Documentation for RAG-Powered Chatbot

Date: 14/06/2024

Author: Pratham Bajpai

1. How I Constructed Our Dataset

To construct the dataset for the chatbot, I followed these steps:

i. **PDF Text Extraction**: I began by extracting the text from the provided PDF document using the PyPDF2 library. This allowed me to access the entire content of the policy booklet.
ii. **Splitting Text into Sentences**: The extracted text was then split into individual sentences to create a pool of potential responses.
iii. **Formulating Queries**: I generated a diverse set of 30 queries that a user might ask regarding motor insurance policies. These queries were designed to cover different aspects such as coverage, claims process, policy renewal, exclusions, and more.
iv. **Creating Query-Response Pairs**: Using the TF-IDF vectorizer and cosine similarity, I matched each query with the most relevant sentence from the text. This resulted in a dataset of 30 query-response pairs.

This dataset ensures that the chatbot can handle a variety of questions related to the policy booklet, providing a solid foundation for initial testing and evaluation.

2. How and Why I Chose These Evaluation Metrics

To evaluate the performance of the chatbot, I chose the following metrics:

i. **Cosine Similarity**: This metric measures the similarity between the expected and predicted responses. It is particularly useful for evaluating text responses as it captures the semantic meaning rather than exact string matches.
ii. **Accuracy**: The proportion of queries for which the chatbot's response was similar enough to the expected response based on the cosine similarity threshold.
iii. **Classification Report**: This includes precision, recall, and F1-score, which provide a more detailed analysis of the chatbot's performance by considering both false positives and false negatives.

These metrics were chosen because they provide a comprehensive evaluation of the chatbot's ability to generate relevant and accurate responses to user queries.

Documentation for RAG-Powered Chatbot

3. What Did I Try to Improve the Accuracy

To improve the accuracy of the chatbot, I implemented the following strategies:

i. **Enhanced Query-Response Matching**: Instead of relying solely on exact matches in the dataset, I incorporated a fallback mechanism that uses a pre-trained language model (DistilBERT) to generate responses when a direct match is not found.
ii. **Fine-Tuning the Similarity Threshold**: By experimenting with different cosine similarity thresholds, I aimed to find the optimal balance that maximizes accuracy without being too lenient or too strict.
iii. **Dataset Refinement**: I ensured that the dataset was diverse and comprehensive, covering a wide range of potential user queries. This helps in improving the chatbot's ability to generalize and provide accurate responses across different topics.

4. Explain Why I Feel This is a Comprehensive Dataset to Gauge the Performance of Our Chatbot

The dataset constructed for this chatbot is comprehensive for several reasons:

i. **Diverse Queries**: The dataset includes a wide variety of queries related to different aspects of motor insurance policies. This ensures that the chatbot is tested on multiple topics and scenarios.
ii. **Relevance and Coverage**: Each query is matched with the most relevant sentence from the policy booklet, ensuring that the responses are accurate and informative.
iii. **Realistic Scenarios**: The queries were designed to reflect real-world questions that users might have about their motor insurance policy, making the evaluation realistic and practical.
iv. **Balanced Dataset**: Care was taken to avoid concentrating the queries on a specific section or page of the document. This helps in evaluating the chatbot's ability to understand and retrieve information from the entire document.