Assessment and Enhancement of RAG-Enhanced Document Generation using Transformer models

Abhinav Lodha Atharva Attarde Deepak C Nayak Jignesh Shah

EYSIP 2024

Introduction

Retrieval-Augmented Generation (RAG) combines retrieval-based and generation-based models to enhance text generation accuracy. By retrieving relevant information from a large corpus, RAG models ground their responses in real-world data, reducing inaccuracies and improving contextual relevance. This approach is particularly effective for applications like question answering, content creation, and interactive assistants, ensuring reliable and context-aware outputs.

Objectives

Assist e-Yantra theme instructors by providing easy access to previous years' resources and technical documentation, thereby enabling the creation of new tasks and projects.

- Advanced Search:
  - Implement advanced search functionalities to fetch relevant tasks from previous offerings.
- Task Document Generation:
  - Generate new tasks based on previous task documents utilizing the RAG system.
- Evaluation:
  - Evaluate various techniques for RAG.

Technology Stack

LangChain, LlamaIndex, python, docker, NumPy, HUGGING FACE, pandas, SBERT.net Sentence-Transformers, Chroma, ragas, OpenAI, unsloth, scikit learn, PyTorch

Features

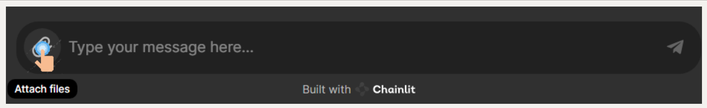


Figure 1. Adding new files.

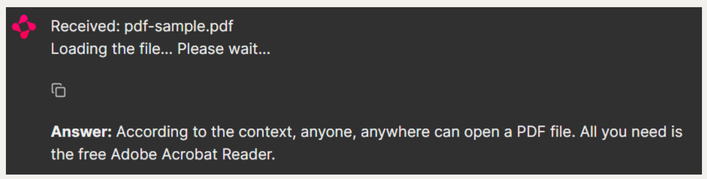
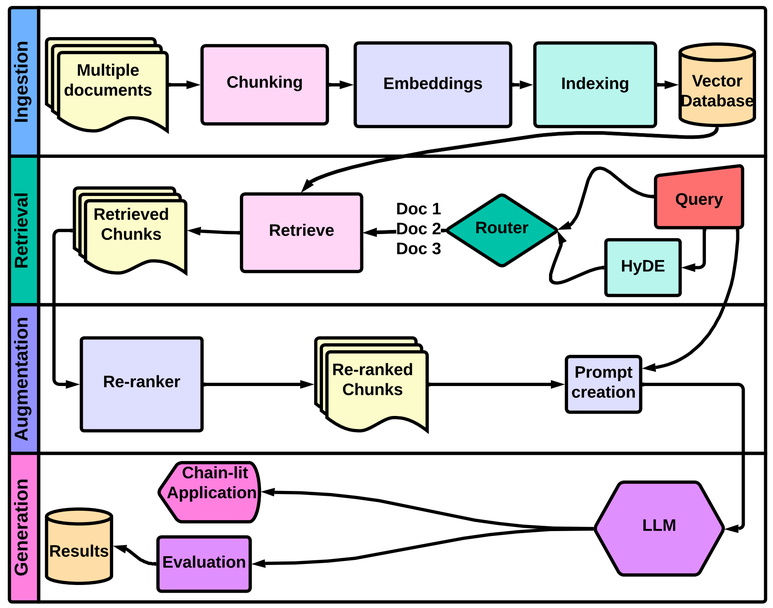


Figure 2. New files processed.

Flow Chart of the pipeline



Features

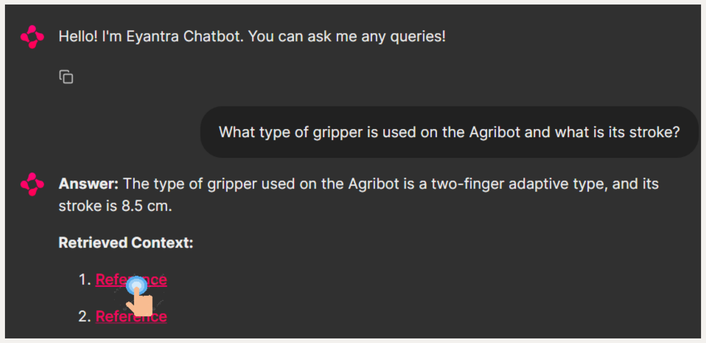


Figure 3. Clicking references.

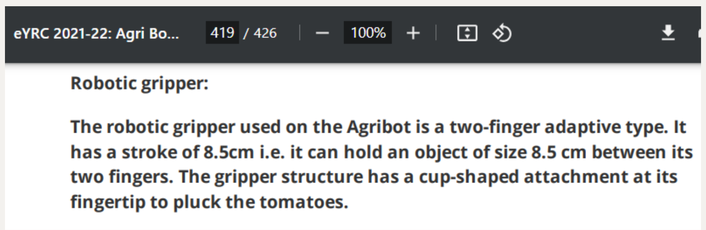


Figure 4. Reference found.

Features

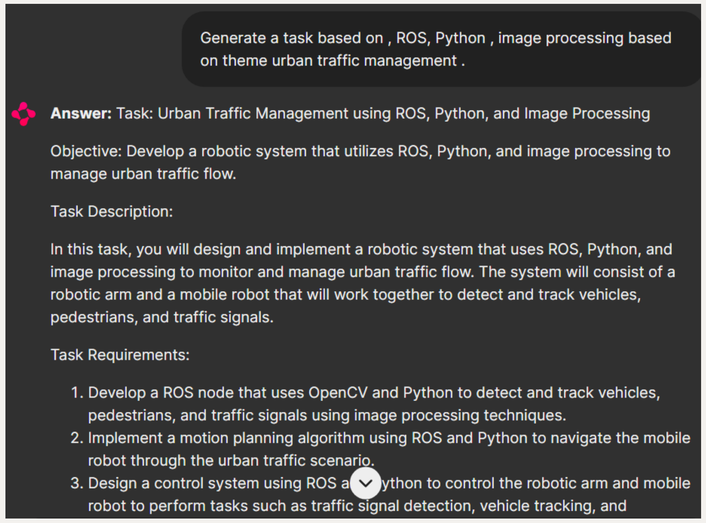


Figure 5. Generating a task based on a tech stack.

Evaluation Results

Method	C. Recall	C. Precision	C. Rel.	A. Rel.	Faith.	A. Corr.	A. Sim.
Recursive Chunking	0.6455	0.5863	0.5918	0.5767	0.5450	0.6026	0.5462
Sentence Chunking	0.3669	0.4540	0.4256	0.3931	0.5644	0.5491	0.4763
Semantic Chunking	0.4000	0.4582	0.4179	0.4591	0.4429	0.4850	0.4825
Sentence Window	0.5612	0.5853	0.6000	0.5638	0.5651	0.5296	0.5766
RAPTOR	0.2640	0.6348	0.4459	0.6401	0.7997	0.8363	0.8116

Table 1. Performance metrics from RAGAS for different methods using a smarter model.

Method	BLEU-1	BLEU-4	ROUGE-L
Recursive Chunking	0.3822	0.1969	0.3982
Sentence Chunking	0.3635	0.1911	0.3877
Semantic Chunking	0.3424	0.1692	0.3632
Sentence Window Chunking	0.3428	0.1653	0.3609
RAPTOR	0.4192	0.2222	0.4381

Table 2. Statistical Performance metrics for different methods using a smarter model.

Future Score

- Integrate with OPENAI for faster and more accurate responses.
- Add multimodal functionality for better user. experience
- Integrate chainlit with Discord for easier interaction.