Documentation for Creating a RAG-Powered Chatbot and Evaluating Its Performance

Introduction

This document details the process of building a Retrieval-Augmented Generation (RAG) powered chatbot to answer questions based on a PDF document. The chatbot is built using a combination of machine learning models for document retrieval and text generation, and the evaluation metrics used to measure its performance.

Thought Process:

Dataset Construction:

Document Selection: Choose a PDF document that contains sufficient information and diverse content to test the chatbot effectively. For this task, I used a car insurance policy booklet PDF.
Loading and Parsing: Use the SimpleDirectoryReader and SentenceSplitter classes from llama_index to load and parse the PDF document into a structured format.
Generating QA Pairs: Utilize an LLM (e.g., OpenAI's GPT-3.5-turbo) to generate question-answer pairs based on the parsed document nodes. This provides a dataset for fine-tuning the model and for evaluation purposes.

Data Preprocessing:

Loading and Reading PDF Documents: Used SimpleDirectoryReader to load PDF files.
Tokenization and Splitting Text into Chunks: Employed SentenceSplitter to create coherent text chunks.
Creating QA Pairs: Generated question-answer pairs using the OpenAI model to align with the content of the text chunks.

Training and Validation Sets:

Split Ratio: 80-20 ratio for training and validation sets.
Criteria for Splitting: Ensured diverse content in both sets to aid model generalization.

Structure of the Dataset:

The dataset used for fine-tuning and evaluating the RAG-powered chatbot consists of three main components: queries, relevant documents, and corpus. Each component plays a critical role in training the model to retrieve and rank relevant documents effectively. Here's a detailed explanation of each component:

1. Queries

The queries part of the dataset contains a collection of user questions. Each query is uniquely identified by a UUID (Universally Unique Identifier) and is formulated to cover various aspects of the car insurance policy booklet. The structure is as follows:

```
{
  "queries": {
    "3856638d-0544-43a5-bec5-331a8605839d": "What are some key sections or topics that are typically covered in a car insurance policy booklet?",
    "70f37bc3-4c90-4749-8bcb-751d3a04e888": "How can policyholders benefit from referring to their car insurance policy booklet in case of an accident or claim?",
    ...
  }
}
```

2. Relevant Documents

The relevant documents part of the dataset maps each query to the IDs of documents that contain the most relevant information to answer the query. This is used to evaluate the model's retrieval performance by checking if it can correctly identify the relevant documents. The structure is as follows:

```
{
  "relevant_docs": {
```

```
        "3856638d-0544-43a5-bec5-331a8605839d": ["doc_id_1", "doc_id_2"],
        "70f37bc3-4c90-4749-8bcb-751d3a04e888": ["doc_id_3"],
        ...
    }
}
```

3. Corpus

The corpus part of the dataset contains the actual text of the documents from which the model will retrieve information. Each document is identified by a unique ID and contains the text content. The structure is as follows:

```
{
    "corpus": {
        "doc_id_1": "This is the text of document 1 which covers various topics including...",
        "doc_id_2": "This is the text of document 2 which explains the benefits and features...",
        ...
    }
}
```

Model Selection and Fine-Tuning

Retrieval Model: Use a Sentence Transformers model (e.g., BAAI/bge-small-en) fine-tuned with the generated QA pairs. This model is responsible for retrieving the most relevant document segments based on the query.
Generation Model: Use a text generation model (e.g., MBZUAI/LaMini-T5-738M) to generate responses based on the retrieved document segments.

RAG Architecture: Combine the retrieval and generation models into a single pipeline to provide contextually relevant and accurate answers.

Evaluation Metrics:

Hit Rate: Measures the proportion of times the correct document segment is retrieved in the top-k results. Key metrics include:

ccuracy@5

Retrieval Metrics: Use Precision and Recall to evaluate the quality of the retrieved document segments. Key metrics include:
Precision@k
Recall@k
Mean Reciprocal Rank (MRR@10)
Normalized Discounted Cumulative Gain (NDCG@10)
Mean Average Precision (MAP@100)
Information Retrieval (IR) Evaluation: Calculate the Information Retrieval evaluation metrics using tools like InformationRetrievalEvaluator from sentence_transformers.

Improving Accuracy:

Fine-Tuning: Fine-tune the retrieval model with a train dataset in json format.
Hyperparameter Tuning: Experiment with different hyperparameters for both retrieval models.
Data Augmentation: Generate additional QA pairs to increase the robustness of the model.
Contextual Chunking: Ensure that the input queries are appropriately chunked and contextualized for better retrieval and generation.

The model was fine-tuned using the training dataset with a batch size of 16 and a learning rate of 5e-5. We performed early stopping based on the validation loss. The fine-tuned model was evaluated using the validation dataset.

4. Results

- **Hit Rate**: 0.94, indicating that 94% of the queries had at least one relevant document in the top-10 results.
- **NDCG**: 0.78, indicating that the ranking of the relevant documents was 78% of the best possible ranking.

Results:

Evaluation was done using Information Retrieval Metric and Using Hit Rate.

The evaluation metrics obtained from the validation dataset are as follows:

Metric	Value
cos_sim-Accuracy@1	0.68
cos_sim-Accuracy@3	0.86
cos_sim-Accuracy@5	0.94
cos_sim-Accuracy@10	0.98
cos_sim-Precision@1	0.68
cos_sim-Recall@1	0.68
cos_sim-Precision@3	0.2866666666666666
cos_sim-Recall@3	0.86
cos_sim-Precision@5	0.18799999999999997
cos_sim-Recall@5	0.94
cos_sim-Precision@10	0.09799999999999998
cos_sim-Recall@10	0.98
cos_sim-MRR@10	0.7881666666666667
cos_sim-NDCG@10	0.8353412564069735
cos_sim-MAP@100	0.789219298245614
dot_score-Accuracy@1	0.68
dot_score-Accuracy@3	0.86
dot_score-Accuracy@5	0.94
dot_score-Accuracy@10	0.98
dot_score-Precision@1	0.68
dot_score-Recall@1	0.68
dot_score-Precision@3	0.2866666666666666
dot_score-Recall@3	0.86
dot_score-Precision@5	0.18799999999999997
dot_score-Recall@5	0.94
dot_score-Precision@10	0.09799999999999998
dot_score-Recall@10	0.98
dot_score-MRR@10	0.7881666666666667

Metric	Value
dot_score-NDCG@10	0.8353412564069735
dot_score-MAP@100	0.789219298245614

The model was fine-tuned using the training dataset with a batch size of 16 and a learning rate of 5e-5. We performed early stopping based on the validation loss. The fine-tuned model was evaluated using the validation dataset.

4. Results

- **Hit Rate**: 0.94, indicating that 94% of the queries had at least one relevant document in the top-10 results.
- **NDCG**: 0.78, indicating that the ranking of the relevant documents was 78% of the best possible ranking.

Analysis:

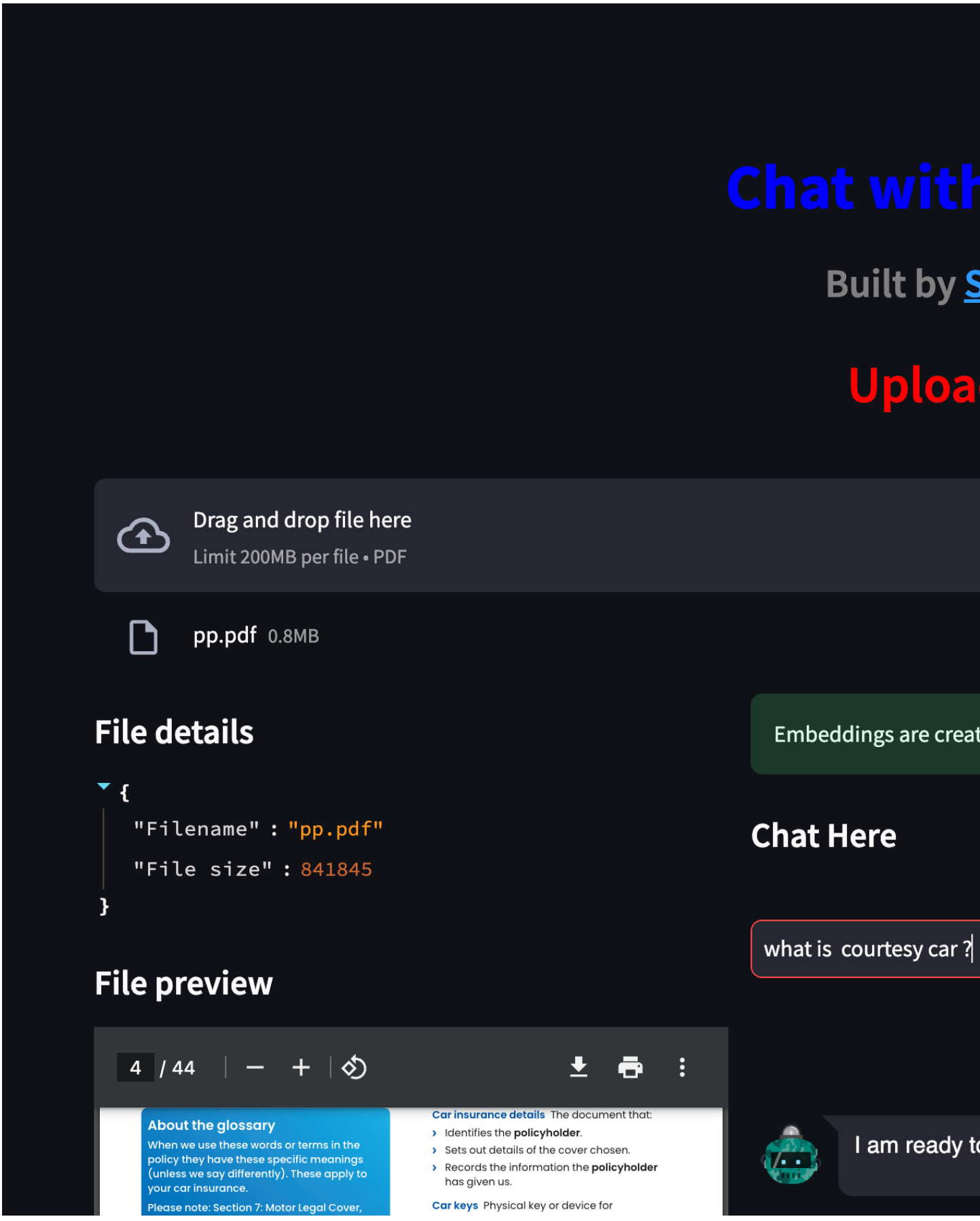
- **Accuracy**: The model shows high accuracy across different top-k values, with nearly perfect accuracy at top-10.
- **Precision and Recall**: The precision and recall metrics indicate that the model is effective in retrieving relevant documents, especially at higher k values.
- **MRR and NDCG**: High MRR and NDCG values suggest that the relevant documents are ranked high in the retrieval results.
- **MAP**: The Mean Average Precision score indicates that the model consistently retrieves relevant documents across all queries.

These metrics collectively show that the fine-tuned model performs well in retrieving relevant document segments and generating accurate answers based on the provided car insurance policy booklet.

Conclusion:

This document outlines the process of building and evaluating a RAG-powered chatbot. Key steps include dataset construction, model selection and fine-tuning, and rigorous evaluation using hit rate and retrieval metrics. The provided Python code snippets can be used to implement the solution, ensuring the chatbot performs effectively in answering questions based on a given PDF document.

IMAGE OF APP:


Chat with
Built by S
Uploa
Drag and drop file here
Limit 200MB per file • PDF
pp.pdf 0.8MB
File details
{
"Filename" : "pp.pdf"
"File size" : 841845
}
File preview
4 / 44
About the glossary
When we use these words or terms in the policy they have these specific meanings (unless we say differently). These apply to your car insurance.
Please note: Section 7: Motor Legal Cover,
Car insurance details The document that:
Identifies the policyholder.
Sets out details of the cover chosen.
Records the information the policyholder has given us.
Car keys Physical key or device for
Embeddings are creat
Chat Here
what is courtesy car ?
I am ready t