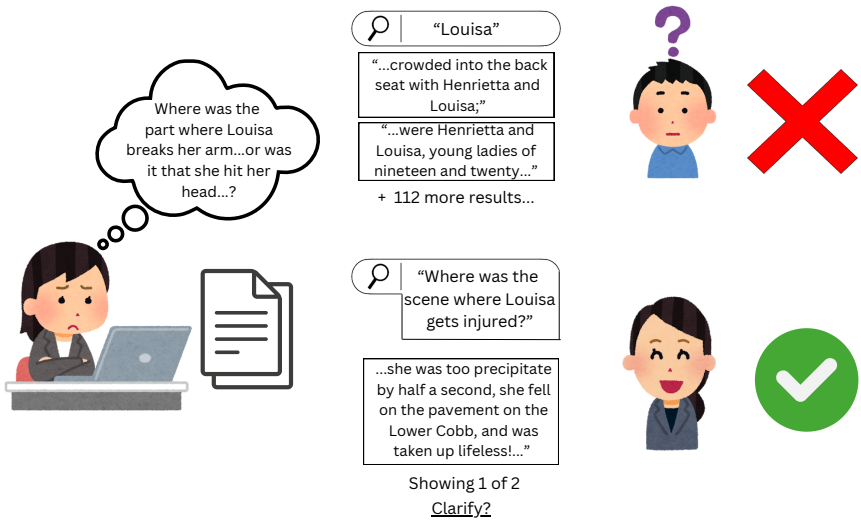
Search Through Text With Natural Language

Luke Patrick Rodgers
Earlham College
Richmond, Indiana
lrodge21@earlham.edu

Abstract



Search-through-text is a potential area of growth given recent natural language processing advancements due to large language models. Search-through-text refers to finding parts of a given text based on a user prompt, like Ctrl+F. I propose a system that allows the user to search through text not by keyword, but with natural language. This task is approached as a natural language inference task, using a model pre-trained for MNLI tasks as a zero-shot classifier to find parts of the text that align with the prompt. This project has the potential to expand the possibility of what large language models can be used for, as unlike traditional Large Language Model utilization, this task is text input to non-text output, instead of text-to-text.

1 Introduction

The capabilities and methods of computers in processing and understanding text has improved by leaps and bounds in recent years. The field of natural language processing (NLP) is experiencing a revolution [8], and new ways of working with text are being explored energetically [9]. Large language models (LLMs) are by far the most popular technology related to NLP tasks in recent years. At the creation of the modern LLM architecture, already this technology was able to achieve the state-of-the-art on machine translation tasks [18]. This transformer architecture, the core of an LLM today, is used for a wide variety of NLP problems. Currently, LLMs and related systems are capable of the best performance in machine translation [6], coreference resolution [4], question answering, summarization, text classification and many other tasks [16].

"LLM" specifically just refers to a language-based transformer architecture that is very, very large — trained on enormous amounts of data and with billions of parameters. They are useful in that they generalize well to many tasks, which means investment can be focused in one place [8]. Many people know of LLMs because of the popularity of ChatGPT, which is widely used for some of the aforementioned tasks, as well as many other generative tasks, like writing an email, code, or poem.

This new wave of AI has naturally found its way

into search features. ChatGPT can be used to find information that you would otherwise search for on the internet. Google provides an "AI Overview" when you search that provides a quick summarization of the searched topic along with links to where the information came from [7]. Bing does something similar [13]. The newspaper *The Economist*'s article search feature sports encouragement to test out their new feature and "try AI-powered search" [17].

While this kind of web information search moves forward, some of our other ways of searching remain stagnant. In the field of searching through text, the only prevalent option is keyword search, or as most people know it, Ctrl+F. In my research, I intend to develop a more flexible alternative to this, and the following survey of relevant work (sections 2 & 3) will cover the techniques and technologies that may prove helpful to this goal.

2 Brief Overview of LLM Structure and Functionality

LLMs are nigh-universally an implementation of the transformer architecture. Introduced in 2017, the transformer built on the models that were being used at the time by moving from models with complex recurrent or convolutional neural networks and attention mechanisms to a model with just an attention mechanism [18]. The vanilla transformer is sequence-to-sequence (converts an input sequence to an output sequence) and consists of an encoder and a decoder. Each is made up of a stack of blocks. In the encoder, each of these blocks is made up of a multi-head self-attention mechanism and a feed-forward network. In the decoder, each block is made up of these two things and an additional multi-head self-attention mechanism that works with the output of the encoder stack. Additionally, the decoder's self-attention mechanisms are modified to mask tokens that come after a given token [11]. A diagram is included as Figure 1.

Transformers now come in many different variations and have branched off in many different directions. These differences can be found in model application, pre-training, architecture, and module design. While originally created as a sequence-to-sequence model, current models can be found to be encoder-only or (in the case of GPT) decoder-only [11]. In Figure 2, a visualization shows how these differences in transformer architecture or otherwise have resulted in a large variety of related LLMs.

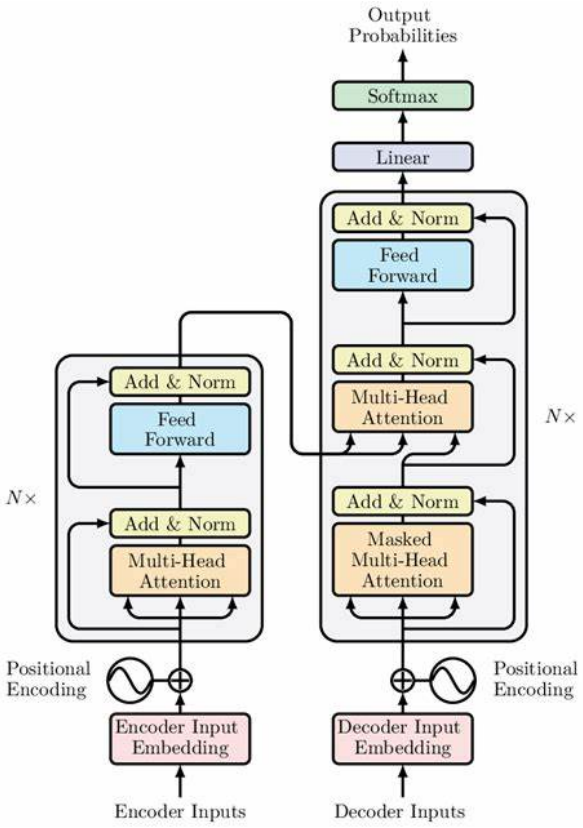


Figure 1: Original transformer architecture diagram taken from Vaswani et al.

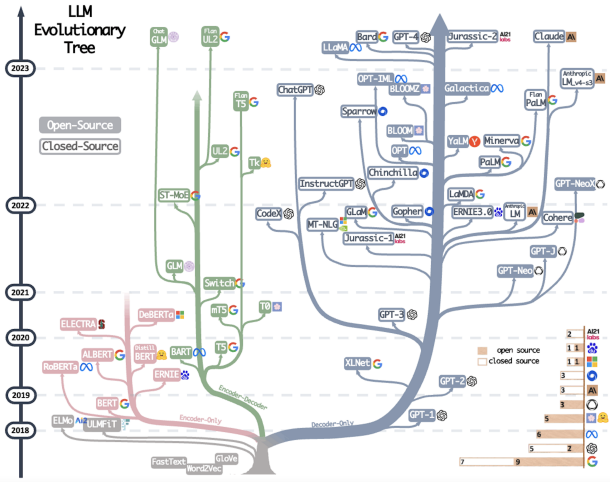


Figure 2: Tree diagram from Yang et al. depicting the variety of LLMs

2.1 Large Language Models in Reading Comprehension

The most talented AI models are capable of performing better than the average human on complicated tasks requiring a somewhat in-depth understanding of the questions being asked. GPT-4 scored a 5 (the maximum score) on the AP U.S. History exam [14] which has questions like: "Which of the following contributed most directly to the enactment of the law in the excerpt?" (the excerpt in question is a portion of The Declaratory Act of 1766) [2]. At the same time, they can sometimes fail at more simple tasks - they have shown poor performance on questions like "Franck read to himself and John read to himself, Anthony and Franck. In this context, was Franck read to?" [3].

Models capabilities are measured in various ways by various benchmarks. Frequently models are judged based on their ability to perform tests that humans take, like the case of the AP history exam, many of which depend on reading comprehension skills. GPT-4 gets good marks on even the bar exam and LSAT, and also shows strong capabilities in specifically designed commonsense reasoning and reading comprehension tasks. One of these, and one of the ways to display LLM's reading comprehension abilities is the DROP (Discrete Reasoning Over the content of Paragraphs) benchmark, which asks models to answer questions based on text. These questions go beyond simple summarization or topic finding, but ask of the model to perform simple reasoning on the things it extracts from the text. An example they give is to answer a question like "Who threw the longest touchdown pass?" [5]. GPT-4 scores 80.9 where humans score 96.4 [14].

2.2 Large Language Models in Information Retrieval

Similarly to how transformer architecture systems are able to demonstrate reading comprehension or create human-like prose, they can also be focused on the task of retrieving information. This information retrieval can be imagined as something like what you can do with ChatGPT: search for and receive information in natural language. For this kind of information retrieval, there are many efforts to improve the abilities of models, one notable one being Retrieval-Augmented Generation (RAG), where the inherent capabilities of a model are improved by supplementing them with additional retrieval-based memory [10]. But other, more formatted approaches to information retrieval also exist. For example, the Universal Information Extraction (UIE) framework, which condenses understanding of the text into a specifically structured language - an example of which is provided in Figure 3 [12].

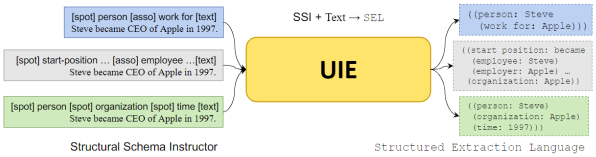


Figure 3: UIE framework including structured language example taken from Lu et al.

3 Utilizing LLMs

The most performative LLMs are prohibitively costly for an average person to develop. They require extensive resources of time and money, far beyond the threshold that would make it practical to develop one's own LLM without serious support [19]. Fortunately, it isn't necessary to build your own LLM to utilize their capabilities.

3.1 For downstream tasks

LLMs are best practically employed for their natural language understanding (good generalization ability), for generation tasks, and for knowledge intensive tasks. They are able to be fine-tuned up until a point - models with too many parameters (for example, upward of 20 billion), aren't practical for individuals or small organizations. An overview of available models and how they differ architecture-wise can be seen once again in Figure 2. Whether to use an LLM of large size or to use a smaller model that can be fine tuned generally depends on the amount of annotated data available. With little annotated data, fine-tuned models aren't as wise of a choice [20].

3.2 For unique tasks

Because LLMs only convert text-to-text, they have some inherent limitations. But they have been utilized in environments outside of pure text response. One example of this Cicero, an AI agent that uses a transformer-based system built off of base language model R2C2 to play the game Diplomacy. It utilizes the messages that it sends and receives as part of the game to inform its moves on the board [1]. This connects to the larger field of multimodal models, which pair an LLM with other systems of a different mode, for example, images. These models can

generally be simplified to systems that first convert a non-text mode into tokens that an LLM can work with, and then give those tokens to an LLM to process [21].

3.3 Fine-tuning

Larger, general language models can be adapted, or *fine-tuned* to utilize the basic language capabilities that the larger model already has for specific tasks. This circumvents some of the need to use massive amounts of manually labeled data. In this process, an additional linear output layer maps the output of the original transformer to task specific goals [15].

4 Methods

4.1 Design

The core of the design is a a pretrained LLM that has been trained for MNLI. The inference task is repurposed as a zero-shot classifier that evaluates how well a piece of text supports the information presented in the question. This is accompanied by a module that parses an input pdf, a module that batches a text into meaningful groups to be processed by the model, and a user interface that allows the user to freely query the model and displays the results of the query alongside the text. An overview is provided in figure 4

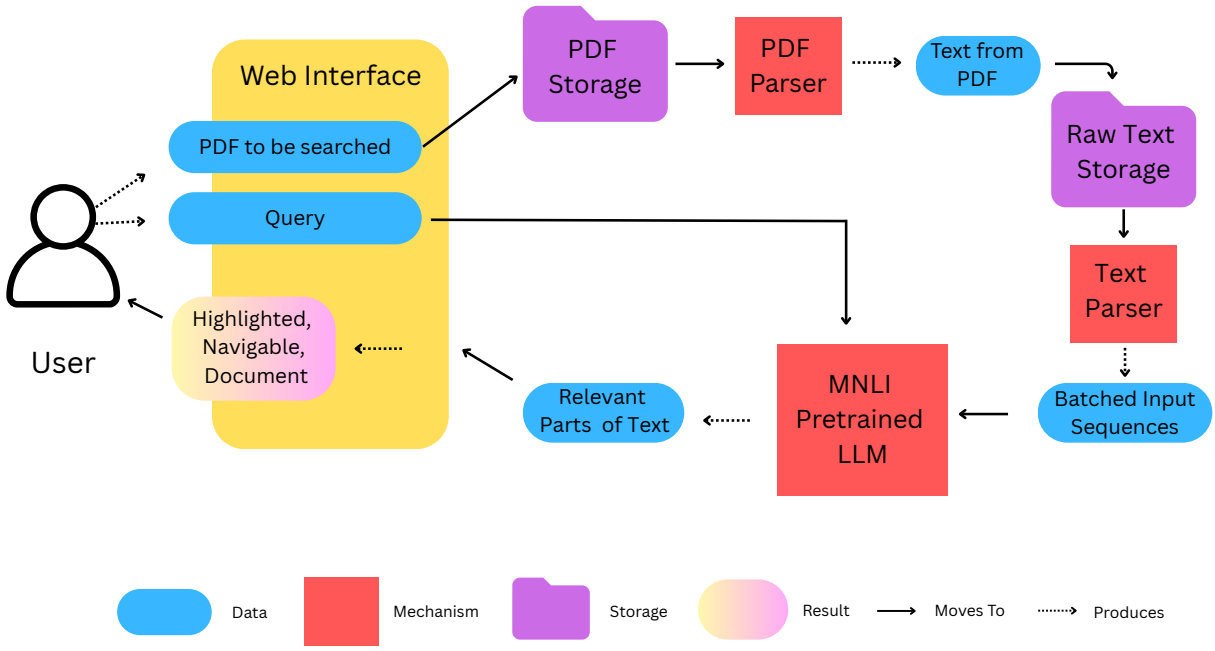


Figure 4: Architecture Diagram

4.1.1 User Interface

The user interface takes the form of a webpage that a field for uploading PDFs and a field for querying them. PDFs are displayed using a modified version of the pdf.js viewer and backend is handled by a Flask application. The modified version of pdf.js allows the embedded viewer to receive and display the parts of the text that the LLM has identified as relevant to the query. The Flask app handles the communication between the user and the programs on the server side.

4.1.2 PDF Parsing

PDFs were parsed with the python library pypdf.

4.1.3 MNLI Pretrained LLM

I used the pipeline architecture from Hugging Face's Transformer library to evaluate the relatedness of parts of the text to the query. This pipeline utilized a checkpoint of Facebook's BART large model that had been trained on the MNLI dataset.

4.2 Evaluation

The initial evaluation of the system's success will be based on informal testing and how well I perceive it to be accurately guiding the user to relevant portions of the text. Once I have some confidence that the model is performing well at least at first glance, I will move on to larger tests. The first will be against an annotated dataset. When crafting the dataset for training, I will leave part of the dataset out, and it will be used at the end of the process for evaluating results. This will compare the sections that the program has identified as solutions to the question with the pre-created answer key, awarding accuracy to the model when it has identified parts of the text that are also identified in the answer key, and subtracting accuracy when the model has identified parts of the text that are not specified in the key, or has failed to identify parts that are. The final portion of evaluating the model will be able to cover the most ground, and will utilize the same technique as was used for generating annotated data. The text, question, and answer given to/by the model will be given to a powerful LLM to confirm or deny. With these three techniques, a large scope of the system's usage and a reasonable sample size should be able to be tested. These are different methods, and it is unlikely that they will be comparable, so the expectation is two different scores will be produced for how good the accuracy of the model is.

5 Results

Initial results of comparing my own answers to the program's for a relatively short form text with a handful of queries like "Peter is in danger" or "It's night time" show that the program's answers are accurate (all but one were relevant portions of the text) but not exhaustive (almost half of the parts of the text that I identified as relevant were not identified by the program). Given the text "persuasion" and prompted to find where "Louisa gets injured" the top five results were all accurate, and 28/34 of all results were accurate.

6 Contributions

This project is somewhat unique in that it focuses on the utilization of LLMs for a system that does not output language. This conversion of fundamentally text-to-text LLMs to text-to-something-else systems is meaningful because it allows the powerful and prohibitively expensive to recreate capabilities of LLMs to be applied to a wider range of problems. This project, as a contribution to this, should be insightful as to what techniques do or do not work and how they can be implemented. Relatedly, it provides insight as to the capabilities of LLMs or transformer systems themselves, with regard to how well they can generalize. The final potential contributions come from the project design, which uses LLMs at multiple steps of the process. This kind of workflow, if successful, may be applied to projects with aspects similar to this one.

References

[1] Meta Fundamental AI Research Diplomacy Team (FAIR)† et al. "Human-level play in the game of Diplomacy by combining language models with strategic reasoning". In: *Science* 378.6624 (2022), pp. 1067–1074.

[2] College Board. *AP United States History Exam*. Accessed: 2024-11-26. 2024. URL: `https://apcentral.collegeboard.org/courses/ap-united-states-history/exam`.

[3] Vittoria Dentella et al. "Testing AI on language comprehension tasks reveals insensitivity to underlying meaning". In: *Scientific Reports* 14.1 (2024), p. 28083.

[4] Vladimir Dobrovolskii. "Word-level coreference resolution". In: *arXiv preprint arXiv:2109.04127* (2021).

[5] Dheeru Dua et al. "DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 2368–2378. DOI: `10.18653/v1/N19-1246`. URL: `https://aclanthology.org/N19-1246`.

[6] Sergey Edunov. "Understanding back-translation at scale". In: *arXiv preprint arXiv:1808.09381* (2018).

[7] Google. *Generative AI in Search: Let Google do the searching for you*. Accessed: 2024-11-25. 2024. URL: `https://blog.google/products/`

search / generative - ai - google - search - may-2024/.

[8] IBM. *What are large language models (LLMs)?* Accessed: 2024-11-25. 2024. URL: https://www.ibm.com/topics/large-language-models.

[9] Diksha Khurana et al. "Natural language processing: state of the art, current trends and challenges". In: *Multimedia tools and applications* 82.3 (2023), pp. 3713–3744.

[10] Patrick Lewis et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 9459–9474.

[11] Tianyang Lin et al. "A survey of transformers". In: *AI open* 3 (2022), pp. 111–132.

[12] Yaojie Lu et al. "Unified structure generation for universal information extraction". In: *arXiv preprint arXiv:2203.12277* (2022).

[13] Microsoft. *The New Bing*. Accessed: 2024-11-25. 2024. URL: https://www.microsoft.com/en-us/edge/features/the-new-bing?form=MA13FJ.

[14] R OpenAI. "Gpt-4 technical report. arxiv 2303.08774". In: *View in Article* 2.5 (2023).

[15] Alec Radford. "Improving language understanding by generative pre-training". In: (2018).

[16] Sebastian Ruder and contributors. *NLP Progress*. Accessed: 2024-11-25. 2024. URL: https://nlpprogress.com/.

[17] The Economist. *The Economist*. Accessed: 2024-11-25. 2024. URL: https://www.economist.com/.

[18] A Vaswani. "Attention is all you need". In: *Advances in Neural Information Processing Systems* (2017).

[19] Yuchen Xia et al. "Understanding the performance and estimating the cost of llm fine-tuning". In: *arXiv preprint arXiv:2408.04693* (2024).

[20] Jingfeng Yang et al. "Harnessing the power of llms in practice: A survey on chatgpt and beyond". In: *ACM Transactions on Knowledge Discovery from Data* 18.6 (2024), pp. 1–32.

[21] Shukang Yin et al. "A survey on multimodal large language models". In: *National Science Review* (2024), nwae403.