Evaluation of Retrieval-Augmented Generation (RAG) Models

Abhishek Shankar

July 24, 2024

Abstract

This report details the methodologies and results of evaluating a Retrieval-Augmented Generation (RAG) model, focusing on the integration of Neo4j graph databases and vector searches. The evaluation framework, Ragas, is utilized to assess the MovieMatch RAG application, offering insights into the model's performance across metrics such as answer relevancy, faithfulness, and latency.

1 Introduction

Retrieval-Augmented Generation (RAG) models enhance the capabilities of language models by integrating external data into the context. This report presents the evaluation of the MovieMatch RAG application, which leverages Neo4j graph databases and vector searches to provide movie recommendations.

2 Evaluation Framework: Ragas

Ragas (RAG Assessment) provides a structured approach to evaluating RAG models by offering a variety of metrics that assess different aspects of the model's performance. The metrics focused on in this evaluation include:

- **Answer Relevancy**: Evaluates the relevance of the generated answers to the user's query.
- **Faithfulness**: Measures the accuracy of the generated answers in relation to the retrieved contexts.
- **Latency**: Measures the response time of the model from input to output.

3 Methodology

The evaluation involves the following steps:

1. **Question**: User's input query.
2. **Context**: Retrieved context from vector search.
3. **Answer**: Generated answer by the LLM.
4. **Ground Truth**: The correct answer, generated by another LLM or human.

3.1 Metrics Calculation

Answer Relevancy is calculated using cosine similarity:

$$\text{Answer Relevancy} = \frac{1}{N} \sum_{i=1}^{N} \cos(E_{gi}, E_o)$$

where E_{gi} is the embedding of the generated answer, and E_o is the embedding of the original question.

Faithfulness measures the accuracy based on the retrieved context:

$$\text{Faithfulness score} = \frac{\text{Number of claims in the generated answer that can be inferred from the given context}}{\text{Total number of claims in the generated answer}}$$

Latency is measured as the time taken from input to output:

$$\text{Latency} = t_1 - t_0$$

where t_0 and t_1 are the start and end times, respectively.

4 Implementation

The implementation involved using the LangChain library, datasets, and the Ragas framework. Below is an example of the code used for the evaluation:

```
from langchain_openai import ChatOpenAI
from datasets import Dataset
from ragas import evaluate
from ragas.run_config import RunConfig
from ragas.metrics import (
    context_precision,
    faithfulness,
    answer_relevancy,
    context_recall,
    answer_correctness,
    answer_similarity,
```

```
    context_entity_recall,
    context_relevancy,
    context_utilization,
)
llm = ChatOpenAI(
    model_name="gpt-4o",
    temperature=0,
    streaming=True,
    max_tokens=1000,
)
eval_dict = {"question": [], "contexts": [], "ground_truth": [], "answer": []}
for row, context, answer in zip(
    dataset,  # this is your initial dataset with question and ground truths
    retrieved_contexts,
    answers,
):
    question = row["question"]
    ground_truth = row["ground_truth"]
    eval_dict["question"].append(question)
    eval_dict["contexts"].append(context)
    eval_dict["ground_truth"].append(ground_truth)
    eval_dict["answer"].append(answer)
dataset = Dataset.from_dict(eval_dict)
score = evaluate(
        dataset,
        metrics=[
            context_precision,
            faithfulness,
            answer_relevancy,
            context_recall,
        ],
        llm=llm,
        run_config=RunConfig(max_retries=2)
    )
print(score)
```

5 Results

The evaluation was conducted on multiple queries. Below are the summarized results:

Question	Faithfulness	Answer Relevancy	Latency (s)
Road Trip Movie	1.00	0.995	7.17
Detective Mystery	1.00	0.973	12.79
Deserted Island Survival	0.93	0.977	6.21
Heist or Robbery	1.00	0.957	5.82
Courtroom Drama	1.00	0.828	3.78
Historical Love Story	0.67	0.862	4.31
Post-Apocalyptic Survival	1.00	0.982	5.54
Sports Team Triumph	1.00	0.841	5.27
AI or Robots	1.00	0.972	5.28
Musician or Band	0.80	0.840	4.12

Table 1: Evaluation Metrics for Different Queries

6 Visualizations

Figures 1, 2, and 3 present the latency, relevancy, and faithfulness metrics, respectively.

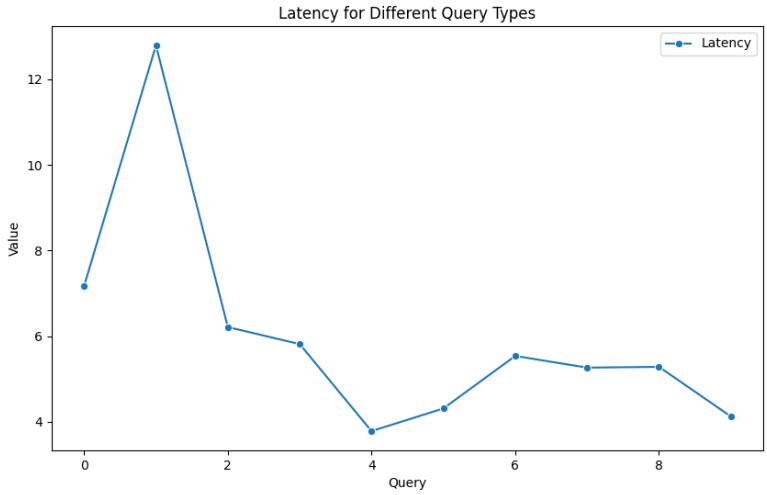

Figure 1: Latency for Different Query Types

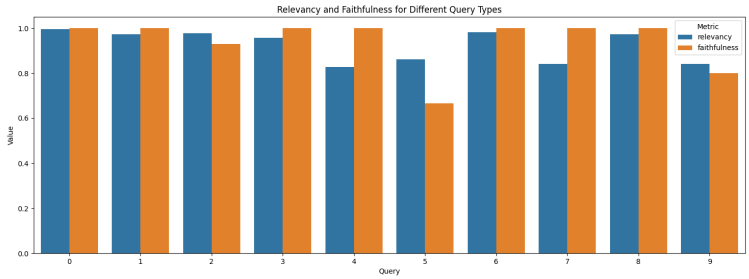

Figure 2: Relevancy and Faithfulness for Different Query Types

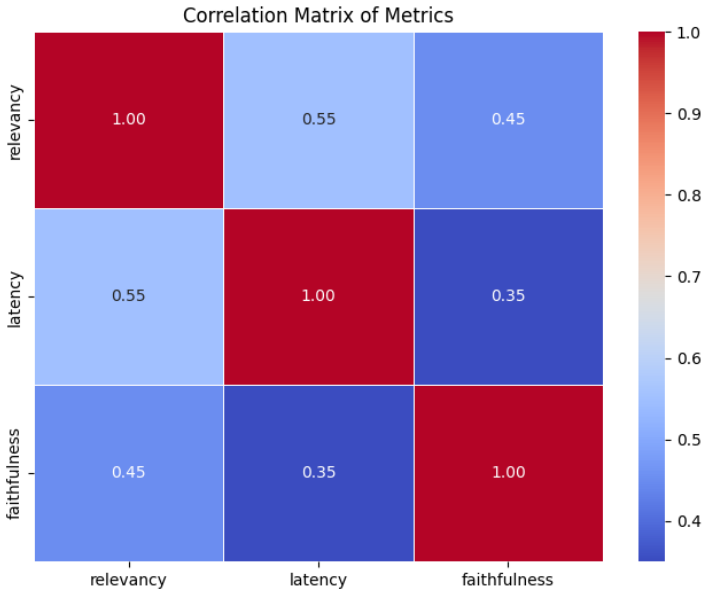

Figure 3: Correlation Matrix of Metrics

7 Conclusions

The evaluation of the MovieMatch RAG application provides the following insights:

- **Overall Effectiveness**: The application shows strong performance in both faithfulness and relevancy, with minor variability in certain queries.
- **Strengths**: High accuracy in specific categories, consistent relevancy scores.
- **Areas for Improvement**: Addressing faithfulness variability and optimizing latency.
- **Future Enhancements**: Fine-tuning for specific query types and continuous performance monitoring.

References

[1] *Evaluating RAG Applications with RAGAS*. Retrieved from Towards Data Science.

[2] *RAGAS Documentation*. Retrieved from RAGAS Documentation.

[3] *Optimizing RAG Applications: A Guide to Methodologies, Metrics, and Evaluation Tools*. Retrieved from Medium.

[4] *HumanFirst Blog on RAG Evaluation*. Retrieved from HumanFirst.

[5] *Neo4j GenAI Ecosystem*. Retrieved from Neo4j.

[6] *YouTube Video on RAG Applications*. Retrieved from YouTube.