Final Project Report

Chat-Driven Document Search Engine

Anthony Rautmann, Jacob Strasler
Electrical Engineering and Computer Science, Milwaukee School of Engineering

CSC 6621 Applied Machine Learning

April 24, 2024

GitHub

Introduction

Large language models (LLMs) such as ChatGPT have changed how people interact with artificial intelligence. Before, people likely were not interacting with artificial intelligence models in the direct, often conversational format that they are today. This use of artificial intelligence has identified a use case for businesses: getting questions about your company answered without involving a human. This frees up human labor capital to focus on more productive tasks while allowing users to get a quicker, and ideally more accurate, response.

Slowing companies from implementing this type of interface is the time and cost expenses involved in training an LLM. However, smaller LLMs can be "trained" in a more focused fashion on a small collection of documents. In the case of this project, two LLMs are "trained", or used, with a RAG approach: one with the course syllabus and one with the National Football League rulebook. A user can then ask questions to the LLM specific to the contextual domain of the document it has been trained on, providing a convenient way to get information quickly.

Dataset

The dataset for this project is simply two PDF documents: the course syllabus and the National Football League (NFL) rulebook. The course syllabus was downloaded from the course Canvas page. The NFL rulebook is downloaded from the NFL website. An important attribute of this project was that a PDF can be used as-is and plugged in to the pipeline. These documents can be considered as the "training" data. Evaluation data consisted of hand-made sets of questions and their expected responses, one set per document.

Model Implementation

Processing for LLM

The backbone of using the LLMs, and what really was the bulk of the work for this project, was processing the data from the PDFs. This process involved loading the documents, cleaning them, creating chunks of the document, embedding these chunks in a vector store, and loading into memory the FAISS index for search and retrieval of chunks for LLM context. LangChain libraries were important in these steps.

The process designed in this project works for one document at a time. Processing the data from the PDFs consists of splitting the document into smaller chunks in order to fit into a model's context window. The chunk size and overlap is something that was experimented on, with the effectiveness of different values being taken into consideration. A recursive text splitter removed extraneous newline and space

characters when splitting. The documents created by splitting are referred to as "chunks".

Chunks are then embedded into a vector store – in this case, FAISS, with the help of a sentence transformer model (allMiniLM from HuggingFace). This vector store is saved locally so as to not have to rebuild it every time a new prompt was given, LLM used, etc. There is a one-to-one relationship between the number of documents and the number of vector stores. The vector store is consequently used in a QA chain with the LLM.

Evaluation Techniques

This project does not lend itself to "classical" evaluation metrics commonly used in classification (precision, recall, receiver operating characteristic area under the curve) and regression (accuracy, mean squared error) problems. Instead, similarity scores were calculated for the response from a prompt given by the model and the expected response. The prompts are from the aforementioned evaluation set. The similarity score was calculated using cosine similarity between the model's response and the correct answer provided in the evaluation set. A similarity score of 1.0 indicates perfect similarity (identicalness), 0.0 no similarity, and -1.0 indicating perfect dissimilarity (oppositeness). The similarity score was calculated using the Sentence Transformers Python library with a Bidirectional Encoder Representations from Transformers (BERT) model.

Baseline LLM

For this project, the LLMs used are from Hugging Face. The initial and baseline model is GPT-2 from OpenAI Community. This model was originally released in 2019 and is the smallest variant of GPT-2 with 124 million parameters. Results from this model were disappointing: the similarity score was just 0.312. To help further frame this score, the raw context from the vector store that is fed into the LLM scores 0.430. As seen in Example 1, the response is neither correct nor intelligible.

Example 1 Chat and response regarding the NFL rulebook using the baseline model, GPT-2. The actual answer should be "8 inches high and 4 inches wide." The similarity score for this response is 0.026.

Q:	What is the minimum size of numerals on the uniforms?
A:	1-1-3 • The Commissioner in the manner indicated may order all jerseys and shirts to be numbered when they are ready for the game in time to make their league appearance at the earliest opportunity. • The uniform manager shall be responsible for the order of uniforms.

> At the interim or final game of the regular season or the final preseason game, he shall make an order for the jerseys and shirts to be numbered when ready."

Techniques for Improvement

The primary method of improving the responses was evaluating different LLM models. The models evaluated, in addition to the GPT-2 model discussed in Baseline LLM, are

- **GPT-2 Large**: developed by OpenAI, published in 2019, and has 774 million parameters.
- **DistilGPT2**: developed by Hugging Face, is a distilled version of GPT-2 (making it smaller and easier to run than GPT-2), published in 2019, and has 82 million parameters.
- **Phi-2**: developed by Microsoft, published in 2023, and has 2.7 billion parameters.
- **Phi-3 Mini**: developed by Microsoft, is a lightweight version of the model trained with the Phi-3 datasets, published in 2024, and has 3.8 billion parameters. The variant used in this project is that which supports context length of 128,000.

Changing LLM models had the most significant impact on the performance. Other changes included different chunk sizes, chunk overlap, and embedding models. The final values used can be seen in Table 1.

Table 1 Values used for processing and embedding parameters.

Parameter	Value
Chunk size	500
Chunk overlap	200
Embedding model	Hugging Face all-MiniLM-L6-v2

Improved LLM

The model that performed best on our NFL dataset was the Microsoft Phi-2. It had a mean similarity score of 0.570, an 82.6% improvement over the baseline GPT-2 model. This can likely be attributed to Phi-2 having a significant amount more parameters than GPT-2 and its recency, being published in 2023, as there has seemingly been in explosion of work in this area since ChatGPT was announced (which uses GPT-3.5, a newer iteration than GPT-2.) As seen in Example 2, the response is exactly right – the expected answer is returned in a succinct, easy-to-read fashion with accurate information.

Example 2 Chat and response regarding the NFL rulebook using the improved model, Phi-2. The actual answer should be "8 inches high and 4 inches wide." The similarity score for this response is 0.736.

Q:	What is the minimum size of numerals on the uniforms?
A:	The minimum size of numerals on the uniforms is 8 inches high and 4 inches wide.

LLM Comparison

Comparisons were made across the five LLMs discussed above as well as the context fed into the LLMs from the context search. In order to make these comparisons, the same 79 questions about NFL rules were asked to each model (with the same vector store being pulled from for each model). As described in the Evaluation section, a similarity score for the model's response is then calculated based on a provided answer. Figure 1 shows the results for each model and the context.

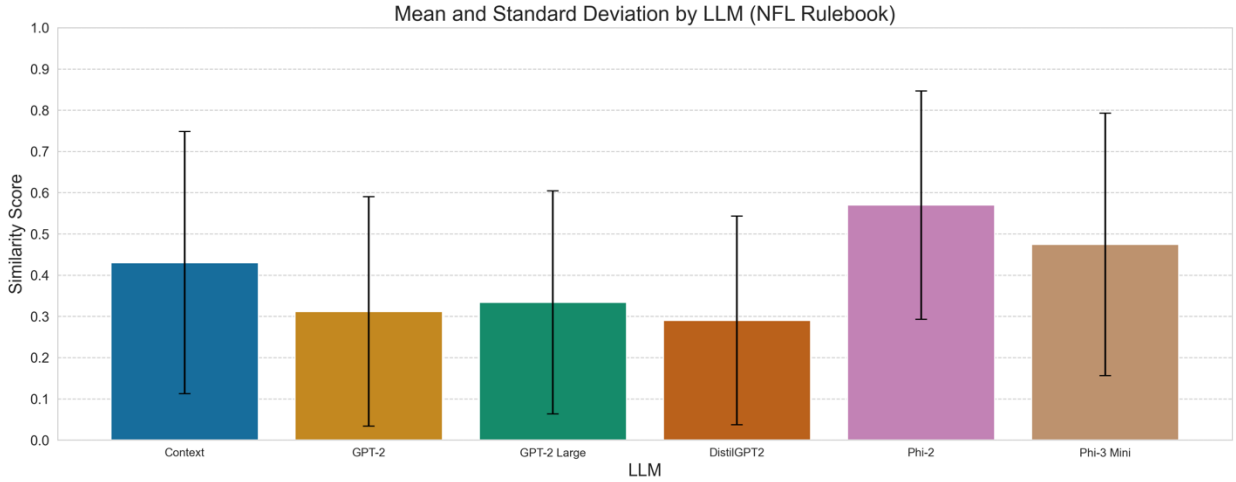

Figure 1 Results of LLM comparison experiment. Raw data for Figure 1 can be seen in Table 2 in Appendix.

As seen in Figure 1, the Phi-2 model is outperforming the other models, both on average and at best. The GPT-2 models struggle compare to the Phi models, with the best GPT-2 model being GPT-2 Large. There seems to be a correlation in these results with parameters: the GPT-2 models have parameters on the order of millions whereas the Phi models are on the order of billions. This large increase in parameters appears to result in the models performing better, and a positive correlation with an R^2 value of 0.75 was found between the similarity score and the number of parameters in the LLM.

The improved responses from the Phi models do come at a cost, however: the base GPT-2 model provides a response in around 10 seconds. The Phi-2 model, however,

takes around 25 seconds to answer a question. While this may not seem like a huge difference in a vacuum, it is a 150% difference that could add up if interacting with a document in a "chat" fashion, where there will likely be iterative questions, follow-up questions, etc.

Finally, it is worth noting that the context does score decently well – this is likely due to the fact that the answer is contained somewhere in the context, but the context is long and not necessarily the convenient answer a user may be expecting. There may be domains where this is appropriate, or perhaps even better (to have the surrounding context straight from the document), but that is not the goal of this project – we feel that searching a document is just as valid of an approach as returning the full context.

Discussion

Applications like ChatGPT have changed how human-computer interaction takes place in our society. Because of ChatGPT, people are increasingly looking for a conversational approach to getting information and do not want to have to read through long documentation, know a querying language, or wait for an email reply. While technology like ChatGPT is certainly a boon to all companies, only the largest of large companies will be able to utilize its full power and have their own proprietary data be returned to users. The goal of this project was to explore whether a scaled-down version of this is accessible to a company with less resources.

After inspecting results to questions about the NFL rulebook, there is convincing evidence that yes, a company could use a pretrained LLM on their own proprietary information and provide its employees, end users, etc. with a way to "chat" with company information. While responses are often not perfect in the similarity score (there are no 1.0 similarity scores), many answers are 100% correct. Take Example 2 into consideration: the similarity score is 0.736 but user's question is answered in a plain text, straightforward fashion with the exact right dimensions as expected. This provides some confidence on top of already convincing similarity scores that the results being returned by the optimized model are, in fact, useful.

This is an exciting result: it suggests that even smaller companies with limited resources can leverage the power of conversational AI to streamline operations, improve user experience, and potentially even offer a service to users outside the company as a business model. Additionally, it is promising to see the playing field being somewhat level in the conversational AI landscape in that companies of all sizes are able to take advantage of the newest technology available.

Conclusion

This project shows the potential for a chat-driven document search engine powered by an LLM that is free of cost and simple to deploy. Models such as Microsoft's Phi-2 and Phi-3 can provide accurate responses to many domain (document)-specific questions. While not all responses are perfect and challenges remain in the relatively slow speed of responses, the results often are accurate and given the alternative methods to get an answer may take hours, or even days, the time quickly begins to feel reasonable. The benefits of deploying a system in this form are likely to be substantial for businesses and individuals.

Reflection

Our curiosity for this subject actually started before we began working on this project. We had had a few casual conversations about how an LLM could be valuable to a data science/data analytics team to reduce the number of what we call "dashboard questions" – questions that are a SQL query away but may not be accessible to non-technical employees and thus fall in the lap of the data team. Such work is probably an underutilization of the knowledge and skill base of the employees on the data team and thus not an effective use of their time. Enter a chat interface: if these non-technical employees can simply enter their question in natural language and get their response in natural language and carry on with their day, that is a fantastic time boon to both the non-technical employee and the member of the data team who would have been stuck with the work. After more discussions we realized it had a lot more potential than just that – any in-house information could be fair game, from data to documentation.

As we started on the project we realized how technical a problem this was. Of course an LLM is technical itself, but there was the problem of getting context to the LLM. There are limitations for providing the entire documentation as context for a LLM depending on the total size of the documentation, including performance degradation, input constraints crossed over, and time to respond increases. Context retrieval led us to learn more about vector stores and performing a similarity search to provide a context based on the questions, which is an interesting problem itself. We further learned about LLMs which plays nicely into the deep learning we learned in class and modern machine learning research.

As discussed in this paper, we find that this is an excellent potential value-add in practice. As we have it implemented, it won't be ChatGPT. But, we don't feel that ChatGPT should necessarily be targeted: it's state of the art, takes a lot of space, and takes a lot of storage. Furthermore, using the ChatGPT API can result in significant costs, and there are security hazards of sending private documentation to the LLM through the API that may need to stay private. ChatGPT takes an enormous amount of

time and resources to train meaning it cannot possibly stay up to date on latest information found publicly online. There may be situations where an LLM should be trained on onfidential data sources that OpenAI, nor any other company, should be able to access. All that is to say: ChatGPT is not a good, or even possible, solution for all companies. The work in this project shows that it is possible for a company less flush in resources to have the ability to create an interactive chat agent that enables its employees or customers to quickly access information that would surely take more than the 10-to-30 seconds that this takes. The relative simplicity of implementing a model such as this (as compared with something even half as complex as ChatGPT) means that a company could, at no cost to them, get an MVP going in a matter of days or weeks and evaluate its impact on their business. We are confident they would find it worth their time and could in turn easily scale up their document corpus.

Appendix

Table 2 Similarity score statistics from the comparison of LLM models.

Model	Mean	Standard Deviation
n/a (context)	0.4301993762484834	0.3180499251083353
GPT-2	0.31182425765650745	0.27828509747004326
GPT-2 Large	0.333954009192088	0.2705386681673722
DistilGPT2	0.2902715327100286	0.2529973981304602
Phi-2	0.5697966280711603	0.2769086110888851
Phi-3	0.4744308906949208	0.318439841363638