Telco-RAG

Overview

Published Year: Aug, 2024

Friday, May 23, 2025

9:50 AM

# Keywords:

**Telco-RAG:** is an open-source RAG framework specifically designed to address the challenges of implementing RAG pipelines on highly technical content, particularly telecommunications standards like 3GPP documents.

**Large Language Models (LLMs):** are sophisticated models designed to understand, generate, and process text using deep learning techniques and vast amounts of training data, enabling them to grasp language nuances and context.

**Retrieval-Augmented Generation (RAG):** is a cost-effective, adaptable, and scalable paradigm that enhances LLMs by fetching knowledge from external sources in real-time to address queries, making it suitable for rapidly evolving fields.

**3GPP:** refers to the 3rd Generation Partnership Project, a set of telecommunication standards that Telco-RAG is specifically optimized for, due to the practical utility of a specialized chatbot and the limited knowledge of these standards in existing language models.

**Chunk Size:** determines the length of each text segment that the RAG processes at once, influencing the granularity of information retrieved.

**Context Length:** refers to the length of the context yielded by the retrieval component, affecting the amount of information provided to the LLM for generating responses.

**Embedding Models:** are algorithms that transform text into numerical representations, capturing the semantics of the text for effective similarity comparisons.

**Indexing Strategy:** is the method (like FAISS) by which the model assesses the relevance of each text chunk related to a given query, influencing the efficiency and accuracy of the retrieval process.

**Query Augmentation:** involves enhancing vague queries with technical terms, abbreviations, and generated candidate answers to improve the precision of the retrieval stage and the relevance of the retrieved information.

**NN Router:** is a neural network model designed to predict relevant 3GPP series based on queries, enabling selective loading of embeddings and reducing RAM usage.

**Prompt Engineering:** involves designing structured, dialogue-oriented prompts to ensure that the RAG maintains focus on the user’s question while comprehending the broader context, thereby improving the accuracy of LLM responses.

# Summary:

1. Applying standard RAG to complex, evolving telecommunications standards like 3GPP is challenging due to technical jargon, vague queries, and high resource needs.
2. The paper introduces Telco-RAG, an open-source framework designed specifically for this domain, featuring innovations like lexicon enhancement, query refinement, and an NN router for efficient retrieval.
3. Telco-RAG demonstrates significant improvements in accuracy and reduces RAM usage compared to baseline RAG methods when handling telecommunications documents.

The paper introduces Telco-RAG, an open-source Retrieval-Augmented Generation (RAG) framework specifically designed to address the challenges of applying Large Language Models (LLMs) to complex telecommunications standards documents, particularly those from the 3rd Generation Partnership Project (3GPP). Existing RAG setups are found inadequate for the highly technical and rapidly evolving nature of this domain.

Telco-RAG proposes a dual-stage pipeline: Query Enhancement and Retrieval. The architecture (Fig. 1) begins with the user query. The Query Enhancement stage involves four steps:

1. **Glossary Enhancement:** Utilizes a custom dictionary derived from the "Vocabulary for 3GPP Specifications" [14] to identify technical terms and abbreviations. Relevant definitions are integrated to enrich the query's embedding, improving similarity evaluation. These definitions are also included in the final prompt for the LLM.
2. **NN Router (Initial):** A neural network model is employed to predict the most relevant 3GPP series (out of 18 distinct series [15]) based on the query. This step optimizes retrieval efficiency by pre-selecting a subset of documents. The NN router architecture (Fig. 2) takes two inputs: the 1024-dimensional query embedding and an 18-dimensional vector where each entry is the inner product between the query embedding and the embedding of a 3GPP series summary description. It uses linear transformations, dropout, batch normalization, and softmax, weighting the two input streams with trainable parameters alpha and beta before a final classifier. The NN router was trained on a synthetic dataset of 30,000 questions from 3GPP Release 18 documents.
3. **Retrieval 1:** Performs a preliminary retrieval of document chunks using the initial query and the document sub-selection provided by the initial NN Router.
4. **Refine Query:** Uses a language model to generate plausible candidate answers based on the context retrieved in Retrieval 1. These candidate answers are then added to the original user query to create an "enhanced query," intended to clarify user intent and improve the accuracy of subsequent retrieval.

Following the Query Enhancement stage, the pipeline proceeds to the Retrieval Stage:

1. **NN Router (Final):** Re-runs the NN router using the \*enhanced\* query to potentially further refine the selection of relevant documents. (Based on the diagram, the router runs before each retrieval step).
2. **Retrieval 2:** Performs the final retrieval of relevant document chunks using the enhanced query and the document selection from the NN Router. The retrieved chunks form the final context provided to the LLM.

Finally, a state-of-the-art language model (e.g., GPT-3.5) generates the response based on the refined query and the retrieved context.

Key technical considerations and optimizations explored in Telco-RAG include:

* **Hyperparameter Optimization:** Through experiments, optimal settings for chunk size, context length, embedding models, and indexing strategies were identified. Smaller chunk sizes (125 tokens) were found to yield better performance for technical documents. Context length generally improved accuracy, although a drop was noted for lengths exceeding 1500 tokens unless specific prompt engineering was applied.
* **Embedding Models:** OpenAI's text-embedding-3-large (1024 dimensions), trained with Matryoshka Representation Learning [19], outperformed text-embedding-ada-002, showing an average accuracy gain of 2.29%.
* **Indexing Strategy:** The FAISS IndexFlatIP [13], based on the Euclidean dot product, slightly outperformed IndexFlatL2 (Euclidean distance) in 80% of experiments, while IndexHNSW performed considerably worse.
* **Query Augmentation Efficacy:** Lexicon-enhanced queries significantly improved accuracy on lexicon-focused questions (from 84.8% to 90.8% for the RAG pipeline). Enhancing queries with LLM-generated candidate answers based on preliminary context also demonstrated notable accuracy gains (up to 4.8% improvement).
* **RAM Usage Enhancement:** The NN router dynamically selects relevant 3GPP series, allowing selective loading of embeddings. This approach reduced average RAM consumption by 45% (from 2.3 GB for a benchmark RAG to 1.25 GB for Telco-RAG). The NN router proved significantly more accurate than GPT-3.5 and GPT-4 in identifying relevant 3GPP series for a given query.
* **Prompt Engineering:** A structured, dialogue-oriented prompt format including the original query, defined terms/abbreviations, retrieved context, and a \*repetition\* of the query before options and instructions was designed. This format resulted in a 4.6% average accuracy gain compared to a standard JSON format.

Experimental results using the TeleQnA benchmark dataset [3] showed that Telco-RAG consistently outperformed a benchmark RAG setup and a baseline GPT-3.5 model without RAG. Across different 3GPP releases and context lengths, Telco-RAG achieved an average improvement of 6.6% over the benchmark RAG and 14.45% over the baseline GPT-3.5.

In conclusion, Telco-RAG provides a robust framework for RAG in technical domains like telecommunications by addressing specific challenges related to complex documents, query ambiguity, resource constraints, and prompt formulation through tailored methods such as vocabulary enhancement, an NN-based document router, candidate answer generation for query refinement, and specialized prompt engineering. The framework is released as open-source to foster further research and application in the field.

Introduction

Friday, May 23, 2025

2:18 PM

What is RAG?

* retrieves relevant information from a large corpus or database and then uses that retrieved information to generate a response.

What is 3GPP?

* an umbrella term for a number of standards organizations that develop protocols for mobile telecommunications.

regulatory adherence/compliance?

* conforming to the laws, regulations, standards, and guidelines established by regulatory bodies and industry-specific authorities

Reducing hallucinations?

* minimizing the occurrence of fabricated or factually incorrect outputs generated by LLMs.
* occur when models produce information that is not grounded in the input data.

"Telecom Language Models: Must They Be Large?" [7]

* focuses on evaluating the capabilities of Phi-2, a smaller model
* challenging the assumption that LLMs must be large to perform effectively in this domain
* paper argues that SLMs can be viable alternatives to LLMs in telecom contexts, provided they are appropriately optimized for domain-specific requirements

"The Chronicles of RAG: The Retriever, the Chunk and the Generator" - focuses on optimizations to reduce hallucinations. Ref: <https://www.themoonlight.io/en/review/the-chronicles-of-rag-the-retriever-the-chunk-and-the-generator>

"RQ-RAG: Learning to Refine Queries for Retrieval Augmented Generation" [10]

* focuses on query refinement , enabling models to improve the quality of retrieval and generation processes by explicitly learning to rewrite, decompose, and disambiguate user queries
* teaches models to refine ambiguous or complex queries, ensuring more accurate retrieval of relevant information from external knowledge sources
* Ref: <https://www.themoonlight.io/en/review/rq-rag-learning-to-refine-queries-for-retrieval-augmented-generation>

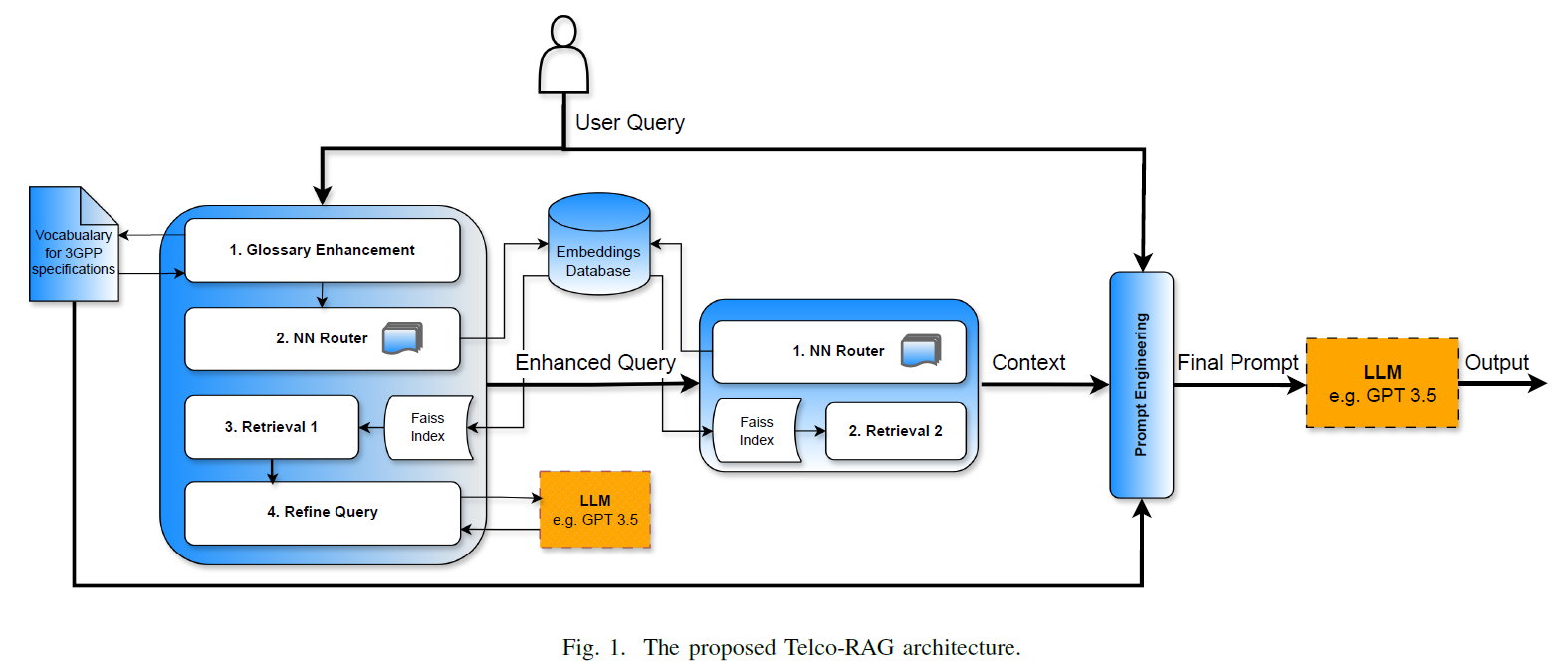
Methodology

Friday, May 23, 2025

12:38 PM

General flow:

* RAG pipelines start by splitting the document corpora into fixed-sized long (chunk size) segments called **chunks**.
* Using an **embedding** model, each chunk is transformed into a vectorial representation capturing the semantics of the segment.
* When a query is presented, the system identifies the relevant chunks by computing a **similarity** between the chunks’ embeddings and the query’s embedding.
* Lastly, RAG presents the relevant chunks, called the **context**, alongside the query to a LLM that generates the final response. (attention scores)



Challenges:

* sensitivity to hyperparameters
* vague user queries
* high RAM requirements
* sensitivity to the quality of the prompts

# Dual-stage pipeline:

* query enhancement stage
* and a retrieval stage

## Query enhancement stage:

|  |  |
| --- | --- |
|  | 1. Custom glossary of technical terms      1. Neural Network for selective document corpus extraction      1. Why NN? Reduces search space, thus increases efficiency      1. Followed by 'further refinement of first round of context' (not discussed in detail) |

Gaps (step 4):

* Probably to: reduce ambiguity, by changing generic terminologies to target specific stuff  
  e.g., "5G performance metrics" --> "5G NR throughput requirements as defined in 3GPP TS 38.101"
* potentially using techniques like attention mechanisms or semantic embedding to prioritize relevant terms
* normalize the query to align with standardized terminology, like converting informal terms (e.g., "cell tower") into formal 3GPP equivalents (e.g., "gNB" or "eNodeB") to improve retrieval accuracy
* Iterative Feedback Loop that iteratively narrows down the most relevant documents. This reduces the risk of retrieving irrelevant or outdated 3GPP specifications

## Hyperparameters:

* Chunk Size: Determines the length of each text segment the RAG processes at once.
* Context Length: Length of the context yielded by retrieval component.
* Embedding Models: Algorithms that transform text into numerical representations.
* Indexing Strategy: The FAISS index2 [13] by which the model assesses the relevance of each text chunk related to the given query.

Cosine similarity vs Dot product:

Think geometrically. Cosine similarity only cares about **angle difference**, while dot product cares about **angle and magnitude.** If you normalize your data to have the same magnitude, the two are indistinguishable. Sometimes it is desirable to ignore the magnitude, hence cosine similarity is nice, but if magnitude plays a role, dot product would be better as a similarity measure. Note that neither of them is a "distance metric".

<https://datascience.stackexchange.com/questions/744/cosine-similarity-versus-dot-product-as-distance-metrics?newreg=7377e9a3db254f82928808ee685bbc0d>

FAISS: <https://github.com/facebookresearch/faiss>

* Similarity search and clustering of dense vectors
* Algorithms that search in sets of vectors of any size,
* Up to ones that possibly do not fit in RAM
* Assumes as vectors & integers
* Vector comparison by L2 norm or dot prod (also cosine similarity => norm of dot prod)
* Low L2 norm [or] high dot prod => vectors are more similar (vice versa)
* Few methods work solely based compressed representation, thus extra costly
* HNSW and NSG like indexing structure make searching more efficient

## Query Augmentation:

Two major issues arise with vague queries:

|  |  |
| --- | --- |
| Lexicon-enhanced Queries  * Problem: capturing technical terms & abbreviations in question * Solution: Vocabulary for 3GPP Specifications (March 2024) * construct two dictionaries: one for abbreviations and another for terms with their definitions * This is a part considered a part of step-1 in above mentioned pipeline | Generating Candidate Answers  * Language modelling to generate (not specific) * Could be transformer based models mostly * Generates based on reduced search space of documents from step-3 in above pipeline   add these generated candidate answers to enhance the user’s query   * embedded enhanced query yields a superior final answer quality. |

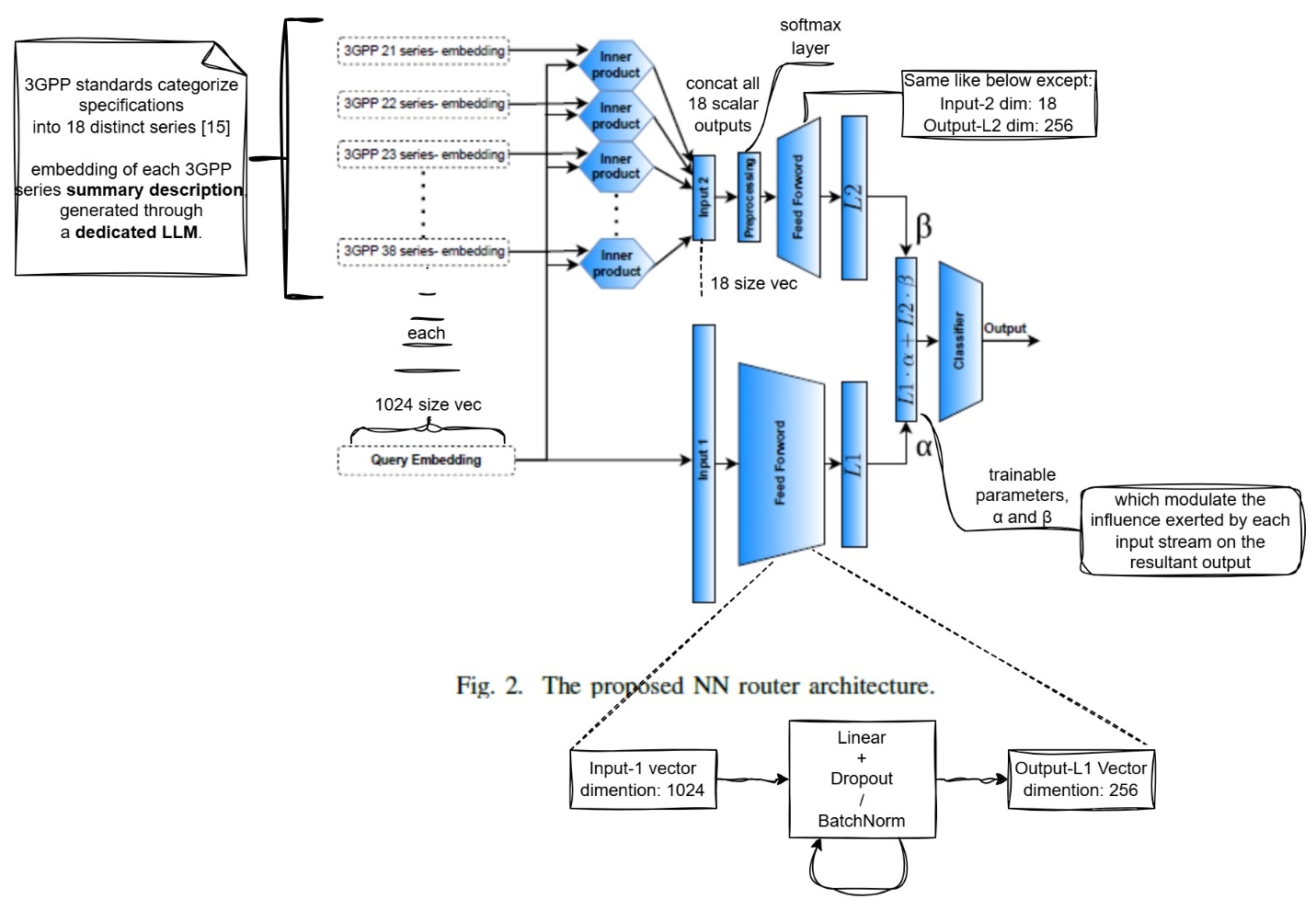
## Enhancing the RAM Usage of the Telco-RAG

To train the NN router, we created a synthetic dataset comprising 30,000 questions from 500 documents from 3GPP Release 18, and their originating series that served as target labels. The adoption of synthetic data for training and testing our NN router reduces the risk of overfitting the dataset on which we test Telco-RAG pipeline [16].

For highly technical documents, smaller chunks yield better performance. However, the smaller the chunks, the more the text segments to be processed by the RAG, which increases the required RAM resources. To deal with this issue, they have proposed this below approach:

"ChatGPT Outperforms Crowd-Workers for Text-Annotation Tasks"? [16]

* ChatGPT outperforms crowd workers in tasks such as relevance. classification, stance detection, topic identification, and frame detection.
* strong zero-shot performance , meaning it could handle annotation tasks without explicit training on the specific dataset.
* strong zero-shot performance , meaning it could handle annotation tasks without explicit training on the specific dataset.
* demonstrated higher intercoder agreement (a measure of consistency among annotators) suggests greater reliability in large-scale text analysis
* LLMs like ChatGPT can serve as cost-effective and scalable alternatives to traditional crowd-sourcing for text annotation.



## Prompt Engineering

* Starts with the query
* Followed by the definitions of the terms and abbreviations
* Finaly includes the generated context
* Note: query repetition before the related options and query instruction
* The designed format:

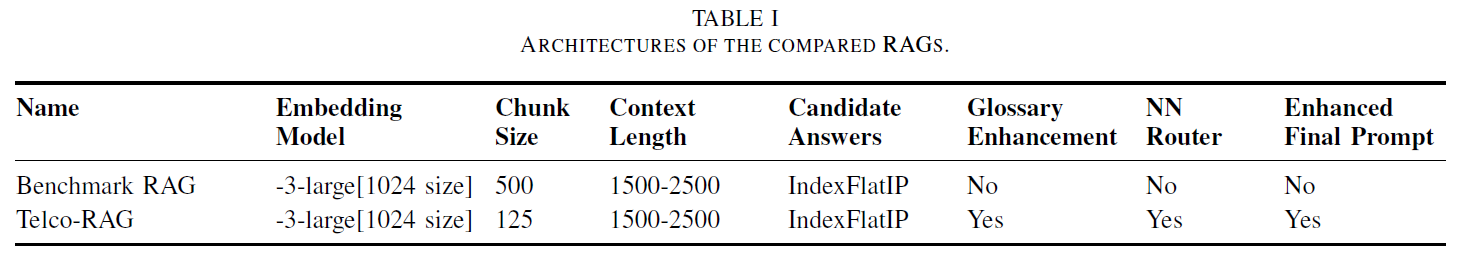
|  |
| --- |
| * + Please provide the answers to the following multiple choice question: <Question>   + Terms and Definitions: <Defined Terms>   + Abbreviations: <Abbreviations>   + Considering the following context: <Retrieved Context>   + Please provide the answers to the following multiple choice question: <Question>   + Options: <Options>   + Write only the option number corresponding to the correct answer. |

Experimental Results

Saturday, May 24, 2025

9:38 PM

* + The optimization set is composed of 2000 MCQs based on documents from 3GPP Rel.18.
  + The second set consists of the 1840 TeleQnA MCQs related to 3GPP documentations
  + The MCQ format, though very convenient for evaluation purposes
  + Not including the options in the retrieval process and use them solely to assess Telco-RAG accuracy.



"Matryoshka Representation Learning" (MRL)? [19]

* term "Matryoshka" refers to the nested structure of representations
* enables a single model to encode information at multiple granularities (scales)
* capture both fine-grained details and coarse-level abstractions
* enables models to use shorter or longer embeddings depending on the task
* single model is trained to generate representations at multiple levels, thus useful in resource-constrained environments
* in the context of recommendation systems, restructures user and item vectors to capture hierarchical interactions
* extended to multimodal tasks by allowing embeddings to scale across modalities
* often involves jointly training the model to minimize reconstruction errors across all scales

NOTE: This technique is used in this paper, that allows the shortening of the embedding vectors, which reduces computational and RAM requirements, while preserving a stronger performance.

|  |  |
| --- | --- |
| I] | Hyperparameters Optimization |
| Selecting the Embedding Model: | **-3-large(with MRL) vs -ada-002(predecessor)**   * on average, -3-large model, with a fixed embedding dimension of 1024, improves the accuracy of Telco-RAG by 2.29%, over the -ada-002 model. |
| Chunk Size Optimization: | **varying chunk sizes— 125, 250, and 500 tokens**   * there is an inverse relationship between chunk size and Telco-RAG accuracy * average improvement of 2.9% in accuracy when selecting as chunk size 125 tokens instead of 500 tokens |
| Context Length Optimization: | Linear regression fitted on the RAG accuracy vs diverse set of context lengths, with different configurations show an ascending trend of the accuracy as a function of context length.    Note: Drop in performance when the context length gets larger than 1500  Tokens is reduced by presenting the query twice, before and after the context |
| Indexing Strategy Selection: | * 1. **IndexFlatL2**      + **Euclidean distance**   2. **IndexFlatIP**      + **Euclidean dot product**      + **outperformed IndexFlatL2 in 80% of our experiments.**   3. **IndexHNSW**      + **approximate method for efficient searching in high-dimensional data spaces using Euclidean distance**      + **Least performance of all** |

|  |  |
| --- | --- |
| II] | Query Augmentation |
| Lexicon-enhanced Queries: | * + Enhances the baseline LLM accuracy on lexicon questions from 80.2 % to 84.8%   + Lexicon-enhanced queries have achieved an accuracy rate exceeding 90% on these questions, gaining 6% compared to the same RAG pipeline without the lexicon enhancement |
| Enhancing User’s Query With Candidate Answers: | Addition of candidate answers considerably improves the query embedding representations. |

|  |  |
| --- | --- |
| III] | RAM Usage Analysis in the Telco-RAG |
| **125-token chunk size** | * + Increases the RAM requirements   + But designed NN router can tackle this issue   + NN router dynamically selects the number of documents processed by the Telco-RAG pipeline as opposed to a fixed number of documents processed by the Benchmark RAG architecture.   + Variability in RAM usage among different queries, which results in the probability density function (PDF) |
| **assess the capability of the NN router** | * + compared it to GPT 3.5 and GPT 4, asking to indicate the top k most related 3GPP series.          * + reduces the consideration of irrelevant content.   + lowers the computational complexity of the retrieval steps &   + also the overall resources needed for processing the retrieved content. |

|  |  |
| --- | --- |
| IV] | Enhanced Prompt Formatting |
| Analysis results: | * + revealed a 4.6% average gain in accuracy, compared to the original JSON format   + suggests that human-like query structures can significantly elevate the contextual understanding |

|  |  |
| --- | --- |
| V] | Overall Performance |
| Comparative analysis: | * + consider three groups of MCQs:     - Rel. 17 MCQs,     - Rel. 18 MCQs, and     - the overall set of TeleQnA MCQs related to 3GPP documentations.   + compare the performance of:     - GPT 3.5 with Telco-RAG,     - GPT 3.5 with the Benchmark RAG, and     - GPT 3.5 without RAG.          * + highlights that Telco-RAG leads to notable gains in all the experiments. |

Conclusions & References

Friday, May 23, 2025

9:33 AM

The provided solutions are general and can deal with frequent challenges encountered in building RAG pipelines for highly technical domain

* + Source code: <https://github.com/netop-team/telco-rag>
  + [8] Ideal Chuck Size: <https://www.llamaindex.ai/blog/evaluating-the-ideal-chunk-size-for-a-rag-system-using-llamaindex-6207e5d3fec5>
  + Chat3GPP (2025 paper): <https://arxiv.org/html/2501.13954v1>
  + Local LLMs & RAG(not working currently): <https://medium.com/@bijit211987/chat-with-your-pdfs-locally-using-llm-and-rag-69ace751fc98>