Erdos Data Science Bootcamp Spring 2024 Executive Summary

Aware Team 3: Craig Franze, Baian Liu, Mohammad Nooranidoost, Himanshu Raj, Anil Tokmak, and Peter Williams
Github: https://github.com/peter-mm-williams/aware-nlp

Overview:

Retrieval Augmented Generation (RAG) is a powerful approach that enhances Large Language Models (LLM) capability to generate a richer and more in-context response to user queries. By retrieving relevant information through an information retrieval system, and then generating responses, RAG ensures reliability and minimizes the risk of misinformation and hallucination.

Objective:

- Build an information retrieval system that has the following key components:
  - **Vector indexing:** The text contents will be converted to high-dimensional vectors using sentence embedding models.
  - **Storing in a vector store:** Embedded vectors are loaded to a vector database
  - **Retrieval based on similarity match:** The similarity between the query and the content vectors will be calculated based on the distance between the vectors
- Major priorities of this project:
  - Device a methodology to gauge the performance of the retrieval
  - Sub-second retrieval process

Evaluation Methodology:

Dataset Pre-processing: To evaluate these pipelines, the Best Buy Worker subreddit was pre-processed using the following steps:

- Statements were composed by concatenating the title and text fields of the individual submissions and comments.
- Documents were then split into 512 token vectors with 50 token overlaps producing 5,667 documents.
- Statements were encoded into an embedding space via a choice of embedding model.
- Three questions typical to the type of questions Aware's clients would ask of the data were handwritten:
  1. What do Best Buy employees think of the company?
  2. What are the most common reasons for employees to leave Best Buy?
  3. Do employees feel understaffed?
- Labeled Datasets were prepared by:
  - Sampling the documents
  - Labeling the documents as relevant or irrelevant by either a group of human observers or a large language model (LLM)

Automated Labeling: For the construction of larger evaluation sets, LLMs were used in preparing labeled data.

- Data was labeled using either the "dolphin-mixtral" and "llama3" models.
- Quality of labeling was judged against on data labeled by 7 independent observers
  - Correctly labeled 10 out of the 12 statements unanimously labeled as relevant
  - Using a consensus threshold of 50% of human labelers produced an F1 score of 0.80

Evaluation: Methodologies were evaluated quantitatively based on the precision, recall and F1 scores of retrieved documents.

Results and Advanced Methods

Embedding Models:

- A 90 statement dataset was used to evaluate the performance of naive retrieval for a range of embedding models.
- "all-mpnet-base-v1" was shown to perform well for both as little as 5 retrieved documents and as many as 30 (f1 scores of 0.47 ± 0.27 and 0.55 ± 0.13, respectively).

Clustering:

- Exploratory data analysis was performed on a set of 650 LLM labeled statements signifying positive sentiment, negative sentiment or neither by analyzing distributions of cosine-similarity to queries and projections to lower-dimensional spaces.
- Clustering methods and hyperparameters were evaluated via completeness, homogeneity, v-score, and the number of clusters.
- A k-means clustering with 500 clusters offered a good tradeoff between the number of clusters and performance.
- Clusters were used in a 2-stage retrieval process by which clusters would be searched for relevant documents in order of their similarity of their centroid to the query in the embedded space.
- Retrieved documents with an F1 score at or better than naive retrieval for 5, 10, 15, 20, 25, and 30 retrieved documents.

Multi-query:

- Technique where an input query is sent into a large language model to generate five different variations of the user query.
- For every LLM generated query, we then repeat the baseline procedure and pick the unique top 20 documents.
- Evaluated on a larger dataset consisting of 298 reddit submissions and posts labeled by "Llama3-70B".
- Data was split with a chunk size of 300 and overlap of 50, using openAI embeddings, we indexed them into ChromaDB.
- A "Mixtral-8x7b" LLM with a temperature setting of 0 was used to generate five different queries.
- The unique top 20 documents for every original query were retrieved.
- First 5, 10, 15 and 20 retrievals generated from the multi-query approach yielded F1 scores 0.39 , 0.64, 0.77, 0.73.
- Baseline ("naive") retrieval scored 0.35, 0.605, 0.74, 0.75, respectively.

Multi-vector Indexing:

- Technique where given context docs are summarized using a large language model.
- Assign a unique id to every summarized document in order to map it to the original document.
- Summarized docs are then indexed into the vector store.
- The user's query is matched against the summarized documents, the top retrieved documents are then identified with the original document which are finally returned as relevant.
- Used the "Mixtral-8x7b" LLM to generate document summaries.
- Evaluated this method using the "Llama3-70B" labeled dataset. For the first 5, 10, 15 and 20 retrievals, we found that the multi-vector indexing approach gives F1 scores 0.46, 0.71, 0.77, 0.76 whereas the baseline approach gives 0.39, 0.60, 0.71, 0.75.

Conclusions and Future Directions:

Conclusions:

- Created a procedure for parsing, chunking, and loading reddit data into vector stores.
- Retrieval on these indexed documents was evaluated for a range of embedding models and retrieval pipelines.
- Clustering, multi-querying, and multi-vector indexing all showed improvements over the naive process.
- Clustering and multi-vector require additional pre-processing that should be considered as a trade-off prior to being implemented at a large scale.

Future Directions:

- Additional evaluation would be aided by (1) a more extensively labeled evaluation dataset spanning a majority of the subreddit, as well as other subreddits (2) making use of other retrieval evaluation metrics such as mean reciprocal rank and normalized discounted cumulative gain.
- Future work on this project could investigate improvements by using hypothetical document embeddings to sample a broader range of the embedded space, searching metadata (self-querying) to make use of timestamps and a sentiment metric generated from the statement.
- Query time of information retrieval systems using these varying methodologies should be loaded with a broad set of subreddit data and evaluated for retrieval time.
- We would like to explore the performance impact of making use of metrics based on the frequency and average length of posts by a given author and the length of the thread from which the statement is sourced.