Retrieval Augmented Generation (RAG) for Generative AI Concepts

April 13, 2024

Erik Sambrailo

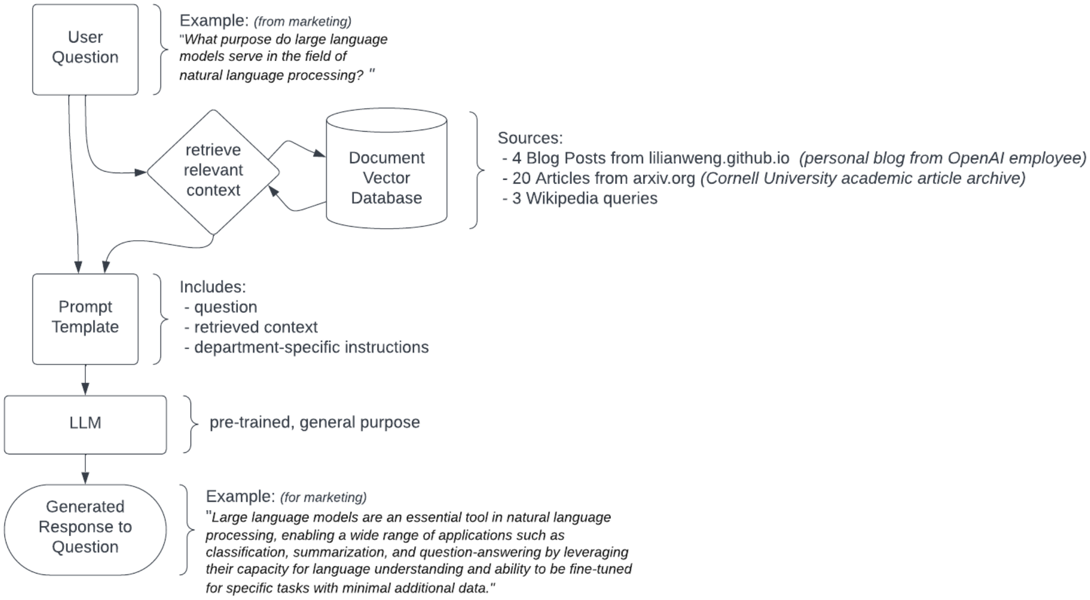
1 Summary

We developed a Retrieval-Augmented Generation (RAG) system specifically tailored for our engineering and marketing departments. This system enhances search and question-answering capabilities related to Generative AI concepts. Through extensive experimentation with various hyperparameters, models, and prompt configurations, we identified optimal configurations for improving the system's effectiveness. The system has shown promising results in handling queries relevant to engineering and marketing. However, further experimentation is necessary to address some identified limitations, such as inaccuracies in responding to factual queries.

2 Introduction

The aim of this project was to develop a proof-of-concept Retrieval-Augmented Generation (RAG) system to enhance our document search and question-answering processes. This system was specifically designed to support queries related to the company's ambition to introduce new Generative AI-based products. It caters to the specific needs of our 300 engineers and our marketing team of 40. To maximize the system's usefulness, we incorporated department-specific instructions so that responses were tailored to the respective audiences. We prioritized the requirements of the engineers for several reasons. Firstly, due to the larger size of the engineering team, we anticipate higher utilization by this group. Secondly, the engineers require more precise and detailed answers to be effective, whereas the marketing team can manage with a broader range of responses due to the higher-level nature of their needs. We discuss how we incorporated this weighting in the evaluation section below.

The system's workflow is illustrated in the chart below. A user submits a question related to Generative AI as text. This question is matched against a vector database containing our documentation to retrieve the most relevant context. The retrieved context, along with the question and additional department-specific instructions, is then fed to a large language model (LLM). The LLM uses this information to compute a response.



Example: *(from marketing)*
"*What purpose do large language models serve in the field of natural language processing?* "

Sources:
- 4 Blog Posts from lilianweng.github.io *(personal blog from OpenAI employee)*
- 20 Articles from arxiv.org *(Cornell University academic article archive)*
- 3 Wikipedia queries

Includes:
- question
- retrieved context
- department-specific instructions

pre-trained, general purpose

Example: *(for marketing)*
"*Large language models are an essential tool in natural language processing, enabling a wide range of applications such as classification, summarization, and question-answering by leveraging their capacity for language understanding and ability to be fine-tuned for specific tasks with minimal additional data.*"

Figure 1: Flow diagram of RAG system

As noted in the diagram, the sources and documents used for this proof-of-concept model are limited. We will further discuss this and the needs of a production system in the limitations section.

3 Methodology

3.1 Technical Approach

As highlighted in the previous diagram, our Retrieval-Augmented Generation (RAG) system comprises multiple components, each capable of being modified or tuned. A single modification within one component can influence the outcomes across the system. Conducting an exhaustive grid search of all potential parameters and modification options would be computationally prohibitive. Therefore, we opted to experiment with specific models, parameters, and prompts that we believed would significantly impact our model. We categorized these into three critical areas, detailed below. Our experimentation process involved stepping through each category incrementally, following the necessary order of their setup. Each step functioned as a mini grid-search to identify the optimal configuration.

Experimental Models, Parameters and Prompts

1) *How documents are stored:*
   - Embedding model
   - Chunking size
   - Chunking overlap
2) *How documents are retrieved:*
   - Retriever search type
   - Number of documents to retrieve
3) *Comprehension of question, context, and generated response:*
   - LLM used

- Composition of generic prompt template
- Fine-tune department-specific composition of prompt template

To streamline the experiments, a master set of functions was defined for building and computing the incremental steps of the RAG system. *See appendix for additional info on functions.*

3.2 Evaluation Metric

We were provided with a dictionary of 75 example questions along with target responses for marketing and engineering departments. Our objective was to compare our generated answers against these target answers. To this end, we devised a composite metric integrating BLEU scores, ROUGE-Lsum scores, and BERTScore F1 scores to accurately assess the accuracy, relevance, and semantic alignment of the generated answers. This evaluation was supplemented by human review. Below, we detail the value of each metric:

- **BLEU** scores measure the lexical accuracy of the generated text compared to a gold standard, emphasizing the preciseness of specific word choices.
- **ROUGE-Lsum** scores evaluate the overlap of longer subsequences between the generated text and the reference texts, indicating the structural coherence at a sentence level and the extent to which the generated text captures the essence of the reference material.
- **BERTScore F1** scores, utilizing contextual embeddings, quantify the semantic similarity between the generated and reference texts, ensuring that the generated answers align contextually with the expected responses.
- **Human Review** despite the robustness of the above metrics, remains essential to assess readability and correctness. We integrated human review with our quantitative metrics at each step of evaluation.

To accommodate the different impacts and requirements across departments, we calculated separate composite scores for the engineering and marketing departments using the following formula:

$$\text{Score}_{\text{dept}} = 0.2 \cdot N_{\text{BLEU}_{\text{dept}}} + 0.3 \cdot N_{\text{ROUGE-Lsum}_{\text{dept}}} + 0.5 \cdot N_{\text{BERTScoreF1}_{\text{dept}}}$$

The weights for each metric reflect their importance in evaluating the RAG system. By prioritizing semantic alignment the highest, followed by structural coherence, and then lexical accuracy, these weights ensure that the evaluation metrics are aligned with the crucial aspects of the RAG system's performance in generating contextually relevant and coherent answers. A total composite score was then calculated by combining the weighted department-specific scores, using the following formula:

$$\text{Total Score} = 0.6 \cdot \text{Score}_{\text{eng}} + 0.4 \cdot \text{Score}_{\text{mk}}$$

While both the engineering and marketing departments are integral to our business, the specific weights reflect their unique contributions and the model's differential impact on each. The higher weight for engineering acknowledges the larger size of the department and the complexity of their technical queries, critical for driving product development and innovation. The weight for marketing reflects the department's strategic focus and its relative size, acknowledging that while important, their queries may not demand the same level of technical detail as engineering queries.

This dual approach of quantitative and qualitative review facilitated a comprehensive analysis of the system's capabilities and aided in identifying optimal model configurations.

3.3 Baseline & Testing

Train, Validation, & Test Sets: To ensure that we did not overfit our experiments to a specific sample set of questions and answers, we divided them into train (10 questions), validation (30 questions), and test sets (35 questions). The smaller size of the train set, while not ideal, was chosen due to the computational intensity of some of our grid searches and limited resources for human review.

Baseline Model: We established a baseline configuration with a generic prompt for comparison:

Parameter	Value
embedding model	"multi-qa-mpnet-base-dot-v1"
chunk_size	128
chunk_overlap	0
retriever search	"similarity"
retr. # to return	4
LLM	cohere

Training Experimentation: The table below outlines the incremental experiments conducted on the train set, detailing the selected parameters and the composite scores achieved for each.

Parameter	Values	Composite (total)
embedding model	'avsolatorio/GIST-Embedding-v0', **'all-MiniLM-L6-v2'**, 'multi-qa-mpnet-base-dot-v1'	0.494
chunk_size, chunk_overlap	(32, 64, 128, **256**, 512), (0, 4, 8, **16**, 32)	0.500
retr. search, # to return, threshold	("similarity", **"mmr"**, "similarity_score_threshold"), (2, 4, **6**, 8, 10), (0.4, 0.5, 0.6)	0.505
LLM	(**cohere**, mistral)	*unchanged*
prompt template	generic improvement*	0.524
prompt template	department specific*	0.529

Alongside the quantitative scoring for each experiment, a human review of the results was conducted. Prior to selecting the LLM, the review focused heavily on the context retrieved from the vector store. Afterward, the review centered on the generated responses.

Validation & Test Sets: With our model nearly finalized, we evaluated it against our validation answer set. The quantitative performance was consistent; however, a brief review of some responses highlighted concerns, particularly with factually specific questions producing inaccurate answers. *See appendix for additional details.*

We conducted one final experiment, reverting the embedding model to 'multi-qa-mpnet-base-dot-v1'. Our research indicated that 'all-MiniLM-L6-v2' sacrificed performance for efficiency. Since it was the first parameter we modified, we were skeptical it was the best choice. Our retest confirmed ‘multi-qa-mpnet-base-dot-v1’ as the superior embedding model, and it also partially corrected some of the inaccuracies observed.

We selected this model, with the revised embedding, as our final model and evaluated it against the held-out test set. Its performance on this set was quantitatively and qualitatively consistent with the previous sets. *Please see the appendix for the finalized model parameters, prompts, and scores.*

4 Results and Findings

4.1 Key Findings, Lessons Learned, Challenges & Limitations

1. **Importance of Document Quality/Quantity**: The critical role of the quality and breadth of our documentation was highlighted during a test where we input target answers into the vector store to see what context was returned. This exercise revealed that many documents lacked the necessary context to achieve the intended answers, emphasizing the need for a comprehensive and well-curated document repository for effective RAG system performance.
2. **Optimal Combination of Chunking and Retrieving**: Experimenting with various chunking and retrieving settings was insightful. Certain settings retrieved substantial ‘references’ noise from documents, requiring iterative adjustments to ensure the retrieval of beneficial context.
3. **Fickle Prompting**: The project highlighted the sensitive nature of prompt formatting, word choice, and sequencing. Minor adjustments could significantly affect the results.
4. **Beyond Quantitative Measures**: While our metrics provided some guidance, they were insufficient on their own. For instance, the model generated factually incorrect answers that were not caught by our metrics due to subtle errors, such as the misplacement of a single word.
5. **Exhaustive Iterations**: Initially, we made one incremental pass through our chosen parameters, selecting the optimal option at each step. A subsequent retest of the embedding model options revealed that our initial choice was suboptimal, emphasizing the complex interplay of the components and the potential benefits of additional iterations if resources permit.

4.2 Conclusion and Recommendations

This proof-of-concept RAG system demonstrates potential as a document search and question-answering solution. Although it achieved its basic objective of retrieving and generating responses from documents, it is far from production-ready and requires substantial development.

4.2.1 Recommendations for Next Steps:

- **Addressing Factually Incorrect Answers**: It is crucial to analyze the causes of incorrect answers and determine if this issue can be mitigated. If not, the model may only be suitable for handling high-level general inquiries.
- **Research on Breadth of Questions and Documentation to Support**: We initially assumed our documents contained the necessary answers. Further analysis is needed to verify this and guide the scope of questions the RAG system can effectively handle. Additionally, implementing structured responses for questions that exceed the documentation’s scope could improve reliability.

- **Department Specific Document Retreival**: If the initial issues are resolved, we recommend exploring department-specific retrieval. This could involve using an LLM to provide the retriever with additional context beyond the query, aiming to align the retrieved content more closely with department-specific needs.

While the RAG system shows promise, significant development and validation work remains to realize its full potential.

Appendix

April 13, 2024

0.1 Parameters and Prompts for Final Model

Below are the specified parameters and prompts for the final selected model.

Parameter	Value
embedding model	"multi-qa-mpnet-base-dot-v1"
chunk_size	256
chunk_overlap	16
retriever search	"mmr"
retr. # to return	6
LLM	cohere

RAG Prompt Template(s) For Engineering

[ ]:

```
eng_rag_template = """[INST]
              Please provide an precise and concise answer to the engineer's␣
 ↪question below based on the context information provided.\n\n
              Below is a context:\n{context}\n
              Below is a question:\n{question}\n
              Below are answer instructions in order of importance:
- Formatting: Provide a succint, single-paragraph answer. Do not use bullet␣
 ↪points. Do not explicitly reference papers in your answer.
- Technical Detail: Include technical details and terminologies that relate to␣
 ↪the question.
- Research Focus: Orient answers towards the research aspects of the questions.
- Objective Tone: Maintain an objective and informative tone, aiming to educate␣
 ↪the reader without persuasive language.
[/INST]
"""
```

For Marketing

[ ]:

```
mk_rag_template = """[INST]
              Please provide a precise and concise answer to the marketer's␣
 ↪question below based on the context provided.\n\n
              Below is a context:\n{context}\n
              Below is a question:\n{question}\n
              Below are answer instructions in order of importance:
- Formatting: Provide a concise, single-paragraph answer that uses the fewest␣
  ↪words necessary to fully address the question. Answer in a single sentence␣
  ↪or phrase if you can. Do not use bullet points. Do not explicitly reference␣
  ↪papers in your answer.
- Succinctness: Make sure your answer is concise and to the point. Provide only␣
  ↪the essential information without delving into the technical depth.
- Broad Overview: Give a broad overview of the topic. Provide only the␣
  ↪essential information without delving into the technical depth.
- Focus on Applications: EFocus on real-world uses and benefits and highlight␣
  ↪how technology can solve problems or create opportunities.
[/INST]
"""
```

0.2 Notebook Reference

The notebook containing the RAG system and all associated experiments has been organized for easy review and reference. Below is the structured outline of the notebook, with notes indicating the location of specific components.

Please Note This notebook includes a subset of the experiments conducted. In many cases, cells were re-run with modifications rather than adding new cells, to reduce the overall size of the notebook and enhance navigation and readability.

0.2.1 Notebook Table of Contents

Certain callout sections have been highlighted.

1) Setup
    - 1.A) Provided Setup
    - 1.B) Validation Question/Answer Set
    - 1.C) My Additional Setup
        - 1.C.1) Installs & Imports
        - 1.C.2) **Define Functions** *This section contains all functions utilized throughout notebook.*
            * Document Functions
            * Building RAG Model Functions
            * Evaluation Functions
2) Loading & Exploring Evaluation Metrics
    - BLEU Score
    - ROUGE
    - BERTScore
    - Human Review
3) Baseline Evaluation
    - 3.A) Rebuilding Baseline Model
    - 3.B) Single Prediction Sample
    - 3.C) Train Set Evaluation
    - 3.D) **Composite Evaluation Metric** *Discussion of evaluation metric.*