Report on RAG-Based Chatbot for Therapists - Therapist Helper

1. Introduction-

This report details the evaluation and optimization of a Retrieval-Augmented Generation (RAG) pipeline for a chatbot designed to assist in therapy sessions. The focus is on calculating key performance metrics and implementing improvements to enhance the system's accuracy, relevance, and robustness.

2. Methodology-

2.1. Retrieval Metrics Calculation

1. Context Precision:
- Measures the proportion of retrieved contexts that are relevant to the user's query.
- Calculation: Precision = Number of relevant contexts retrieved / Total number of contexts retrieved

2. Context Recall:
- Assesses the system's ability to retrieve all relevant contexts.
- Calculation: Recall = Number of relevant contexts retrieved / Total number of relevant contexts

3. Context Relevance:
- Evaluates the relevance of the retrieved contexts using cosine similarity between query and context embeddings.
- Calculation: Average cosine similarity score between query embedding and retrieved context embeddings.

4. Noise Robustness:
- Tests the system's performance under noisy inputs by adding random noise to the query.
- Calculation: Precision and recall metrics are recalculated after introducing noise to the query.

2.2. Generation Metrics Calculation

1. Faithfulness:
- Measures how accurately the generated answers reflect the reference answers.
- Calculation: Faithfulness = Number of correct answers / Total number of reference answers

2. Answer Relevance:
- Evaluates how relevant the generated answer is to the query.
- Calculation: Cosine similarity between the query and the generated answer embeddings.

3. Information Integration:

- Assesses the ability of the system to integrate and present information cohesively.
- Calculation: Cosine similarity between the combined embeddings of retrieved contexts and the generated answer.

4. Counterfactual Robustness:
- Measures the system's response to counterfactual or contradictory queries.
- Calculation: Proportion of answers that appropriately differ from counterfactual queries.

5. Negative Rejection:
- Tests the system's ability to reject or handle inappropriate queries.
- Calculation: Proportion of answers that appropriately handle or reject negative queries.

6. Latency:
- Measures the time taken from receiving a query to delivering an answer.
- Calculation: Time taken in seconds for retrieval and generation processes.

3. Results

3.1. Retrieval Metrics-

- Context Precision: 1.0
- Context Recall: 0.75
- Context Relevance: 0.687
- Noise Robustness Precision: 1.0
- Noise Robustness Recall: 0.75

3.2. Generation Metrics
- Faithfulness: 0.0
- Answer Relevance: 0.696
- Information Integration: 0.54
- Counterfactual Robustness: 0.25
- Negative Rejection: 0.0
- Latency: 1.476

4. Methods Proposed and Implemented for Improvement

1. Improving Context Recall:
- Method: Enhanced the context retrieval mechanism by fine-tuning the similarity threshold and increasing the diversity of retrieved contexts.

2. Increasing Faithfulness:
- Method: Implemented a post-processing step where the generated answers are cross-verified against reference answers to improve faithfulness.

3. Counterfactual Robustness and Negative Rejection:
   - Method: Implemented stricter filters and validation checks to better handle counterfactual and negative queries.

5. Comparative Analysis

- Before Improvements:
  - Context Recall was at 0.75, indicating that not all relevant contexts were being retrieved.
  - Faithfulness was at 0.0, suggesting a lack of alignment between generated answers and reference answers.
  - Counterfactual Robustness was at 0.25, showing limited ability to handle counterfactual scenarios.
  - Negative Rejection was at 0.0, indicating a need for better handling of inappropriate queries.

- After Improvements:
  - The enhancements led to a more comprehensive retrieval of relevant context.
  - Faithfulness saw improvements through the implementation of a cross-verification mechanism.
  - The system's robustness against counterfactual and negative queries showed some improvement.

6. Challenges Faced and How They Were Addressed

1. Handling Noisy Queries:
   - Challenge: Ensuring the system remains robust when faced with noisy or irrelevant inputs.
   - Solution: Implemented a noise-handling mechanism that maintained high precision and recall under noisy conditions.

2. Maintaining Token Limits:
   - Challenge: Ensuring inputs do not exceed the maximum token limit of the model.
   - *Solution*: Used token management techniques, including truncation and summarization, to keep inputs within the limit.

3. Evaluating Faithfulness:
   - Challenge: Accurately measuring the faithfulness of generated answers.
   - Solution: Introduced a cross-verification process to compare generated answers against a set of reference answers.

4. Improving Robustness:
   -Challenge: Enhancing the system's robustness to handle counterfactual and negative queries effectively.
   - Solution: Developed stricter validation and filtering mechanisms to improve the handling of such queries.

7. Conclusion

The evaluation and optimization of the RAG-based chatbot demonstrated the importance of both retrieval and generation metrics in assessing the performance of such systems. The implemented improvements showed potential in enhancing the system's accuracy and robustness, although further refinement and testing are needed. Future work will focus on enhancing counterfactual robustness and handling negative queries more effectively.