang. 2023. Large Language Models for Networking: Applications, Enabling Techniques, and Challenges. *ArXiv preprint* (2023). arXiv:2311.17474 [cs.NI] https://arxiv.org/abs/2311.17474

[8] S. Jero, M. L. Pacheco, and D. Goldwasser. 2018. Leveraging Textual Specifications for Grammar-based Fuzzing of Network Protocols. *ArXiv preprint* (2018). arXiv:1810.09478 [cs.CR] https://doi.org/10.1609/aaai.v33i01.33019478

[9] M. L. Pacheco, M. Von Hippel, B. Weintraub, and D. Goldwasser. 2022. Automated Attack Synthesis by Extracting Finite State Machines from Protocol Specification Documents. *ArXiv preprint* (2022). arXiv:2202.09470 [cs.CR] https://arxiv.org/abs/2202.09470

[10] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. Von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, and A. M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv preprint* (2019). arXiv:1910.03771 [cs.CL] https://arxiv.org/abs/1910.03771