ANLP Assignment 2 Report

Akshita Gupta[1], Krishnaprasad Vijayshankar[1], Mahita Kandala[1]

[1]Carnegie Mellon University,

This is the report for Assignment 2, as a part of 11-711 Advanced Natural Language Processing. The code for this has been uploaded to Github[1] and the submission made on Canvas.

1 Data Creation

1.1 Compilation of Knowledge Resource

The data was categorized into three parts, each assigned to a team member:

General information about Carnegie Mellon University (CMU) and Pittsburgh, events and sports teams related to CMU and Pittsburgh, Music and cultural events associated with CMU and Pittsburgh.

The starting point for each of these subcategories was the set of links provided in the assignment. Additional data was gathered through web searches on the relevant topics. Documents that could potentially skew the model were excluded, such as websites with sub-links that did not contain pertinent information (e.g., advertisements) or sources with sparse and repetitive content that would not be useful when scraped.

1.2 Extraction of Raw Data

The primary tools used for extracting data from websites was beautifulsoup and pdfminer.

However many websites were dynamic and hence directly scraping them rendered no results. In such cases, the extraction was automated using Selenium (for example: to scroll the pages completely) and then extracted element by element (for example: in case of events dates and timings) to achieve complete and cohesive data.

The data extracted was then cleaned and stored in text files according to the topic. This includes removal of extra white spaces, escape characters, non-unicode characters, citations, and hyperlinks. This was done for ease of comprehension, clarity and further processing.

1.3 Train and Test Data

The data in the text files was then analyzed for relevant information and any data that might potentially mislead the model was cleaned out. The annotation process was divided into two parts: manual annotation and annotation through large language models (LLMs).

For manual annotation, the team reviewed the topics and contents present in each text file, creating questions based on factual information found on the internet about each topic. This involved generating a few questions for each text file to provide a generalized representation across the various domains present in the data. Most of these questions were allocated to the test dataset, while a few were used as references during the process of generating training data through LLMs.

1.4 Annotation Interface

The manual annotations were stored in a CSV file, for the ease of manipulation and access for later use. The data was also stored in JSON format for LLM fine tuning attempts.

The automated data was generated through Ollama by splitting the scraped data into chunks of information and asking the model to generate relevant questions by providing it with a few manually annotated examples as context.

1.5 Data Quality

The first step for assessing the quality of automatically annotated data was through human evaluation. Some of the questions generated were removed due to unexpected or poor answers (question and answers did not match, answers did not match, or the answers were present in the questions asked)

For automatically annotated dataset, the uniformity across model generated answers was main-

[1]https://github.com/aks-gupta/ANLP-End-to-End-RAG-System

tained through prompting with specific instructions on how the answer should be formatted and formulated.

Additionally, Inter Annotator Agreement(IAA) on a subset of test questions. The average similarity and agreement was calculated for this subset. The final IAA score on this subset was 0.86

2 Model Details

As a part of checking how baseline models may perform on the data, a few models were run without any data finetuning or context, which returned extremely poor results, justifiably so since most of the information tested was not present in the context. Some of the baselines tested were:

2.1 Baseline Models

2.1.1 GPT-2

GPT-2 (Radford et al., 2019) returned extremely poor results even post finetuning, due to the extremely small size of the model. The model returned semi-decent answers when trained with a smaller corpus of information, but on training with a large number of question-answer pairs, despite providing it context, it failed returning the right answer on elementary questions - such as "What are Pittsburgh Steelers colours?"

2.1.2 RoBERTa

Given how RoBERTa (Liu et al., 2019) is specifically aimed at question-answering tasks, the model performed much better compared to GPT-2, however it still failed on questions that may require a certain level of contextualization before answering - such as "When is Alan Walker coming to Pittsburgh?"

2.1.3 Llama

On receiving below average results on smaller, fine-tuned models - the focus was shifted to employing larger, more sophisticated models. For this, Ollama was chosen as the primary platform to run bigger models, such as Llama.

Llama 2 (et al., 2023) - The first model tried on Ollama was two variations of Llama 2 - one trained on question-answer pairs, and one trained on raw context. While training on question-answer pairs might've taught the model how to frame and return the answers better, it is quite tough to ensure no context is skipped, and maintain the quality of answers while generating question-answer pairs, even with automatic annotation. Seeing Llama 2 was able to contexualize on raw data with competitive output, question-answer pair approach was abandoned for testing on Llama 3.2.

Llama 3.2 (et al., 2024) - The final model tested which gave the most accurate results for the purpose of this particular dataset happened to be Llama 3.2. Llama 3.2 outperformed all other baselines with simple indexing, and gave much better results on improving areas such as indexing and retrieval, expanded on in the next section.

2.2 Workflow

The refined workflow aimed at maximizing performance through efficient data processing, embedding generation, and result evaluation. An overview can be found in 1

Data Collection - Aggregated data from multiple text files covering relevant topics.

Data Splitting - The collected data was split into manageable chunks of a specified size to ensure optimal processing. This was determined by trying the model's accuracy with 2000, 1000, and 500 character length while recursive splitting.

Embedding Creation - Generated embeddings for each chunk of data and stored them in a vector database using FAISS (Jegou et al., 2018). This process enabled efficient retrieval of relevant information.

Retrieval of Relevant Data - For each query, the system retrieved the top 'k' chunks from the vector database. The value of 'k' was fine-tuned based on performance outcomes.

Re-ranking Context - To improve the accuracy further, the context obtained used FAISS was then re-ranked to make sure the most relevant one is used for text generation.

BM25 - BM25 (Robertson and Zaragoza, 2009) was applied to combine semantic similarity with term-based relevance and improve retrieval.

Query Expansion - Query Expansion (Nogueira et al., 2020) was used to generate multiple variations of the input query, enhancing retrieval coverage and addressing potential vocabulary mismatches.

Improved Retrieval - The final improved retrieval ensembled the FAISS(Dense) and BM25(Sparse) retrieval methods to product the context for a query.

Answer Generation Using Zero-Shot Prompting - Employed Zero-Shot prompting with the Llama 3.2 model on Ollama to generate answers based on the retrieved data.

Result Review - Results from the model were care-

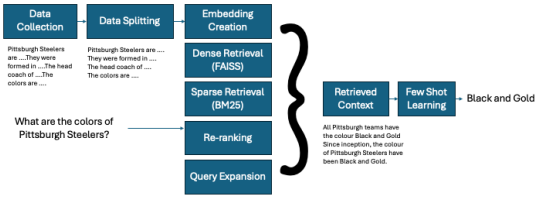

Figure 1: Workflow Diagram

fully reviewed to ensure relevance and accuracy.

Streamlined Workflow with LangChain - Implemented LangChain as a wrapper to streamline the entire process, enabling smooth interaction between components.

Few-Shot Learning - Introduced few-shot learning (Izacard et al., 2022) by selecting 10 representative examples across different topics. These examples were used to prompt the model, improving its ability to generate accurate responses.

3 Results

3.1 Exact Match (EM)

Measures how often the predicted answer exactly matches the ground truth answer.

$$ExactMatch = \frac{NumberofExactMatch}{TotalNumberofQuestions} * 100$$

3.2 F1 Score

The F1 Score is the harmonic mean of precision and recall.

$$F1Score = 2 * \frac{Precisio*Recall}{Precision+Recall}$$

where precision is the number of correct words in predicted answer compared to the length of the answer, and recall is the number of correct words in predicted answer compared to number of words in ground truth answer.

The metrics obtained post testing on different models is mentioned in Table 1. The results for GPT-2 have been ignored, due to them being statistically insignificant.

Model	EM	F1	Recall
RoBERTa	0.10	0.21	0.15
Llama2	0.03	0.23	0.28
Llama3	0.13	0.31	0.26
Llama3 enhanced	0.32	0.52	0.53

Table 1: Performance Metrics of Different Models

4 Analysis

Based on the above results, as well as manual inspection, there were a few points that struck out.

4.1 Size of Data

The performance of models varies greatly with the amount of data that is fed into the model. While some models performed moderately okay when tried on individual systems, and with individual text files for domain specific questions, on combining multiple text files, the model seemed to retrieve slightly unrelated context from various files.

4.2 Question type

All the models tested definitely showed much better performance on questions that directly tested fact-of-matter knowledge as opposed to questions that may require a slight inference to be derived from the context, such as "When is the first Penguin's match against the Braves?"

4.3 Topics

Tying in with the previous observation, certain topics contain more event and inference based text, which leads to relatively poor performance in those domain areas. The performance of history and information pages was comparatively much better than events and schedules pages.

5 Conclusion

After comparing various pre-trained models both with and without fine-tuning and different retrieval models, Llama3 with the improvised retrieval engine with few shot prompting gave the best results with an F1 score of 0.52, exact match score of 0.32 and answer recall of 0.53.

Acknowledgments

References

Abhimanyu Dubey et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Hugo Touvron et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Atlas: Few-shot learning with retrieval augmented language models. *Preprint*, arXiv:2208.03299.

H. Jegou, M. Douze, and J Johnson. 2018. Faiss: A library for efficient similarity search, engineering at meta. https://engineering.fb.com/2017/03/29/data-infrastructure/faiss-a-library-for-efficient-similarity-search/. [Accessed 23-10-2024].

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2020. Document expansion by query prediction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*. [Accessed 23-10-2024].