Project Title:

RAG-based Question-Answering System Development

Submitted by:

Muhammad Uzair
MS-Data Science
29414

Fariha Masroor
MS-Data Science
17639

Submitted to:

Dr. Sajjad Haider

Date of Submission:
[November 10th, 2024]

Platform:
Google Colab with GPU Support

Data Source:
Gulliver's Travels by Jonathan Swift, sourced from Project Gutenberg

GitHub Link:

Git Hub

Python Code:

Google Colab

Table of Contents

1. Platform Details

For this project, Google Colab was used to run all stages of experimentation due to its accessibility to GPU resources, which are required for efficient execution of large language models (LLMs) and embeddings. The Colab environment provided support for the selected models and allowed for optimal processing speeds.

2. Data Details

- **Dataset Source**: The dataset used is an edited version of *Gulliver's Travels* by Jonathan Swift, sourced from Project Gutenberg.
- **Size and Structure**: After preprocessing, the corpus was divided into smaller chunks. The dataset contains one text file, which was split into manageable chunks of 500 characters each, with a 50-character overlap. These chunks were used for context retrieval and answer generation to ensure accurate, contextually relevant answers.

3. Algorithms, Models, and Retrieval Methods

a) Models and Algorithms

- **Embedding Model**: sentence-transformers/all-mpnet-base-v2
    - **Reason for Choice**: This model was selected due to its robust performance on semantic search tasks. It creates dense embeddings optimized for similarity matching, balancing accuracy, and efficiency in context retrieval.
- **Language Model for Answer Generation**: google/flan-t5-large
    - **Reason for Choice**: This model was chosen for its balance of quality and speed. flan-t5-large provides high-quality answers without the extensive computational demands of larger models, making it suitable for Colab's environment while maintaining effectiveness.

b) Retrieval Methods

- **Retrieval Approach**: Semantic Search via FAISS with embeddings from sentence-transformers/all-mpnet-base-v2
    - **Justification**: Semantic search allows for more meaningful retrieval compared to keyword-based methods, especially in a literary text where direct keywords may not capture the nuance needed. FAISS was used to build an efficient vector database, enabling fast similarity searches.

c) Summarization Techniques

- **Summarization**: Initially, we experimented with summarization using the facebook/bart-large-cnn model before passing chunks to the LLM. However, we found that summarization often truncated relevant context details, affecting answer quality.

- **Final Choice**: The final implementation did not use summarization before generation, allowing the LLM to directly process full chunks of text to preserve context.

d) Additional Techniques

- **Text Splitting and Overlap**: RecursiveCharacterTextSplitter was used with a 50-character overlap to ensure each chunk contained enough context from preceding and following sentences. This overlap helped retain coherence, which improved answer generation.

4. Performance Metrics

a) Accuracy and Quality

- **Answer Quality**: The chosen models provided answers with high accuracy for factual questions about *Gulliver's Travels*. Questions related to the main locations, characters, and plot points were answered reliably.

- **Comparison Across Model Variants**:

  - flan-t5-large provided reliable answers in a shorter time compared to larger models like flan-t5-xxl.

  - Variants such as bigscience/T0_3B and Qwen models were also assessed but either returned less relevant answers or required more computational resources.

b) Time Taken

- **Execution Time**: flan-t5-large took approximately 5 minutes for the entire question set on Colab, whereas larger models like flan-t5-xxl took up to 1 hour and 15 minutes. This significant difference in response time was a deciding factor in selecting flan-t5-large for final implementation.

5. Reproducibility

a) Preprocessing Steps

- **Text Splitting**: The text file was split into 500-character chunks with a 50-character overlap. RecursiveCharacterTextSplitter was used to maintain coherence across splits.
- **Embedding Generation**: Each chunk was embedded using sentence-transformers/all-mpnet-base-v2 and stored in FAISS for similarity-based retrieval.

b) System Configuration

- **Colab Environment**: The experiments were run on Google Colab with GPU support to ensure efficient processing.
- **Model Loading**: Models were loaded with device_map="auto" to automatically use available GPU resources, optimizing both retrieval and generation tasks.

c) Model Fine-Tuning and Hyperparameters

- **Parameter Settings**:
  - For retrieval, the k=3 parameter was set in FAISS to retrieve the top 3 similar chunks for each question.
  - Answer generation was limited with max_length=100 and early_stopping=True to optimize response length and prevent excessive generation time.
- **Notes on Model Limitations**: While larger models like flan-t5-xxl could potentially improve answer quality, their computational cost and time requirements in a Colab environment were prohibitive. The chosen setup balances speed and quality effectively for practical application.

Performance

The current implementation using google/flan-t5-large with sentence-transformers/all-mpnet-base-v2 embeddings provides a balance of quality answers and manageable processing time on Google Colab. The retrieval step (FAISS) is very efficient, and the chosen T5 model completes answer generation in a reasonable time for each question. Higher-capacity models like google/flan-t5-xxl were assessed but increased processing time significantly without substantial accuracy gains.

Future Improvements

- **Summarization**: Further experimentation with summarization models could help streamline long answers and improve answer quality.
- **Additional Datasets**: Evaluating the pipeline on a broader set of texts for generalization.

Journey of Iterative Improvement

"We began with an initial attempt using Qwen models and found the answers to be lacking in accuracy. This led us to experiment with various embeddings, such as sentence-transformers/all-mpnet-base-v2 and msmarco-distilbert-base-v4, and progressively move through different LLMs, including google/flan-t5-large and bigscience/T0_3B. Throughout, we adjusted chunk sizes, tried different summarization models, and even considered newer models like Mistral and Llama, aiming to balance quality and speed. The final choice of google/flan-t5-large with sentence-transformers/all-mpnet-base-v2 was informed by both efficiency and answer quality, demonstrating the importance of tuning and systematic testing in RAG pipeline development."

Acknowledgments

Special thanks to Hugging Face for model hosting, Google Colab for providing GPU support, Project Gutenberg for corpus sourcing, and OpenAI's ChatGPT for guidance and assistance throughout the project. This collaboration helped optimize the pipeline and improve answer quality.

References

1. Muhammad Uzair. *RAG - Retrieval Augmented Generation*. GitHub Repository. Available at: https://github.com/muhammaduzair99/RAG
2. Google Colab Notebook. *RAG-based QA System Implementation*. Available at: https://colab.research.google.com/drive/17FLXjtoOQ-FShatfoh_plr8dqYKEE3Cu?usp=sharing
3. Hugging Face Model Repository. *FLAN-T5 Large Model*. Hugging Face. Available at: https://huggingface.co/google/flan-t5-large
4. Project Gutenberg. *Gulliver's Travels by Jonathan Swift*. Available at: https://www.gutenberg.org/ebooks/829
5. OpenAI ChatGPT. *Assistance in Project Development*. Available at: https://chatgpt.com/share/672fcf67-74b0-800a-ac85-08b9cd215238