NLP Project Presentation: Implementing a RAG System

Overview of the RAG System

- **Definition**: A Retrieval-Augmented Generation (RAG) system combines retrieval-based models and generation-based models to enhance the performance of NLP tasks. The system first retrieves relevant documents from a corpus and then uses a language model to generate responses based on both the query and the retrieved documents. This method significantly improves the accuracy and contextual relevance of the responses.
- **Process**:
  - First retrieves relevant documents from a corpus.
  - Then uses a language model to generate responses based on the query and retrieved documents.
- **Benefits**: Improves accuracy and contextual relevance of responses.

Preprocessing

- **Dataset Loading**: Used the `eli5_category` dataset, combining training, validation, and test subsets.
- **Text Processing**:
  - Split list of answears into separate elements.
  - Added a sentencizer to segment text into sentences.
- **Cleaning**: Filter out sentences that are too short (based on the percetiles).

Encoding and Retrieval

- **Embedding Models**: Used `all-mpnet-base-v2` to create text embeddings.
- **Retrieval Mechanism**: Employed dot similarity to find relevant documents based on query. In our case embedding model returns normalised embeddings so dot porduct and cosine similarity gives the same result thus will be faster.
- **Top-K Selection**: Selected the top-K most relevant documents for the next stage.

Generative Model

- **Model Selection**: Chose the "google/gemma-7b-it" model.
- **Tokenization**: Preprocessed text using `AutoTokenizer` from Hugging Face.
- **Model Loading and Configuration**:
  - Instantiated the model with CPU-friendly settings.
  - Set the model to use CPU.
- **Size**: We prepared function to calculate size of the model (32GB)

Prompt Design:

- **Base Prompt Creation**: Designed a base prompt with guidelines for the model to extract relevant passages and provide explanatory answers.
- **Template Formatting**: Formatted the context and query into a predefined template for consistency and clarity.
- **Instruction-Tuned Model**: Wrapped the prompt in a dialogue template suitable for instruction-tuned models, specifying user roles and preparing the prompt for generation.
- **Inclusion of Examples**: Included examples of the desired answer style to guide the model in producing high-quality responses.
- **Context Integration**: Augmented the query with text-based context from the top-K retrieved documents, joining context items into a coherent paragraph.

Connecting Everything Together

Results and Observations

- **Enhanced Relevance**: Effectively retrieved and utilized relevant information.
- **Improved Contextual Understanding**: Better understanding and response to complex queries.

Conclusion

- **Summary**: Implementation of a RAG system significantly advances NLP applications.

- **Key Benefits**: Provides accurate, relevant, and context-aware responses.
- **Future Improvements and Fields to Develop**:
    1. **Model Fine-Tuning**: Continuously fine-tune the generative model on specific datasets to improve its accuracy and relevance for particular domains or tasks.
    2. **Enhanced Retrieval Mechanisms**: Develop more sophisticated retrieval algorithms, possibly incorporating user feedback to improve the relevance of retrieved documents.
    3. **Context Management**: Implement advanced context management techniques to handle longer documents and maintain coherence in generated responses over extended conversations.
    4. **Scalability and Performance Optimization**: Optimize the system to handle larger datasets and more complex queries efficiently, ensuring scalability for real-world applications.
    5. **Multilingual Support**: Extend the system to support multiple languages, allowing for a broader range of applications and user interactions.
    6. **User Interface Improvements**: Develop more intuitive and user-friendly interfaces for interacting with the RAG system, enhancing usability and accessibility.
    7. **Integration with Knowledge Bases**: Integrate the system with structured knowledge bases or databases to provide more factual and detailed responses.
    8. **Explainability and Transparency**: Enhance the explainability of the model's responses, allowing users to understand how and why certain answers were generated.
    9. **Security and Privacy**: Implement robust security measures to protect user data and ensure privacy, especially in sensitive applications.
    10. **Real-Time Processing**: Improve the system's capability to process and generate responses in real-time, enabling applications in live settings such as customer support or virtual assistants.

Closing

- Thank you for your attention.