Enhanced RAG (Retrieval-Augmented Generation) Based Chatbot For Educational Institutions

Harshita Sharma

harshita02.sharma@gmail.com

ABSTRACT

This document presents a two-stage prototype of a Retrieval-Augmented Generation (RAG) based chatbot tailored for educational institutions. The chatbot allows students to ask questions about university operations, academic calendars, faculty, and syllabi using institutional documents as its knowledge base. The goal of this document is to explain the reasoning, design, techniques, and improvements made through both prototypes. It is adapted from a full-length university submission into a concise technical document for public readers and potential collaborators.

INTRODUCTION

Chatbots have become a powerful tool in improving user interaction, especially within domains that require access to structured and semi-structured data. In educational institutions, students often have frequent questions related to schedules, academic processes, course structures, and faculty information. Addressing these queries through a static FAQ page is inefficient, especially when students require contextual understanding or multi-turn conversations.

The RAG framework is well-suited to such dynamic information retrieval tasks, allowing a model to access institution-specific data and generate answers accordingly. This project applies RAG principles to develop a chatbot that answers student queries based on documents sourced from a specific university.

PROBLEM STATEMENT

Students often face delays and confusion when trying to access accurate academic information. Even though the information about happenings at the university is being made readily available through emails and multiple university portals, sometimes students find it difficult to navigate through such a vast amount of information. Traditional university portals are poorly organized or unintuitive, leading to increased dependency on faculty, staff, or peers. This system aims to:

- Provide a self-serve conversational interface for students
- Deliver real-time responses based on university-authenticated sources
- Preserve conversational context across multiple interactions

WHY RAG FOR THIS USE CASE?

RAG (Retrieval Augmented Generation) is a framework to make a large language model answer queries based on a particular domain. It's basically a cost-effective technique to train a model on the data it has not been trained on initially. For scenarios as simple as an educational institution, with the help of RAG, we can train a model on university-related documents and develop a chatbot that can easily provide answers to some common or even more complex questions related to a university.

Compared to fine-tuning a language model, RAG allows for:

- Real-time knowledge updates without retraining
- Fine-grained control over the document source
- Easier debugging and interpretation of outputs

OVERVIEW OF PROTOTYPE V1

Prototype V1 implements a foundational RAG-based chatbot using LangChain and the Google Gemini API. It leverages a vector database to index and retrieve content from university documents.

For experimentation purposes, documents specific to Vellore Institute of Technology (VIT), Vellore, were used. However, the chatbot is designed to be adaptable; replacing these documents with those from any other institution will allow it to function similarly without altering the underlying logic.

Documents Used:

- Academic Calendar (Fall/Winter 2024–2025)
- General Info about VIT
- Faculty list from the School of Computing (SCOPE)
- Syllabi for M.Tech and MCA programs

Pipeline**:**

1. PDFs converted to plain text
2. Text split into chunks and embedded using Gemini's Embeddings API
3. Stored in a vector database (ChromaDB)
4. User queries are matched with relevant chunks
5. Prompt generated with retrieved context and passed to the Gemini model.

This version uses a history-aware retriever for multi-turn conversation management.

OVERVIEW OF PROTOTYPE V2

Prototype V2 enhances both response accuracy and retrieval robustness by incorporating selected advanced RAG techniques. The RAG pipeline includes multiple stages where sophisticated methods can be applied; in this prototype, a deliberate combination of a few key techniques is used to strike a balance between performance and simplicity.

Key Enhancements:

- Query Rewriting: Reformulates user queries into multiple variations to capture missing keywords.
- Reciprocal Rank Fusion (RRF): Combines ranked retrievals from different query forms for better context.
- Few-shot Prompting: Uses examples in the prompt to guide the model toward more accurate and consistent responses.

These techniques were selected based on insights from the DeepSeek R1 paper, which advocates returning to core principles to improve model accuracy and efficiency, rather than merely scaling computational resources. This aligns with the project's objective to remain lightweight while improving output quality.

Pipeline:

1. User query rewritten into 5 variations
2. Parallel document retrieval for each variation.
3. Combine the documents and find the Reciprocal rank of the documents.
4. Arrange the documents according to their computed ranks.
5. Documents and questions are given to the model.
6. The model uses examples provided in prompts to generate the appropriate answer.

RESULTS & OBSERVATIONS

Due to the token generation limitations imposed by the Gemini API, testing was conducted using a limited set of fifteen user queries. These fifteen questions served as the evaluation dataset for assessing the chatbot's performance. A comparative analysis — included in the original paper

linked in this repository — showcases the differences in response quality before and after implementing the enhancements in Prototype V2. While there are publicly available benchmark datasets for evaluating RAG models, they were not suitable for this project, as the chatbot is trained on domain-specific educational documents. Hence, using those datasets would not yield meaningful insights into this model's accuracy. The selected fifteen questions were crafted to align with real student queries from the educational domain. From the observations, the following things can be concluded:

- Accuracy: Responses from V2 were notably more accurate in edge cases where user phrasing was vague.
- Retrieval Diversity: RRF produced more contextually rich results by merging independent retrievals.
- Usability: V2 was more conversationally resilient over long sessions.

LIMITATIONS

1. The Gemini API imposes a restriction on the number of tokens processed in a single request, which can be problematic when working with large documents.
2. The system may experience slower performance due to the use of Streamlit for the user interface.
3. Extracting content from PDFs containing unstructured elements (like tables or images) can result in incomplete or low-quality text conversions.
4. Document updates must be handled manually, as there is currently no dynamic update mechanism integrated into the pipeline.

FUTURE WORK

1. Integration with university databases and web scraping tools will be implemented to ensure that information remains current and updates are automatically reflected in the model.

2. A more refined and user-friendly interface will be designed, customized to the branding and structure of specific universities.
3. Advanced routing and retrieval techniques will be explored to further improve the chatbot's response accuracy and adaptability.

APPENDIX

For the original university submission, see advanced_rag_academic_paper.pdf in the /prototype_v2/paper folder.