Med-RAPTOR Agent:
A Generative and Agentic AI System for Evidence-Based Medical Question Answering

Arpit Mishra
SR: 24131
arpitmishra@iisc.ac.in

Archie Gaur
SR: 24826
archiegaur@iisc.ac.in

Rahul Dewangan
SR: 24670
drahul@iisc.ac.in

Sreeja Sri
SR: 19167
sreejasri@iisc.ac.in

Abstract

Large Language Models (LLMs) have recently become strong at answering complex questions, but their reliability in the medical domain is still a major concern.
Even minor inaccuracies or hallucinated facts can lead to harmful decisions. This is our motivation for building the Med-RAPTOR Agent, a system designed to combine generative AI with a structured verification process to ensure that answers are not just fluent, but also grounded in real medical evidence.
Our approach combines a RAPTOR-style hierarchical retrieval system with an agent-driven verification loop. The model first explores the available medical literature, organizes the information into a multi-level structure, and retrieves the most relevant references.
A generative model uses this information to draft an initial answer. This answer is then broken down into its individual claims, and each of them is checked using a Natural Language Inference (NLI) model.
Whenever a claim lacks strong supporting evidence, the agent searches for additional references and rewrites the answer accordingly.
By the end of the project, we aim to demonstrate that this retrieval–generation–verification pipeline can significantly reduce hallucinations and produce answers that are both accurate and supported by citations. The Med-RAPTOR Agent represents an effort to bring trustworthy, evidence-based AI closer to real-world medical applications.

1 Introduction

1.1 Background and Context

Despite their remarkable advancements, LLMs continue to have difficulty in situations when factual truth is crucial. This is crucial when answering medical questions since inaccurate information can mislead patients, students, or healthcare professionals.Although retrieval-augmented models aid in incorporating outside information into the generation process, they are not always able to guarantee that the final result is perfect. Newer methods, such hierarchical retrieval systems like RAPTOR, which provide a more structured perspective of massive text archives, have been inspired by this challenge.
When combined with agentic reasoning, this enables the model to verify and improve its own output. Together, these concepts provide a promising basis for developing medical quality assurance systems that are safer and more reliable.

1.2 Objectives and Significance

Our main objective is to build and test the **Med-RAPTOR Agent**, a system that integrates retrieval, generation, and verification into a single pipeline. More broadly, our objectives are to:

- Identify and retrieve relevant medical research using a hierarchical retrieval structure.
- Generate concise and readable answers.
- Assess each factual claim using NLI-based verification models.
- Automatically correct any unsupported statements through iterative refinement.

The broader significance of this project lies in demonstrating how generative AI can be made more dependable for high-stakes applications. Ensuring that each response is grounded in evidence is a step toward safer and more responsible medical AI systems.

2 Methodology

2.1 Technology Stack

To build the Med-RAPTOR Agent, we rely on the following tools and components:

- **Libraries:** PyTorch, HuggingFace Transformers, sentence-transformers, bitsandbytes, accelerate, NLTK.
- **Embedding Models:** Sentence-BERT
- **NLI Models:** DeBERTa-v3-base (cross-encoder with MNLI)
- **Orchestration Tools:** LangChain
- **Datasets:** PubMedQA
- **Generation Models** Gemma-2b-it
- **Evaluation Tools:** Custom EM/F1 implementation with token normalization.

For our system, we chose models that offer strong performance in general-language tasks while still transferring well to biomedical content. The goal was to use lightweight but capable models that do not require expensive domain-specific fine-tuning.

Sentence-BERT: This model has been trained on a wide variety of question–answer pairs, which makes it particularly effective for semantic similarity tasks. Its ability to generalize helps it perform surprisingly well in medical settings through transfer learning.

DeBERTa-v3-Base: Although it is trained primarily on MNLI data, DeBERTa has consistently shown strong results in NLI tasks across different domains. Its robust reasoning ability allows it to evaluate medical claims accurately without additional domain adaptation.

Gemma-2B-it: This is a compact but powerful instruction-tuned model from Google, developed using the same research foundations as the Gemini family. Its small size and strong reasoning capabilities make it an ideal choice for efficient answer generation.

2.2 Planned Workflow

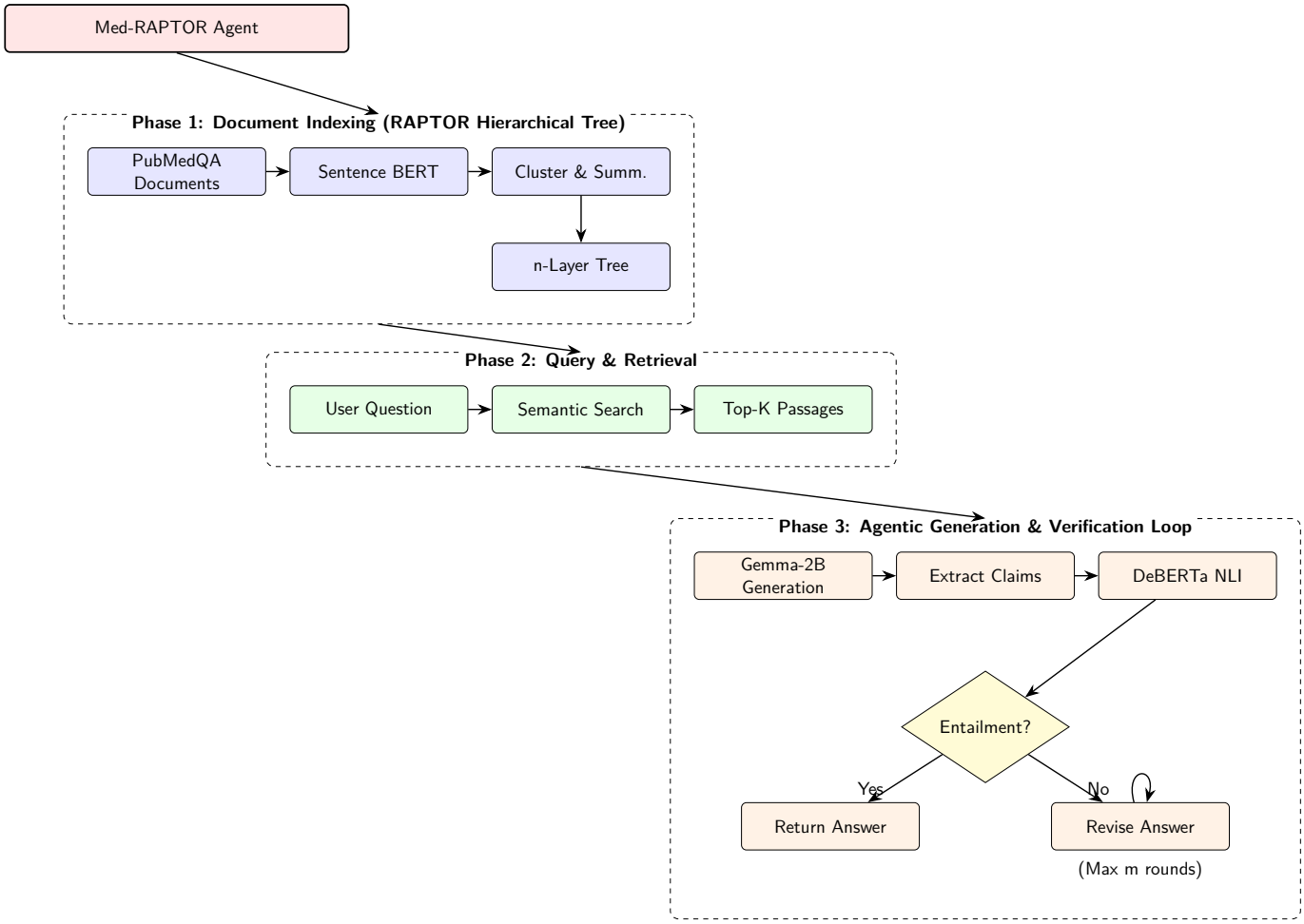


Figure 1: System architecture of the Med-RAPTOR Agent with RAPTOR indexing, semantic retrieval, and an agentic verify-revise loop.

2.2.1 1. Data Collection & Pre-processing

We begin by choosing a medical QA dataset and preparing it for downstream processing. This involves cleaning the text, normalizing formats, and structuring the question–context pairs so that they can be effectively embedded and retrieved.

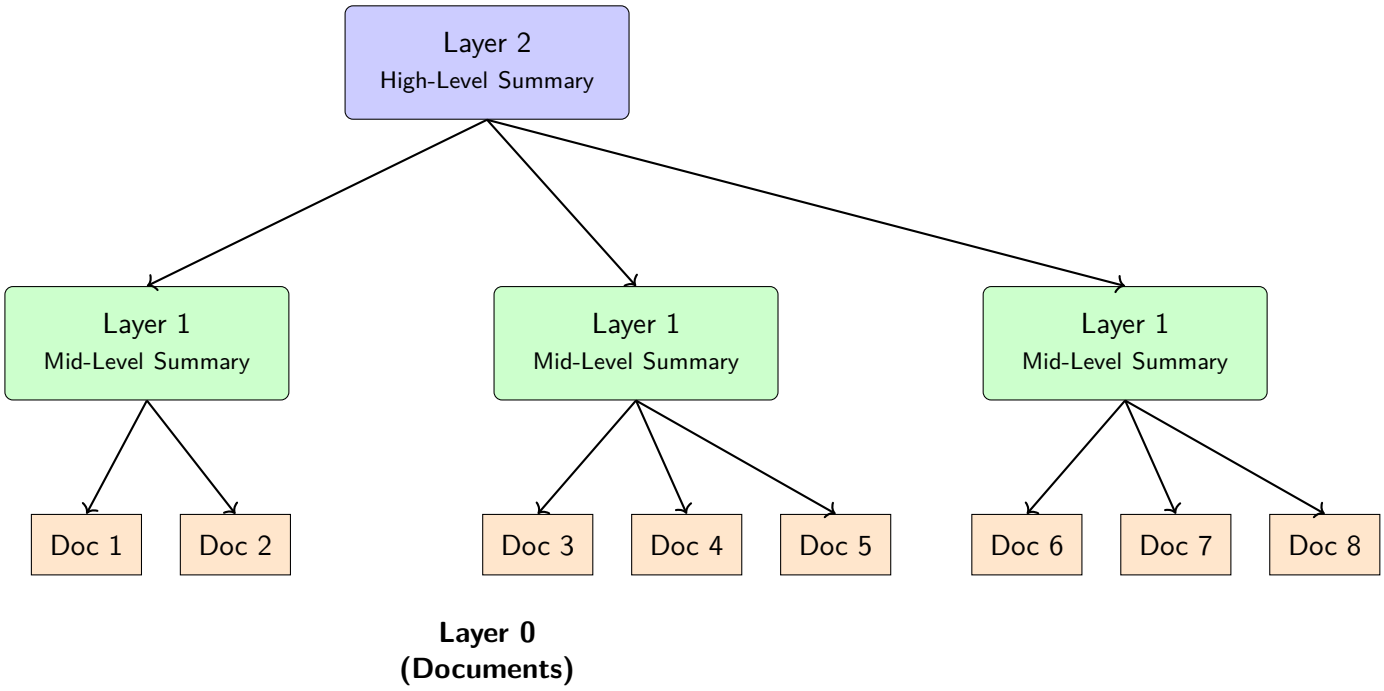
2.2.2 Building the RAPTOR Hierarchical Index

To support efficient retrieval, we embed all documents using Sentence-BERT and treat each one as a leaf node in the hierarchy. Similar documents are then clustered based on embedding similarity, and each cluster is summarized. These summaries are recursively clustered and summarized again, producing a multi-level hierarchical structure that allows the agent to retrieve both fine-grained passages and higher-level abstractions.

2.2.3 Hierarchical Indexing

RAPTOR constructs a multi-level retrieval structure that allows the system to access both fine-grained passages and higher-level summaries. The process involves:

- Adding each document to the tree as a leaf node,
- Computing embeddings and clustering similar documents together,
- Generating abstractive summaries for each cluster,
- Recursively creating parent nodes from those summaries.



This hierarchical layout enables flexible retrieval: the model can pull detailed text when needed or rely on broader summaries when the query requires higher-level context.

2.2.4 Baseline RAG System

As a baseline, we implement a traditional Retrieval-Augmented Generation (RAG) pipeline using RAPTOR's built in heirarchial retrival mechanism. The retrieved passages provide context for the generative model to craft an initial answer.

2.2.5 Med-RAPTOR Agent (Verify-and-Revise Loop)

The core component of this project is the agentic refinement loop:

1. The system breaks the generated answer into smaller factual units,
2. An NLI model evaluates each claim against the retrieved evidence,
3. Unsupported claims trigger answer revision, and
4. The process repeats until all claims are validated.

This creates a self-correcting mechanism that strengthens the reliability of the final output.

2.2.6 Evaluation & Reporting

Our evaluation includes both quantitative and qualitative components. For datasets with explicit answer labels, we compute metrics such as Exact Match (EM) and F1 score. For more open-ended biomedical questions, we rely on factuality metrics and manual analysis. We will also compare the performance of the baseline RAG model, the RAPTOR-enhanced system, and the full agentic approach.

3 Individual Member Contribution

- **Archie Gaur – Core Logic:** Designed and implemented the main agentic loop that connects retrieval, answer generation, and verification. Built the iterative refinement process so the agent can improve its answers based on feedback. Wrote the prompts for both the initial QA and the revision steps to ensure evidence-based responses. Also added checks to prevent infinite revision loops and to help the agent reach a stable final answer.
- **Arpit Mishra – Verification & Claim Extraction:** Implemented the NLI-based verification module using the DeBERTa-v3 cross-encoder. Created the claim extraction function to break down the model's answers into clear, verifiable statements. Also developed a cleaning utility to remove unwanted artifacts from the output and improve verification accuracy.
- **Rahul Dewangan – Infrastructure & Model Integration:** Built the `ModelManager` to load and manage all models efficiently so that each model is initialized only once. Integrated the Gemma 2B model into the LangChain framework and handled the technical setup, including imports, device configuration, and environment settings. Added the SentenceTransformer embedding model for building the RAPTOR index.
- **Sreeja Sri – Data Pipeline & Evaluation:** Developed the data ingestion pipeline (`build_raptor_index`) to load the PubMedQA dataset into the RAPTOR structure. Implemented the evaluation process to test the system against ground-truth data. Wrote the scoring functions and the parser used to standardize answers. Also improved the retrieval step by tuning `top_k` and preparing the retrieved text for use as evidence.

4 Conclusion

In this project, we built **Med-RAPTOR**, a system that combines retrieval, generation, and verification to answer medical questions in a more reliable way. The agent uses a lightweight generative model together with an **NLI-based verifier**, and the RAPTOR index helps it gather useful evidence before forming an answer. By allowing the model to check and revise its own responses, the system produces more consistent and well-supported outputs.

We also designed a complete data and evaluation pipeline using the **PubMedQA dataset**. This helped us measure how much verification and refinement improve the quality of the answers. The overall architecture is modular, so individual components-models, retrieval methods, or verification steps-can be replaced or extended as needed.

Med-RAPTOR shows that combining smaller models with careful verification can lead to dependable results, especially in sensitive areas like healthcare. The project also highlights how agentic workflows and structured reasoning can make AI systems more trustworthy and easier to evaluate.