LLM-based analysis of sensemaking provenance

Submitted Sep 2023, in partial fulfillment of
the conditions for the award of the degree **MSc Computer Science.**

Xinhao Yang
20495020

Supervised by Kai Xu

School of Computer Science
University of Nottingham

I hereby declare that this dissertation is all my own work, except as indicated in the text:

Signature _____

Date _____ / _____ / _____

I hereby declare that I have all necessary rights and consents to publicly distribute this dissertation via the University of Nottingham's e-dissertation archive.

Public access to this dissertation is restricted until: DD/MM/YYYY

Abstract

In today's society, sensemaking decision is critical due to the popularity of datamining, especially when interpreting huge datasets in industries such as business and healthcare. With the advent of large-scale language models (LLMs), traditional data analysis, which is often laborious and error-prone, is undergoing a transformation. This paper explores the capabilities of OpenAI's GPT-3.5 Turbo, a leading-edge LLM, in enhancing dataset awareness. Its integration with Langchain tools simplifies application development and ensures seamless interaction with language models. In addition, the streamlit library facilitates the creation of interactive web applications, making data interaction more user-friendly. It allows easy uploading of data, efficient querying, and maintaining dialogue consistency by storing interactions in a single session. In conclusion, this paper presents an efficient, low-threshold tool that combines advanced LLMs with an intuitive dialogue interface in the hope of creating an efficient and accurate tool for sensemaking construction.

Acknowledgements

With the completion of this thesis, I am deeply impressed with the passage of time and the accumulation of knowledge. I would like to express my deepest gratitude to those who have given me endless support and encouragement during this process. First and foremost, I would like to thank my supervisor, who not only provided me with invaluable academic guidance, but also encouraged and supported me when I encountered difficulties. I have benefited from his/her professional knowledge and educational experience, and his/her patience and concern have strengthened me on my academic path. I would also like to thank our classmates, whose help and advice played a key role in my completion of this thesis. The discussions and exchanges between us have benefited me a lot and given me a deeper understanding of my research. In addition, I would like to thank my family and friends. Their support and encouragement were my motivation to complete this thesis. They were the ones who gave me the strength to carry on when I encountered difficulties and setbacks. I would also like to thank my school and college for providing me with an excellent learning environment and resources. I have had the opportunity to be exposed to cutting-edge academic research and interact with top-notch professors and fellow students, all of which have provided me with invaluable assistance in completing this thesis. Finally, I would like to thank those who have directly or indirectly contributed to this thesis. Whether it was providing data, supplying literature, or giving me advice and guidance during the dissertation writing process, I am deeply grateful. This thesis could not have been completed without the help and support of many people. I hope that my work can make a small contribution to the research in the related field, and I also hope that it can be recognised by all of you. Once again, I would like to thank all the people who supported and helped me, and I will always remember your kindness.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Sensemaking plays an important role everywhere in life. In business, sensemaking Building is used to help entrepreneurs understand market changes and consumer needs by analysing a large amount of consumer buying preferences and media trends, and ultimately formulate effective strategies. Similarly, in healthcare, sensemaking can analyse large amounts of patient data to help doctors identify diseases more accurately and suggest effective treatments. However, traditional data analysis is cumbersome, time-consuming and results in inaccurate analyses. At the same time, the rise of artificial intelligence, especially the large language data model, has gained more and more attention, and people are very surprised at how fast and accurate it is. As a result sensemaking building has grown by leaps and bounds with the help of big language models, making it easier, more accurate, and faster to interact with people. These models have revolutionised the way businesses and healthcare professionals deal with data analysis and decision-making processes.

This paper improves the perceptual process of different datasets by using Large Language Models (LLM) on different datasets. And, building a dialogue platform on top of LLM reduces the difficulty for users to use it and allows them to express their questions using natural language to better understand their data. The goal of this paper is to provide an intelligent, user-friendly system that bridges the gap between large datasets and the insights sought by users, which helps them to use and improve the accuracy of data analysis.

To achieve this goal, this paper uses OpenAI GPT-3.5 Turbo's Big Language Engine. This is one of the hottest and most popular LLMs available today, which ensures the sophistication and popularity of the system. Based on this LLM, this paper also uses the very advanced Langchain tool. It is a powerful framework designed to help developers build end-to-end applications using language models. It provides a set of tools, components, and interfaces that simplify the process of creating applications powered by the Large Language Model (LLM) and the Chat Model.LangChain makes it easy to manage interactions with the language model, link multiple components together, and integrate additional resources such as APIs and databases.The GPT-3.5 Turbo performs very well with langchain. very good performance under langchain. In addition, GPT-3.5 Turbo This paper uses the advanced streamlit library to build the interaction interface. streamlit is an open source Python library for creating machine learning and data science web applications with minimal effort. It provides an easy and fast way for data scientists and developers to transform data scripts into interactive web applications without any web development skills. Under the Streamlit framework, user interaction with GPT-3.5 Turbo becomes very simple and easy, and for program developers, they do not need to rewrite the front-end code and back-end code, which greatly simplifies the workflow and reduces the workload. Based on the above techniques, this paper implements: Data uploading and processing: The system provides a straightforward mechanism for users to upload datasets in CSV format. After uploading, these datasets will be converted into vectors through the FAISS vector storage system for efficient querying. Dialogue retrieval mechanism: By establishing a dialogue retrieval chain, the project ensures that the most appropriate response to a user query is matched to the uploaded dataset. Dialogue management and storage: To maintain context and coherence, all interactions are stored in a single session. In addition, users have the flexibility to save and download their dialogue history.
In summary, the project is a tool for creating simpler and more accurate sensemaking using state-of-the-art technology through LLM and dialogue interfaces.

1.1 Motivation

In today's information society, data has become inextricably linked to everyday life, generating large amounts of data every day, which is highly valuable in a wide range of industries. However, with the increasing volume and scope of data, it has led to the following issues:

Challenges for users from non-technical backgrounds: Many key decision makers and stakeholders, such as business leaders, doctors and researchers, may not have a deep technical background. This means they may not have the ability to use sophisticated data analytics tools, but they still need to gain insight into their data. We need a tool that is both powerful and easy to use to meet this need.

Communication barriers between humans and machines: while modern AI technology has made great strides in many areas, there are still huge challenges in interacting with people. Many people with different cultural backgrounds have different speaking styles and we need a tool that can understand and parse natural language to provide answers in a more human and intuitive way.

Information overload and the need for a solution: In this digital age, we are inundated with massive amounts of data every day, resulting in information overload. Traditional data analytics tools are no longer relevant as they are unable to process and parse such large amounts of data efficiently. This situation strongly suggests that we need a new, more powerful and efficient tool to process and parse data.

Considering the existence of the above problems, this project uses the capabilities of advanced large-scale language models to provide an intuitive, efficient, and easy-to-use platform that helps different users to solve the above problems and enhance the user experience, and at the same time allows the system to better understand and analyse the data, and to output the results that the users want.

1.2 Aims and Objectives

Based on the above existential motivation, this project aims to:By creating a user-friendly interface, users can query, analyse and interpret data through simple, natural language inputs. It can be easily used by anyone without the need for specialised knowledge of data analysis. This will enable cross-domain applications, from business, healthcare to research and other fields, the system will provide accurate data interpretation to meet the specific needs of different fields for data analysis. At the same time, by leveraging the latest language models, the system will be able to process and analyse large amounts of data in real-time and provide meaningful insights and answers in a short period of time, helping users to build sensemaking.

1.3 Description of the work

This project implements the advanced OPENAI 3.5 language model using Langchain technology and creates a user-friendly interface on top of the advanced streamlit library. The following functions are implemented in this project:

1.Interactive data querying: Users can enter simple natural language queries to obtain specific information from data files without technical background.

2.Real-time response: Combined with the advanced OPENAI big model technology, the platform can provide accurate answers instantly.

3. User-friendly: In addition to interactive queries, it also allows users to save, download and view the complete dialogue history with the data.

The following are the running steps of this project:

1. API Secret Key Input: To ensure access to OpenAI, users need to provide their OpenAI API secret key.

2. File Upload: Users can upload a file of type "txt". The system will read this data and prepare it for interactive querying.

3. The file is uploaded as a "txt", and then each line of the file is obtained as a string and stored in the DataFrame format.

4. Data Transformation and Indexing : Embedding data and storing it in FAISS , which is an efficient library for similarity search .

5. Interactive Queries : By combining OpenAI models and FAISS search functions, real-time responses are provided to users. Each query entered by the user interacts with the processed data and provides relevant answers.

6. By combining OpenAI models with FAISS retrieval capabilities, real-time responses are provided to the user. Each query entered by the user interacts with the processed data and provides the relevant answer. Before providing an answer, the sum of the tokens of the input question and the output answer is also calculated to prevent exceeding the maximum number of tokens specified in OPENAI 3.5, and if it exceeds the maximum number of tokens, the first record in the history is deleted in a loop until the sum of the tokens of the question and the answer is less than the maximum number of tokens specified in OPENAI 3.5.

7. Save and Download: Users have the option to save their entire dialogue history with the system. In addition, they can download that dialogue history as a JSON file for future reference or other uses.

In addition, before uploading the file, it will be processed from *csv* format to *txt* format, and from complex data to simple statements, so as to better allow OPENAI to analyse the big language model.

In conclusion, this project implements the use of the Large Language Model and applies the concept of human-centredness to achieve fast and accurate answers for sensemaking regardless of people's disciplinary background.

Chapter 2

Background and Related Work

2.1 Sensemaking

Sensemaking is a complex and critical process that involves gathering, organising and creating information to gain an in-depth understanding of issues, which means that it is not just a simple information processing activity, however the art of integrating dispersed data and knowledge into meaningful conclusions that can be clearly communicated to others.[1].Manual and automated methods of capturing information have an important place, and the judicious use of visualisation methods will help sensemaking to be better, even though sensemaking has many challenges and analysing the information can be difficult[2] . One of the very important processes in building sensemaking is capturing user interactions, insights, and reasoning, and to better accomplish information acquisition a visualisation framework can be used, which can be used to compare information and guide the assessment of provenance support [3]. Kai also presented a model for capturing and representing the analytical flow in visual analytics to help users understand and review their analytical steps, which includes multiple layers from the bottom event to the top task, which describes the entire analytical process, where states and operations can be displayed visually in various ways[4]. Additionally for addressing the use of narrative and visualisation in decision making, the researchers created the HTVA prototype to support narrative and visualisation of the decision making process using the ProveML format, which has been positively evaluated by peers and is seen as a powerful tool for informed

and decision making in human terrain analysis, as well as highlighting the future potential of visual analytics in this area[5].Furthermore, David describes a taxonomy of operations in a visual analytics environment that breaks down operations into three broad categories: data exploration, view exploration, and insights, and introduces meta-operations such as undo and redo to process the history of user operations, aiming to capture sources of insight and deepen understanding of how to combine multiple operations to complete high-level sub-tasks[6].

2.1.1 SensePath

To build semsemaking, a tool called SensePath to analyse user interactions in perceptual tasks, it a timeline visualisation tool that enables researchers to quickly assess overall user performance and identify interesting user behaviours, and the results show that the tool helps analysts to understand the way users approach a task and to identify recurring patterns in user behaviour patterns in user behaviour, although there are a number of further things that need to be developed with this tool, such as Analysts found it difficult to find the start and end times of user behaviours in the timeline view, and therefore had to refer to video and audio recordings to find out more information[7]. It includes a browser view, a history map and a knowledge map[8]. Of these, the Browser View is a standard web browser with additional perceptual features. The history map captures and visualises user actions, thus providing an overview of the perception creation process.

2.1.2 SenseMap

Another SenseMap tool designed for Human-computer interaction(HCI)/visualisation researchers to analyse the sense-making process provides timeline view, browser view, playback view and transcription view, as well as interaction/evidence capture, scaling, filtering, coding and communication[8]. The Browser View is a standard web browser with additional perceptual features. The history map captures and visualises user actions to help provide a sense of the creation process.

2.2 Large Language Model

Large language models are a type of artificial intelligence technology based on deep learning that has made significant breakthroughs in the field of natural language processing. These models have millions to tens of billions of parameters, enabling them to understand and generate human language, perform various natural language processing tasks such as text generation, text classification, question-answering, translation, and more. The development of large language models has gone through several crucial stages, including the early adoption of neural networks, the transformative impact of the Transformer architecture, and the explosive growth in model size.

2.2.1 Development of Neural Networks

The development of large-scale language models has become a trend that cannot be ignored in the field of artificial intelligence and machine learning. The emergence and advancement of these models is largely due to cutting-edge advances and continued innovation in neural network technologies. These technologies have laid a solid foundation for the modern field of Natural Language Processing (NLP), providing us with powerful tools for processing and understanding textual data.

In the early stages of language modelling, Recurrent Neural Networks (RNN) and Long Short-Term Memory Networks (LSTM) were widely adopted. Originally designed to process sequential data, these models process textual data sequentially, allowing information to propagate between different time steps[9]. However, although they perform well in some applications, they still have limitations when dealing with long sequences and establishing long-range dependencies.

To overcome these limitations, researchers are turning to other neural network architectures. Convolutional Neural Networks (CNNs) are among the best of these, having achieved great success in the field of computer vision [10]. However, its performance is somewhat limited when applied to NLP tasks. Nevertheless, some advanced research has successfully combined CNN with RNN to achieve higher performance in text modelling.

Furthermore, the emergence of word embedding techniques, such as Word2Vec and GloVe,

has revolutionised neural network language modelling. These techniques embed words into low-dimensional vector spaces, allowing neural networks to understand and generalise textual data in greater depth[11]. This embedding approach provided a rich semantic representation of the textual data, which in turn provided richer information for subsequent models.

However, the real turning point came with the introduction of the Seq2Seq (sequence-to-sequence) model and the attention mechanism. These techniques allow neural network models to handle sequence-to-sequence tasks, such as machine translation, more efficiently[12]. The attention mechanism, in particular, significantly improves performance by allowing the model to focus on different parts of the input sequence as it is processed. This mechanism provides a dynamic way for the model to focus on key parts of the input data, thereby improving the accuracy and robustness of the model.

The cumulative development of these techniques and methods laid the foundation for the emergence of the Transformer architecture and the development of large-scale language models.The Transformer architecture achieves outstanding performance by processing textual data through the mechanism of self-attention and multi-head attention. With the further refinement and application of these techniques, the field of NLP is undergoing an unprecedented phase of innovation and development.

2.2.2 Transformer-Based Revolution

In recent years of artificial intelligence research, the Transformer architecture has undoubtedly become a bright star in the field of natural language processing. It has not only revolutionised the design thinking of traditional language models, but also provided a solid cornerstone for subsequent model innovations. In particular, the emergence of models such as BERT, GPT, and XLNet, all of which rely deeply on Transformer's core ideas, especially the self-attention mechanism and the multi-head attention mechanism, have achieved unprecedented performance for models on a wide range of tasks.

The Transformer architecture, first proposed by Vaswani et al[13]. in 2017, was originally designed to address the limitations of traditional Recurrent Neural Networks (RNNs) and

Long Short-Term Memory Networks (LSTMs) when dealing with long sequences.At the heart of Transformer are[14]:

(1) The self-attention mechanism: this mechanism allows the model to "flow" freely between locations in the input sequences This mechanism allows the model to "flow" freely between positions in the input sequence, thus capturing long-distance dependencies without the constraint of a fixed window size;

(2) Multi-attention mechanism: In this way, the model can capture information in multiple representation subspaces in parallel, thus providing a more comprehensive understanding of the input data;

(3) Positional encoding: since the Transformer itself does not have the ability to deal with sequence order, positional encoding becomes the key to endowing the model with this ability.

2.3 Performance Evaluation

The emergence and development of the Transformer architecture in the field of natural language processing undoubtedly marks the beginning of a new era. It not only challenges the limitations of traditional RNN and LSTM models in processing long sequence data, but also introduces a series of innovative mechanisms, such as self-attention mechanism and multi-attention mechanism, which provide models with deeper and more comprehensive text comprehension capabilities. In particular, the success of Transformer-based models such as BERT, GPT, and XLNet further proves the power and potential of this architecture.The design ideas and technical features of Transformer, such as free information "flow", parallel information capture, and sensitive handling of sequential ordering, provide support for its excellent performance in a variety of complex tasks. support. In short, the Transformer architecture not only revolutionises the field of natural language processing, but also opens up new possibilities and directions for future research and applications.

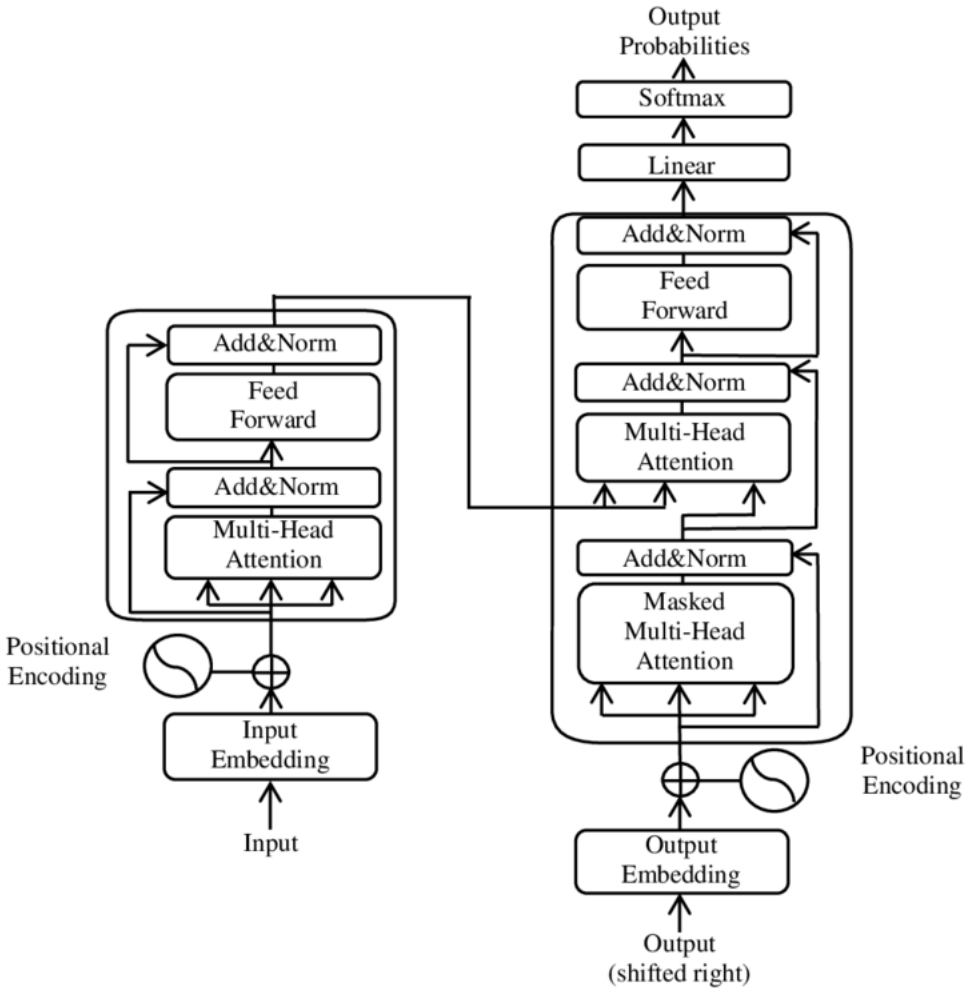

Figure 2.1: The-Transformer-model-architecture

BERT(Bidirectional Encoder Representations from Transformers)

BERT (Bidirectional Encoder Representations from Transformers) has become a benchmark in the field of Natural Language Processing, introduced by Jacob Devlin and his team in 2018. Not only has this model made significant technological advances, but it has set new performance standards across a wide range of NLP tasks. Here is a more in-depth look at BERT through two-way context modelling and large-scale pre-training:
In traditional language modelling, text typically flows in one direction, from left to right or right to left. This approach, while valid to some extent, ignores a key fact: understanding the true meaning of a word usually requires taking into account its entire context, including the words that precede and follow it.BERT solves this problem by employing a bi-directional Transformer encoder, which allows it to take into account both the context to the left and the context to the right of any given word when parsing it[15]. This comprehensive contextual analysis provides BERT with an in-depth understanding of lexical contexts, especially in those cases where the meaning of a word is variable or dependent on a specific context.
Data is key to the success of deep learning, and the BERT model takes full advantage of this. During its development, BERT was pre-trained on a large-scale text corpus, which allowed it to learn not only vector representations of words, but also the deep structure and complex semantic relationships of language[16]. This large-scale pre-training strategy infuses BERT with rich linguistic knowledge, enabling it to demonstrate superior performance on a variety of NLP tasks, ranging from text categorisation to question-answer systems.
BERT represents a giant leap forward in the field of natural language processing. Its ability to model bi-directional context and its strategy of large-scale pre-training endow it with a deep and comprehensive understanding of language, enabling it to achieve unprecedented performance on a variety of NLP tasks. It also provides new directions and inspiration for future research and applications.

GPT (Generative Pre-trained Transformer)

The GPT (Generative Pre-trained Transformer) family of models is a series of pre-trained language models based on Transformer introduced by OpenAI, and they have attracted much attention and discussion in the field of natural language processing. Unlike models like BERT which mainly focus on text encoding and classification tasks, GPT focuses more on generative tasks, such as text generation, text completion, story creation, etc. GPT has gone through four versions of iterations until it has now become a household name for large language models.

In the initial version of GPT-1, researchers proposed a new approach to enhance language understanding that first pre-trains on a large amount of unlabelled text and then fine-tunes the model for a specific task. This two-stage training strategy combines unsupervised pre-training and supervised fine-tuning, proving its effectiveness on multiple language comprehension tasks[17]. This approach performs better on 9 out of 12 tasks compared to a model designed specifically for a particular task. In addition, the model demonstrated robust language knowledge in a zero-sample context. Overall, this study successfully combines unsupervised pre-training and supervised fine-tuning, setting a new standard for improving language understanding.

The focus of GPT-2's research is to show how language models can excel on a variety of natural language processing tasks through unsupervised multitask learning. Not only is this model able to handle complex tasks such as text summarisation, conversational question answering and translation of rare words, but several papers are referenced to explore its capabilities in depth[18]. However, despite the fact that GPT-2 shows strong performance in several aspects, it still has some limitations, such as being more biased towards focusing on the recent content in the text and possibly confusing some specific details. Nonetheless, the results of GPT-2's research still demonstrate the great potential of natural language processing techniques, especially in terms of the model's ability to learn and perform tasks from real-world data.

GPT-3 (Generative Pre-trained Transformer 3) is now the most used large-scale language model, with a staggering 175 billion parameters, making it one of the largest language

models to date. Compared to its predecessor, GPT-2, GPT-3 has achieved significant improvements in model size, performance, and breadth of applications.

Firstly, in terms of model size, GPT-3 has nearly 100 times more parameters than GPT-2. This scale expansion enables GPT-3 to better capture and understand complex linguistic structures and contextual relationships, thus achieving higher accuracy on a variety of linguistic tasks [19]. In fact, GPT-3 demonstrates comparable or even superior performance to task-specific trained models on several natural language processing tasks. Second, one of the distinctive features of GPT-3 is its "zero-sample" or "few-sample" learning capability[19]. This means that, unlike traditional deep learning models that require large amounts of labelled data for fine-tuning, GPT-3 can understand and perform specific tasks with few or no examples. This powerful generalisation capability greatly reduces the difficulty and cost of model deployment.

In real-world application scenarios, the potential of GPT-3 is almost ubiquitous. In terms of text generation, GPT-3 can write articles, stories, poems, and even code. The quality of the text it generates is so high that it is often difficult to distinguish it from human authors. In addition, GPT-3 has demonstrated excellent performance in the areas of question and answer systems, dialogue robots, and natural language interfaces. For example, users can ask GPT-3 questions in natural language format, and GPT-3 is able to understand the intent of the question and provide an accurate answer.

In addition to text processing, GPT-3 has been successfully applied to developer tools such as code completion and syntax correction. Its ability to understand and generate programming languages allows developers to write and debug code more efficiently[20]. In addition, GPT-3 has found applications in a variety of fields such as education, healthcare, and entertainment, such as intelligent educational assistants and medical consultation robots.

GPT-4, as OpenAI's flagship language model, is undoubtedly the pinnacle of natural language processing technology in recent years. Inheriting the foundation of GPT-3, GPT-4 has been comprehensively upgraded and expanded in terms of model structure, algorithm optimisation, application breadth and depth[21]. In terms of model scale, the number

of parameters in GPT-4 exceeds that of all previous language models, and this unprecedented scale provides it with richer knowledge reserves and more powerful computational capabilities. This not only means that GPT-4 can understand more complex linguistic structures, but also that it can capture the details and contexts in the text more precisely. Technically, GPT-4 employs a series of cutting-edge algorithms and techniques. New self-attention mechanisms and optimisation strategies make the model more efficient during training, while also ensuring stability and reliability in real-world applications. In addition, GPT-4 introduces a variety of new techniques, such as knowledge distillation and meta-learning, to further improve the generalisation ability and adaptability of the model.

In real application scenarios, the performance of GPT-4 even exceeds expectations. In addition to its continued leadership in traditional NLP tasks, GPT-4 has also successfully entered some new fields. For example, in professional fields such as medicine, law and finance, GPT-4 has demonstrated astounding comprehension and generation abilities. Meanwhile, its application in creative writing, art creation and other fields also shows us the infinite possibilities of AI[22].

What's more, GPT-4 has also made many innovations in combining with other advanced AI technologies[22]. The deep integration with neural symbol system, reinforcement learning and other technologies makes GPT-4 not only able to handle traditional text tasks, but also able to make intelligent decisions and interactions in more complex environments, such as virtual reality and robot navigation.

GPT-4 not only represents the highest level of current natural language processing technology, but also foretells the future development trend and direction of AI technology. Its excellent performance, wide range of applications and unlimited innovation potential all make GPT-4 a well-deserved star of the AI world.

Applications of GPT-3.5

With the advancement of technology, the field of Natural Language Processing (NLP) is undergoing unprecedented changes. In the midst of this transformation, OpenAI's GPT-

3.5 has undoubtedly become the hottest focus in the field [23]. Not only does this model represent a new era in the field of NLP, but it provides researchers and developers with a powerful and versatile tool that enables them to achieve unprecedented results in a wide range of applications.

The emergence of GPT-3.5 has led to significant enhancements in a number of subfields, including text generation, dialogue systems, and machine translation. For example, in text generation, GPT-3.5 is able to understand complex user commands and generate accurate, coherent, and high-quality text[24]. This makes it easier for content creators to create articles, stories, or other text content.

In addition, the application of GPT-3.5 in dialogue systems greatly enhances the interaction experience between machines and humans. While traditional chatbots may give inaccurate answers due to misinterpretation, GPT-3.5 is able to more accurately understand the user's questions and provide appropriate answers, thus providing a more natural and smooth interaction experience[25].

In terms of machine translation, GPT-3.5 is able to more accurately capture the subtle differences between the source and target languages through its deep learning capabilities, thus improving the accuracy and smoothness of translation[26].

However, while GPT-3.5 brings many positive changes, it also brings some problems. For example, there are issues such as how to ensure that the output of the model is fair and unbiased, and how to protect user privacy[27]. These issues need to be further explored and solved by researchers and developers. Overall, GPT-3.5 brings great opportunities and challenges to the NLP field. With the continuous progress of technology, we have reason to believe that there will be more breakthroughs and innovations in this field in the future.

2.3.1 LangChain: Revolutionizing Language Model Application Development

LangChain is not merely a framework; it represents a paradigm shift in how developers approach language model applications. With its rich array of tools, components, and

interfaces, LangChain simplifies the process of building applications supported by Large Language Models (LLMs) and chat models[28].One of the standout features of LangChain is its modular approach to application development. Components serve as the building blocks, which can be seamlessly integrated to create powerful applications1. The concept of Chains, a series of interconnected components, exemplifies the flexibility and scalability of the framework[28].Furthermore, LangChain introduces innovative features like Prompt Templates and Values, which streamline the interaction with language models. The integration of Example Selectors, Output Parsers, and Indexes and Retrievers further enhances the framework's versatility, making it adaptable to a wide range of application requirements.

LangChain's influence extends beyond mere text generation. Its capabilities have been harnessed in the development of advanced chatbots and virtual assistants, setting new benchmarks in user interaction. Moreover, its relevance in sectors like education underscores its potential to revolutionize various industries[29].

While LangChain offers a plethora of advantages, it is not devoid of challenges. As with any technology, ensuring the reliability and accuracy of applications developed using LangChain is paramount. Developers need to be cognizant of potential pitfalls and must fine-tune and validate models meticulously.The future of LangChain looks promising [30]. As more developers and researchers engage with the framework, it is anticipated that LangChain will continue to evolve, offering more advanced features and tools to cater to the ever-growing demands of the language model application landscape.

How LangChain works

Simply speaking, LangChain is about combining large amounts of data so that LLMs can be easily referenced with as little computational effort as possible. It works by taking a large data source, such as a 50-page PDF file, breaking it up into chunks, and then embedding them into a Vector Store.

Now that Langchain has a vectorised representation of large documents, it can be used to work with LLM to create a prompt-completion pair by retrieving only the information that

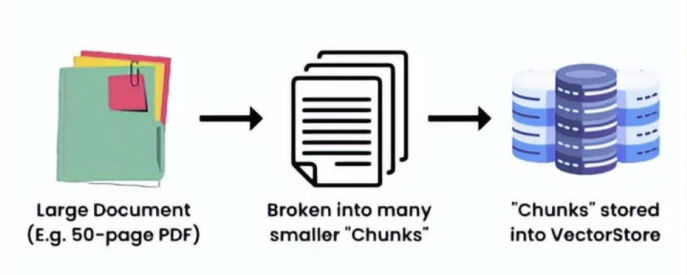

Figure 2.2: storage vector

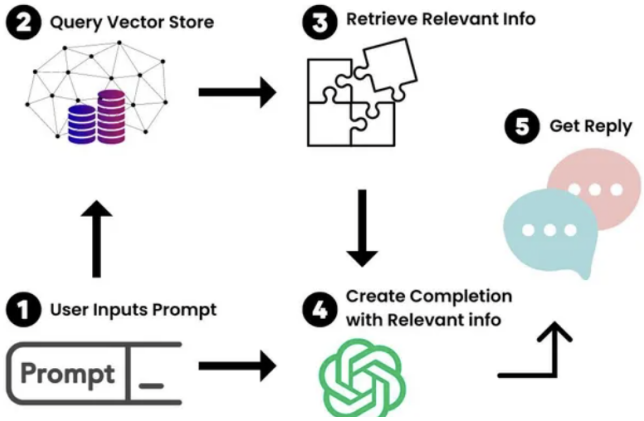

Figure 2.3: How LangChain works with OpenAI LLM

needs to be referenced. When a user enters a prompt into our new chatbot, LangChain will look up the relevant information in the vector store. You can think of it as a little Google dedicated to your documents. Once the relevant information is found, we use it to feed the LLM with the hint, generating the user's answer.

2.3.2 LangChain core concepts

Components and Chains

In LangChain, Components are modular building blocks that can be combined to create powerful applications, and Chains are a series of Components (or other Chains) that are combined together to accomplish a specific task. For example, a Chain might include a Prompt Template, a Language Model, and an Output Parser that work together to process user input, generate responses, and process output.

Prompt Templates and Values

Prompt Templates are responsible for creating PromptValues, which are what is ultimately passed to the language model.Prompt Templates help to convert user input and other dynamic information into a format suitable for the language model.PromptValues are classes with methods that can be converted to the exact type of input expected by each model type (e.g., text or chat messages).PromptValues are classes that can be converted to the exact type of input expected by each model type (e.g., text or chat messages). PromptValues are classes with methods that convert to the exact type of input expected by each model type (e.g., text or chat messages).

Example Selectors

Example Selectors are useful when you want to dynamically include examples in Prompts. They take user input and return a list of examples to use in the prompt, making it more powerful and context-specific.

Output Parsers

Output Parsers are responsible for structuring language model responses into a more useful format. They implement two main methods: one for providing formatting instructions, and the other for parsing the language model response into a structured format. This makes it easier to work with output data in your application.

Indexes and Retrievers

Indexes are a way of organising documents to make it easier for the language model to interact with them. Retrievers are interfaces for fetching relevant documents and combining them with the language model.LangChain provides tools and features for working with different types of indexes and retrievers, such as vector databases and text splitters.

Chat Message History

LangChain interacts with the language model primarily through the chat interface. the ChatMessageHistory class is responsible for remembering all previous chat interactions, which can then be passed back to the model, aggregated, or otherwise combined. This helps maintain context and improves the model's understanding of the dialogue.
7. Agents and Toolkits Agents are entities that drive decision making in LangChain. They have access to a set of tools and can decide which tool to invoke based on user input.Tookits are a set of tools that, when used together, can accomplish a specific task. The agent executor is responsible for running the agent with the appropriate tool. Help me summarise the above in one paragraph.

2.3.3 Streamlit

Streamlit is the first application development framework specifically for machine learning and data science teams, it's the fastest way to develop custom machine learning tools, and you can think of it as aiming to replace Flask in machine learning projects, and can help machine learning engineers to quickly develop user interaction tools[31]. At the same time. It is also an open source Python library that allows you to create project-attractive web applications for machine learning and data science. It provides a simple mapping approach to building and deploying web applications without the need to understand web development languages such as HTML, CSS, or JavaScript. In recent years, Streamlit has become the tool of choice for data scientists and machine learning researchers to create interactive data visualisation and model presentation applications, and although Streamlit is primarily used in data science and machine learning applications, its simplicity and flexibility have led to its widespread use in other areas such as web development and data engineering[32].
Streamlit makes it easy to create custom interfaces, visualise data and display machine learning models. It is compatible with various data visualisation libraries such as Matplotlib, Plotly and Altair, making it easy to create interactive charts. Notably, Streamlit incorporates caching to improve application performance and supports real-time updates

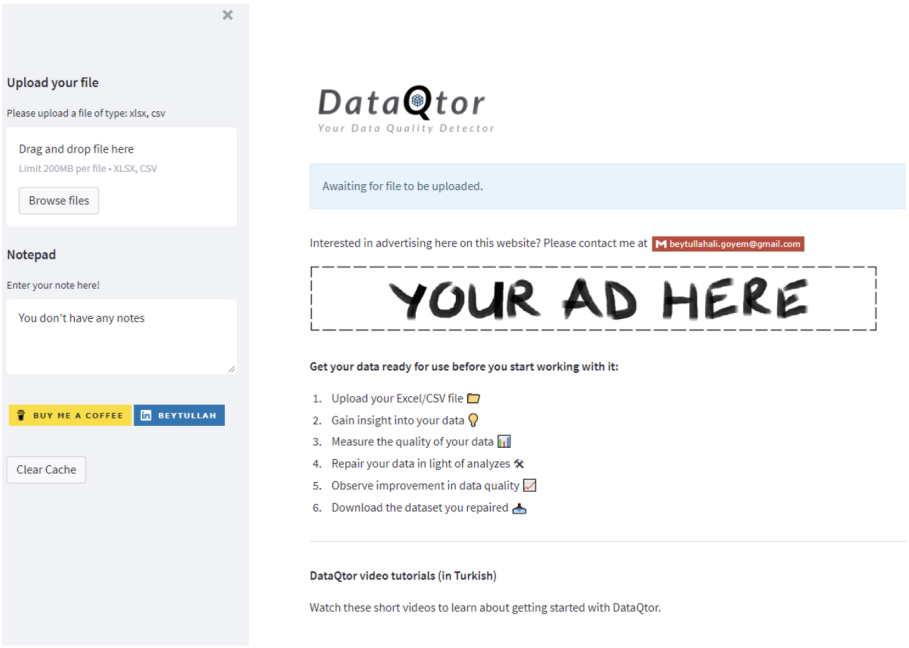

Figure 2.4: Example of a website built with streamlit

for dynamic responsiveness[33]. To use Streamlit, you need to write your application as a Python script that takes advantage of Streamlit's API to implement layouts and functionality. The script is run through the Streamlit command, which starts a local web server to host the application. Essentially, Streamlit simplifies the process of creating web applications for data science and machine learning projects, allowing developers to focus on core application functionality without having to deal with the intricacies of web development[34].

The advent of Streamlit has certainly revolutionised the field of data science and machine learning. It has simplified the process of creating web applications, making it easy for non-web developers to build and deploy interactive applications. In the future, Streamlit will continue to lead the field in innovation as the technology evolves, making it easier for researchers and developers alike!

Chapter 3

Design

In today's information society, data has become a central part of our lives. From business decisions to healthcare to everyday choices, data analytics provide valuable insights and guidance. However, traditional methods of data analysis often require specialised knowledge and are cumbersome and not easy to understand. This creates significant challenges for users from non-technical backgrounds. The design of our system endeavours to address the above issues as well as meet the need for an intuitive, efficient and user-friendly tool. With the rise of Artificial Intelligence and large-scale language models, the design of this project leverages the capabilities of OpenAI GPT-3.5 Turbo model, combined with a user-centric interface, to provide a seamless conversational experience for users from diverse backgrounds.

3.1 Design Ideas and Strategies

3.1.1 Simplicity

In today's digital age, complexity is a common problem with many technology products. Complicated user interfaces and difficult to understand features often leave users confused and frustrated. This project understands this and was designed with simplicity in mind first and foremost. The goal was to get users up to speed the first time they used it, without having to spend a lot of time learning and getting used to it.

3.1.2 Intuitive

Intuition is at the heart of design. The project wanted users to be able to intuitively understand how to use the system without having to consult complex user manuals or online tutorials. To this end, the project conducted extensive user testing to ensure that the interface and interaction design matched user intuition and expectations. In addition, the project uses clear icons and prompts to help users better understand the purpose of each feature.

3.1.3 User-centred

The project strongly believes that technology should serve people. Therefore, the project has always been user-centred in its design to ensure that the project's system meets the actual needs and expectations of its users. The project conducted in-depth interviews with users from different backgrounds to understand the challenges and pain points they encountered in data analysis. Based on this feedback, the project optimised the system to ensure a truly valuable solution for users.

3.1.4 Seamless Interactive Experience

This project believes that for data analytics to be truly valuable, user interaction with the system should be smooth and natural. We don't want users to experience barriers or interruptions in the process. That's why our system is designed to be conversational, allowing users to interact with the system as if they were talking to a real person. This type of interaction not only makes the user feel comfortable, but also provides more in-depth and personalised analysis results.

3.1.5 Designed for non-technical users

This project knows that not everyone has a technical background or experience in data analysis. But that doesn't mean they can't gain valuable insights from their data. Our system is designed for these non-technical users, ensuring that it is easy for them to use

without any programming or data analysis experience. Our goal is to democratise data analysis so that everyone can benefit from it.

In short, the design aim of this project is to ensure that our system is both simple and intuitive, whilst still providing valuable data analysis for users of all backgrounds. In this way,we hope that this project will help users in the construction of sensemaking.

3.2 Problems solution

Our system is designed to solve the user's problems and provide a more efficient, accurate and user-friendly data analysis tool. Here is how our system solves the problems encountered by users:

3.2.1 Real-time interactivity

Users no longer have to wait long periods of time for data to be processed and analysed compared to traditional data analysis tools. The project provides instant feedback, enabling users to interact with their data in real time, thus gaining valuable insights more quickly.

3.2.2 Natural Language Processing

Natural Language Processing: by leveraging the GPT-3.5 Turbo model, the system is able to understand and parse natural language queries, meaning users can interact with the system using their everyday language rather than relying on complex query languages or programming knowledge.

3.2.3 Easy-to-use interface

Easy-to-use interface: our design focuses on the user experience, ensuring that the interface is both intuitive and easy to use. This makes it easy for users from all backgrounds, whether they have a technical background or not, to use our system.

3.2.4 Integration of Advanced Technologies

Integration of Advanced Technologies: Our system integrates state-of-the-art technologies such as the GPT-3.5 Turbo model, FAISS Vector Storage System, etc. to ensure accuracy and efficiency in data analysis.

3.3 Design Functions

This system is not just a traditional data analysis tool, it combines the latest technology and design ideas to provide users with a series of powerful and innovative features:

3.3.1 Natural language queries

Users can ask questions directly to the system using natural language, without having to learn complex query languages or programming knowledge. This greatly simplifies the process of data querying and makes it easy for non-technical users to access the data information they need.

3.3.2 Instant Feedback

The system is able to provide users with answers in a short period of time, ensuring that users receive instant feedback when interacting with the data, thus improving work efficiency.

3.3.3 in-depth dialogue analysis

With the GPT-3.5 Turbo model, the system is able to perform in-depth dialogue analysis, providing users with in-depth, insightful answers to help them better understand the data.

3.3.4 Interactive Interface

This project provide an intuitive interactive interface that allows users to interact with the system through simple clicks and inputs without any additional training.

3.3.5 File Upload Function

Users can directly upload text files and the system will automatically read and process the file contents to provide users with relevant data analysis.

3.3.6 Chat History Management

The system provides chat history saving and downloading function, users can view, save or download their chat history at any time.

3.3.7 Token Counter

In order to help users better manage and control the complexity of queries, we provide a token counter which calculates the number of tokens used in a query in real-time.

3.3.8 Data Vectorisation and Retrieval

The system uses FAISS internally to vectorise data and provide fast data retrieval to ensure efficient user queries.

These functions ensure that our system not only meets the needs of programmers, but is also very useful for non-technical users who wish to gain data insights quickly and easily.

3.4 Reasons why it is designed this way

This design idea and functionality is chosen for this project because we believe that the true value of data analytics lies in the insights it can provide people, rather than its complexity or technicality. With our system, we want to make it easy for everyone to derive valuable information from their data without the need for any specialised skills or knowledge. Additionally, given the importance of data analytics in modern business and healthcare, our design aims to provide professionals in these fields with a powerful and intuitive tool to help them better understand market changes, consumer needs, and patient data, as well as, importantly, to help users better build sensemaking.

In summary, our design goal is to create a simple, intuitive, user-centred data analytics tool that provides valuable insights to everyone, not just technologists. In this way, we believe we can truly unlock the potential of data to deliver real value to individuals and organisations.

Chapter 4

Implementation

In the previous section, we described the design concept, design features, and the problems and reasons to be addressed in this project. Next, we describe how to implement this project. Firstly, the main technical tools used are described, then how the dataset was processed, followed by how it was implemented step-by-step, and finally the problems I encountered while programming.

4.1 Application Tools

4.1.1 Python

Python is a high-level, interpreted, interactive and object-oriented programming language. Due to its clean syntax and powerful data handling capabilities, Python has become the language of choice in the fields of data science, machine learning, and artificial intelligence.Python's rich set of libraries and frameworks, such as Pandas, Numpy, and Matplotlib, provide powerful support for data processing, analysis, and visualisation. In this project, Python not only serves as the main development language, but also provides the basis for data preprocessing, model integration and result analysis.

4.1.2 OpenAI GPT-3.5 Turbo model

OpenAI's GPT-3.5 Turbo is a large-scale language model that understands and generates natural language text. This model is characterised by its huge number of parameters and training data, allowing it to perform well on a wide range of tasks, from simple text generation to complex question answering. In this project, GPT-3.5 Turbo is used as the core analysis and answering tool, which is capable of providing in-depth, insightful answers to user queries.

4.1.3 Langchain

Langchain is a library of tools for developers that simplify the process of interacting with OpenAI models.Langchain provides a range of embeddings, chat models, and document loaders that allow developers to easily integrate OpenAI models into their applications. In this project, Langchain is used to process user queries, generate vector representations, and interact with GPT-3.5 Turbo models.

4.1.4 Streamlit

Streamlit is an open source Python library that allows developers to quickly create and share data applications.The main advantage of Streamlit is its simple syntax and intuitive interface design. Developers can build powerful, great-looking data applications without complex front-end development experience. In this project, Streamlit was used as the main development platform, which provided the user interface, file upload functionality, and real-time feedback.

These tools provided a solid foundation for the successful realisation of this project; Python provided a flexible programming environment, the OpenAI GPT-3.5 Turbo model provided in-depth answers to user queries, Langchain simplified the process of integrating the model, and Streamlit provided an intuitive, easy-to-use interface for the user. The combination of these tools ensures that the project is efficient, accurate, and user-friendly.

4.2 Dataset

4.2.1 Introduction

In this project, we have used a specific dataset which focuses on user interactions in visual analytics software. This dataset contains several labels such as ‘x‘, ‘y‘, ‘$algo$‘, ‘new_x‘, ‘new_y‘, ‘new_size‘, ‘ne
These labels document how users interact in the software and how they change the visual representation of the data.

The main content of the dataset describes how the user displays data for different countries, e.g. GDP, population, etc. Users can choose to display this information using ‘x‘ coordinates, ‘y‘ coordinates, size or colour. In addition, the user can change these mappings interactively, for example by changing the year of the data or choosing a different colour to represent the data. Each row in the data describes such a change, providing us with valuable information about the user's interactive behaviour.

4.2.2 Dataset Processing

The following data processing steps were performed to better understand and analyse user interaction behaviour:

1. File Output: firstly, we create an output file named ‘output.txt‘ to store the processed data.

2. Data Iteration: we iterate through each row of the dataset to check if the user's interaction behaviour has changed.

3. Interaction change detection: for each row of data, we compared ‘new_x‘, ‘new_y‘, ‘new_size‘, ‘new_colour‘,
file.

4. For country data processing: For the‘$new_country$‘tag, we do special processing. First, we extract the list of countries from the string, and then we convert these countries into a formatted string to make it easier to read and analyse, and to prevent subsequent operations from going wrong due to the complexity of the country data.

5. File Save: Finally, we save all the changes to the ‘output.txt‘ file and output a message informing the user that all the rows have been processed.

With the above processing, we obtain a clear, organised data file that details all user interactions in the visual analytics software. This provides a solid foundation for our subsequent analyses and research.

4.3 Project implementation steps

During the implementation of the project, we followed a clear set of steps to ensure that each function is properly implemented and provided a seamless experience for the users. Below are the detailed steps we took to realise the project:

1. Environment configuration and initialisation

Firstly, to ensure that the project's dependencies did not conflict with other Python projects, we used Python's virtual environment feature to build a separate development space. Using the venv module, we were able to easily build and maintain this virtual environment. 2. Importing necessary libraries and modules The implementation of the project relies on several external libraries and modules, such as streamlit, openai, langchain, etc. These libraries provide the necessary tools for the project. These libraries provide the necessary tools and functionality for the project, making the implementation process more concise and efficient.

3. Interface Configuration Using streamlit's set_page_config method, which is the st.write() method, we set the title, icon and layout for the application. This ensures that the application has a user-friendly interface at startup.

4 User input In order to interact with OpenAI's GPT-3.5 Turbo model, users need to provide their OpenAI API key. Additionally, we provide users with a file uploader that allows them to upload text files that will be used in subsequent dialogue interactions. This ensures the security of the key and prevents it from being placed in the programme and accessed by strangers.

5. Data processing The uploaded files are read line by line as string and converted into a Pandas DataFrame, which is then saved as a temporary CSV file for subsequent processing.

6. Embedding and vectorisation Using OpenAIEmbeddings, we generated embedding

vectors for the data. These embedding vectors were subsequently stored in the FAISS vector store for subsequent retrieval.

7 Dialogue chain creation We used ConversationalRetrievalChain from the Langchain library to create a dialogue chain. This chain combines the GPT-3.5 Turbo model with the vector retriever we created earlier. This way, when the user enters a question, the ConversationalRetrievalChain is simply called to answer the user's question.

8 Interactive dialogue Users can enter their questions via an input box. These questions are passed to our dialogue chain to get answers from the model. To ensure that interactions do not exceed the model's maximum token limit, we implement a token counting mechanism, which is to count the number of tokens generated by dialogues and new questions already in the history, and if it is greater than the maximum number of tokens specified by the$OPENAI_3$.5 model, then we loop through deleting the first record in the history dialogues until it is less than.

9. Saving and downloading dialogues Users can choose to save their dialogue history as a text file or download it as a JSON file. This provides a convenient way for users to review and share their conversations. This enhances the user experience and simplifies the user's use.

10. Running Guide To make it easier for other developers or users to run this project, we also provide a short runtime guide that describes in detail how to set up the environment, install dependencies and start the application.

Overall, the implementation of this project was a comprehensive process involving multiple steps and techniques. By combining an advanced language model, efficient data processing tools, and a user-friendly interface, we provide users with a powerful and intuitive data analysis tool.

The interface of the final project is as follows:

4.4 problems encountered

During the course of the project, I encountered many unanticipated difficulties and challenges. When I was first introduced to the knowledge domain of large language modelling

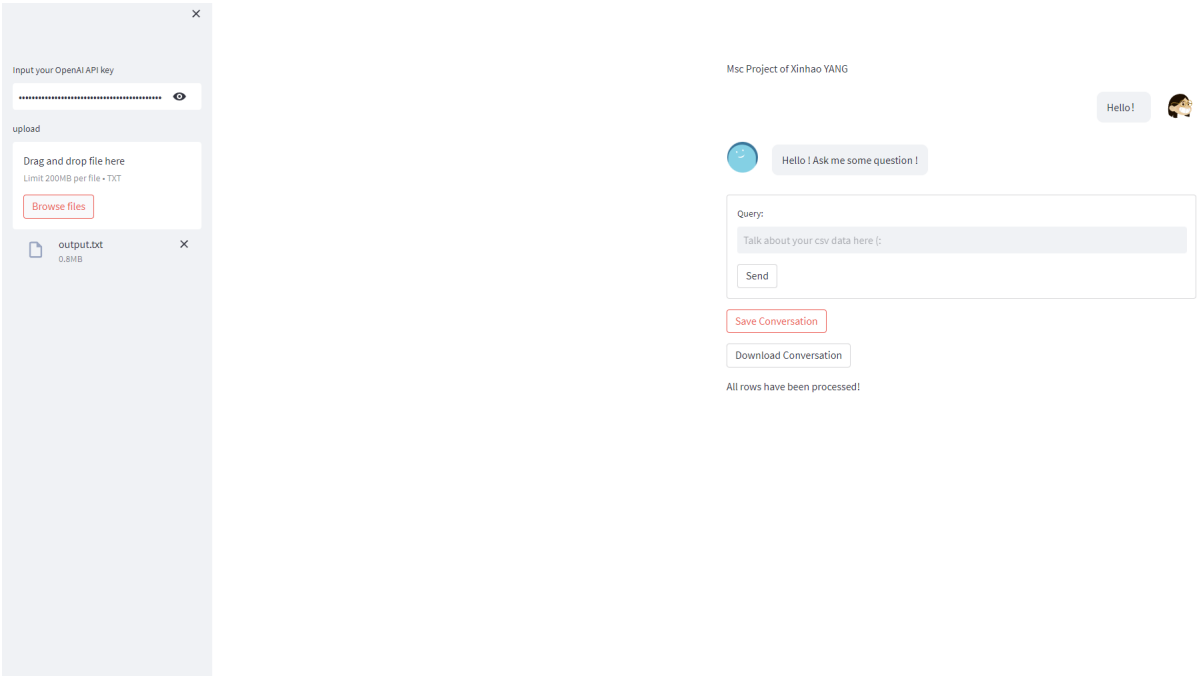

Figure 4.1: User Interface

and natural language processing, I was deeply overwhelmed. Although I had some previous programming experience, I felt somewhat overwhelmed by the amount of unfamiliar knowledge, code, and concepts. Every line of code, every function, and every module seemed to tell me that this was a whole new world that I needed to learn from scratch. Initially, I tried to make sense of this new knowledge by reading documentation and online tutorials. However, I soon realised that this approach was not enough to help me gain a deeper understanding of how big language models work and the complexity of natural language processing. Therefore, I decided to take matters into my own hands and study the code little by little, trying to understand the meaning and logic of each line of code. It was a time-consuming and tedious process, but I felt a sense of accomplishment every time I solved a problem or understood a new concept.

However, when I started running Langchain with OpenAI's code, I ran into a problem I never expected. On Google Colab, I couldn't successfully bring in the docarry package. I tried various methods and searched various online resources, but the problem persisted. This was very frustrating as I had invested a lot of time and effort into understanding the code, but was now stuck with a seemingly simple problem.

After many failed attempts, I decided to change my strategy. I began to consider if I could

run the code in another programming environment. After some research, I decided to try it locally in Visual Studio Code, using Anaconda's environment to manage dependencies. To my relief, this approach worked. I managed to bring in the docarry package and was able to run the code without any problems.

Looking back at the whole process, I realise that programming is not just about writing code, it's more about problem solving. Every problem, no matter how big or small, is a learning opportunity. Through this project, I not only learnt about big language modelling and natural language processing, but more importantly, I learnt how to face and solve problems, and how to stay calm and persevere in difficult situations.

Chapter 5

Evaluation

When it comes to software development, especially applications related to Artificial Intelligence and Natural Language Processing, ensuring software quality and user satisfaction is always paramount to the success of the project. In this digital age, users expect not just a fully-functional application, but more importantly an application that provides them with a seamless, efficient and enjoyable experience. For developers, this means that they not only need to master advanced technologies, but also need to have a deep understanding of users' needs and expectations. In this regard, the project not only demonstrates the application of advanced technology, but more importantly, it takes into account the needs and experience of the users. The following is a comprehensive assessment of the project.

5.1 Code Evalaution

5.1.1 Code structure and organisation

The code structure of the project is clear and well organised, and the modular design idea makes the interaction and integration between various components easy and intuitive. From importing necessary libraries and modules, to setting up basic page configurations, to defining the core chat functionality, each step was carefully designed and organised to ensure the readability and maintainability of the code.[**?**]

5.1.2 User Interaction

The application provides an intuitive sidebar where users can easily enter OpenAI API keys and upload files. This design not only keeps the user interface simple, but also ensures that users are able to seamlessly interact with the application and access the functionality they need.

5.1.3 Functionality Implementation

The project successfully implemented functionality for interacting with large language models. By using LangChain, a powerful framework, the project streamlined the interaction with OpenAI, making the entire process from user input to response generation to result output smooth and efficient.

5.1.4 Performance Optimisation

Considering the token limitations of large language models, the project took a number of measures to optimise performance. For example, when the number of tokens in the chat history exceeds the maximum limit, the project automatically deletes the early part of the history. This not only ensures the smooth running of the application, but also avoids potential errors caused by exceeding the token limit.

5.1.5 Data Processing

The project is able to efficiently process text files uploaded by users, converting them into a format suitable for further processing. In addition, the project also provides the ability to save and download chat history, which provides additional convenience for users to view and share chat history at any time.

5.1.6 Operation Guide

A detailed runtime guide is provided at the bottom of the project to instruct users on how to set up and run the application in their local environment. This ensures that even

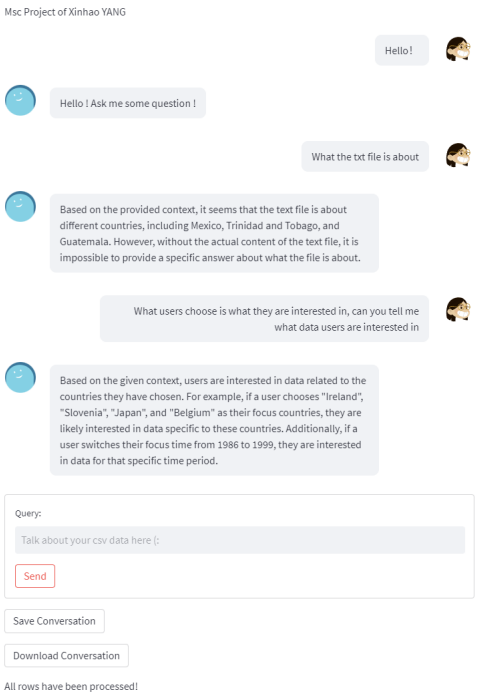

Figure 5.1: User Interface

first-time users of Streamlit and OpenAI can easily run and test the application.

5.1.7 Deployment and Access

It is worth noting that the project also supports deployment on the Streamlit Cloud. This means that once deployed, anyone can use the application by accessing the specified URL, without having to set up or install anything locally. This greatly increases the accessibility and popularity of the project, making it easy for more users to experience the convenience of the app.(URL:https://msc-final-project-kh6rnyohvzdwyfgxrgkpiw.streamlit.app/)
Here is our test output for sensemaking Build:

5.2 User case and user satisfaction assessment

During the development and implementation of this project, we recognised the importance of user feedback and decided to adopt two main approaches to user review: user surveys and user interviews. These two approaches will help us ensure that the project is more

closely aligned with user needs and expectations.

5.2.1 User Survey

Firstly, we designed a detailed user satisfaction questionnaire in order to get a full understanding of the user's comments and suggestions about the project. This questionnaire covers all aspects of the project, such as the completeness of the functionality, the intuitiveness of the interface, and the accuracy of the output results. In this way, we hope to find out the strengths and weaknesses of the project from the user's point of view, and provide directions for subsequent optimisation.

5.2.2 User Interviews

Secondly, in order to gain a deeper understanding of the actual user experience, we invited some of the main users of the project to conduct face-to-face in-depth interviews. This interview not only helped us to understand the problems encountered by users in actual operation, but also allowed us to harvest many valuable suggestions for improvement. These suggestions will provide important references for the next iteration of our project. Through the two review methods of user surveys and user interviews, we expect to be able to define the development direction of the project more clearly and ensure that each step is designed to better meet the needs of users.

5.2.3 User interface (UI) clarity

The project's UI was evaluated for intuitiveness and ease of navigation. Test example: A group of 20 users unfamiliar with the project were asked to navigate the interface and perform specific tasks. Results: Sixteen of the 20 users were able to perform the tasks without any guidance, indicating an 80 per cent clarity rate for the UI design. Here are statistics on whether the system output is clear or not

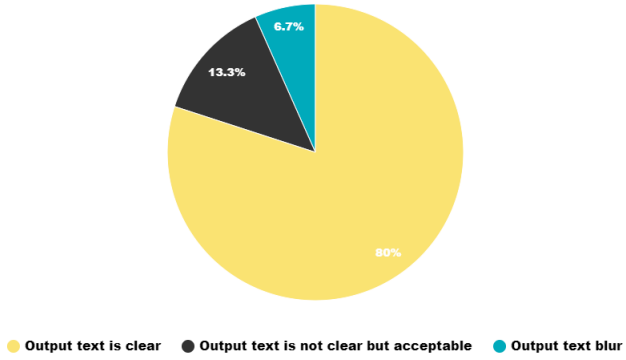

Figure 5.2: statistics on whether the system output is clear or not

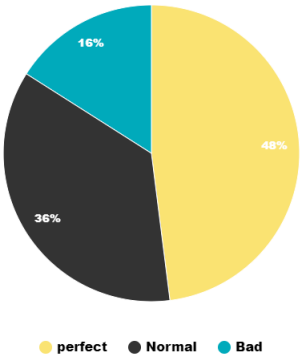

Figure 5.3: statistics on whether users are satisfied with the system

5.2.4 User satisfaction level

Evaluate the level of user satisfaction with the system, test example: A group of 25 users uses the system and then evaluates the system: 12 out of 25 think it is at a perfect level, 8 think it is normal, and 4 think it is bad. Here are the statistics on whether users are satisfied with the system

In summary, we have thoroughly evaluated the software from multiple perspectives and dimensions to ensure its quality, performance and user satisfaction. The results of these evaluations provide us with valuable feedback that helps us to continuously optimise the software to better meet the needs of our users and help them to build sensemaking.

Chapter 6

Summary and Reflections

In the course of the project, we have not only been developing and optimising on a technical level, but also thinking deeply and exploring the whole field in a broader context. Compared with other related work, our project has clear advantages in some aspects, but there are also some limitations and challenges.

The project started with an initial requirements analysis and went through a number of iterations and optimisations to reach the desired goal. Tasks for each phase were clearly listed and adjusted according to the actual progress. In order to ensure the smooth progress of the project, we set up a detailed schedule and adjusted it according to the actual situation. In terms of resource management, we rationally allocated human, material and financial resources to ensure that each task was adequately supported. We made full use of existing technical resources such as open source libraries, tools and platforms, and sought external support and co-operation where necessary.

The project demonstrated a certain degree of innovation in terms of technical implementation, user interaction and application scenarios. We are not just satisfied with existing technologies and methods, but endeavour to find and try new solutions.

Novelty: Compared with other similar work, our project has obvious novelty in some aspects, such as data processing methods and interaction design. Personal reflection: During the implementation of the project, I deeply appreciated the importance of teamwork, and each member made great efforts for the success of the project. At the same time, I have also recognised my shortcomings in technology and management, which provides me with

valuable experience for my future study and work.

Critical appraisal: Although the project has achieved certain results, there are still deficiencies in some aspects. For example, the performance and stability of the software still need to be improved, and user interaction and experience need to be further optimised. All these issues are directions we need to continue to work on in the future.

Overall, this project is not only a technical implementation, but also an in-depth learning and exploration process. We gained valuable experience and lessons from it, and also laid a solid foundation for future research and development.

Bibliography

[1] Phong H Nguyen, Kai Xu, Andy Bardill, Betul Salman, Kate Herd, and BL William Wong. Sensemap: Supporting browser-based online sensemaking through analytic provenance. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 91–100. IEEE, 2016.

[2] Kai Xu, Simon Attfield, T.J. Jankun-Kelly, Ashley Wheat, Phong H. Nguyen, and Nallini Selvaraj. Analytic provenance for sensemaking: A research agenda. *IEEE Computer Graphics and Applications*, 35(3):56–64, 2015.

[3] Eric D. Ragan, Alex Endert, Jibonananda Sanyal, and Jian Chen. Characterizing provenance in visualization and data analysis: An organizational framework of provenance types and purposes. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):31–40, 2016.

[4] Phong H Nguyen, Kai Xu, and BL Wong. A survey of analytic provenance. *Middlesex University*, 2014.

[5] Rick Walker, Aiden Slingsby, Jason Dykes, Kai Xu, Jo Wood, Phong H. Nguyen, Derek Stephens, B.L. William Wong, and Yongjun Zheng. An extensible framework for provenance in human terrain visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2139–2148, 2013.

[6] David Gotz and Michelle X. Zhou. Characterizing users' visual analytic activity for insight provenance. *Information Visualization*, 8:42 – 55, 2008.

[7] Phong H. Nguyen, Kai Xu, Ashley Wheat, B.L. William Wong, Simon Attfield, and Bob Fields. Sensepath: Understanding the sensemaking process through analytic provenance. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):41–50, 2016.

[8] Phong Hai Nguyen, Kai Xu, and B. L. William Wong. Interaction log and provenance for sensemaking. 2016.

[9] Donghyun Lee, Minkyu Lim, Hosung Park, Yoseb Kang, Jeong-Sik Park, Gil-Jin Jang, and Ji-Hwan Kim. Long short-term memory recurrent neural network-based acoustic model using connectionist temporal classification on a large-scale training corpus. *China Communications*, 14(9):23–31, 2017.

[10] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[11] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[12] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[13] Anthony Gillioz, Jacky Casas, Elena Mugellini, and Omar Abou Khaled. Overview of the transformer-based models for nlp tasks. In *2020 15th Conference on Computer Science and Information Systems (FedCSIS)*, pages 179–183. IEEE, 2020.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[15] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[17] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

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

[19] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[20] Min Zhang and Juntao Li. A commentary of gpt-3 in mit technology review 2021. *Fundamental Research*, 1(6):831–833, 2021.

[21] Anis Koubaa. Gpt-4 vs. gpt-3.5: A concise showdown. 2023.

[22] OpenAI. Gpt-4 technical report, 2023.

[23] Xuanting Chen, Junjie Ye, Can Zu, Nuo Xu, Rui Zheng, Minlong Peng, Jie Zhou, Tao Gui, Qi Zhang, and Xuanjing Huang. How robust is gpt-3.5 to predecessors? a comprehensive study on language understanding tasks. *arXiv preprint arXiv:2303.00293*, 2023.

[24] Qingyu Chen, Jingcheng Du, Yan Hu, Vipina Kuttichi Keloth, Xueqing Peng, Kalpana Raja, Rui Zhang, Zhiyong Lu, and Hua Xu. Large language models in biomedical natural language processing: benchmarks, baselines, and recommendations. *arXiv preprint arXiv:2305.16326*, 2023.

[25] Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, et al. Summary of chatgpt/gpt-4

research and perspective towards the future of large language models. *arXiv preprint arXiv:2304.01852*, 2023.

[26] Maciej Rosoł, Jakub S Gasior, Jonasz Łaba, Kacper Korzeniewski, and Marcel Młyńczak. Evaluation of the performance of gpt-3.5 and gpt-4 on the medical final examination. *medRxiv*, pages 2023–06, 2023.

[27] Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375*, 2023.

[28] Oguzhan Topsakal and Tahir Cetin Akinci. Creating large language model applications utilizing langchain: A primer on developing llm apps fast. In *International Conference on Applied Engineering and Natural Sciences*, volume 1, pages 1050–1056, 2023.

[29] Sebastian Lobentanzer and Julio Saez-Rodriguez. A platform for the biomedical application of large language models. *arXiv preprint arXiv:2305.06488*, 2023.

[30] Rodrigo Pedro, Daniel Castro, Paulo Carreira, and Nuno Santos. From prompt injections to sql injection attacks: How protected is your llm-integrated web application? *arXiv preprint arXiv:2308.01990*, 2023.

[31] Sujay Raghavendra. Introduction to streamlit. In *Beginner's Guide to Streamlit with Python: Build Web-Based Data and Machine Learning Applications*, pages 1–15. Springer, 2022.

[32] JM Nápoles-Duarte, Avratanu Biswas, Mitchell I Parker, JP Palomares-Baez, MA Chávez-Rojo, and LM Rodríguez-Valdez. Stmol: A component for building interactive molecular visualizations within streamlit web-applications. *Frontiers in Molecular Biosciences*, 9:990846, 2022.

[33] Vipul Jain, H Kavitha, and S Mohana Kumar. Credit card fraud detection web application using streamlit and machine learning. In *2022 IEEE International Conference on Data Science and Information System (ICDSIS)*, pages 1–5. IEEE, 2022.

[34] Saurabh Shukla, Arushi Maheshwari, and Prashant Johri. Comparative analysis of ml algorithms & stream lit web application. In *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, pages 175–180. IEEE, 2021.

GitHub URL:https://github.com/xinhaoyan/MSc-Final-Project.git Streamlit Cloude :https://msc-final-project-kh6rnyohvzdwyfgxrgkpiw.streamlit.app/