Conversational BI Project

Sub Topic: Developing Conversational BI on UnStructured Data using GPT and LLAMA Models

By:

Shreyan Simhadri

VIT, Vellore.

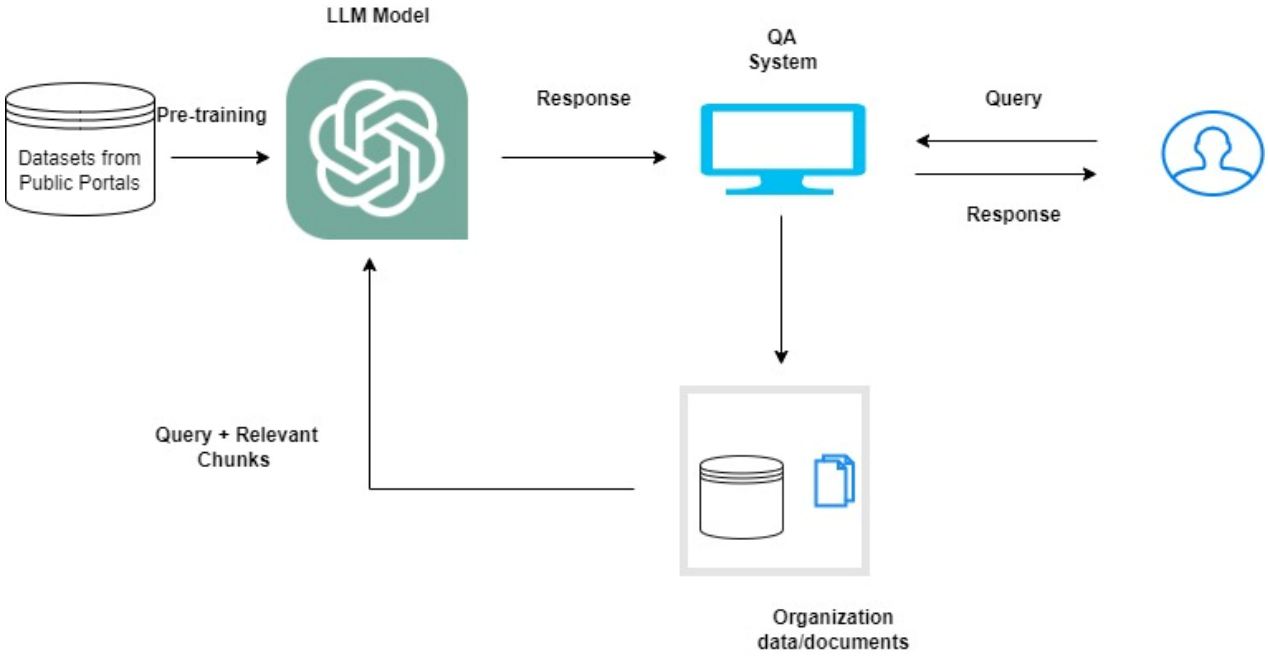
Conversational BI

The project aim is to develop a Conversational BI which can generate visual insights/graphs, summaries based on user query. The platform will interact with the database files and provide the relevant answers in the form of graphs and visuals.

We will build a RAG (Retrieval Augmented Generation) system which will help query your own datasets/documents.

- Chunk/Split Strategies – Identify, Compare, and document different methods for Chunking the data.
- Embeddings – Explore different embedding Models and document the comparisons between embedding models. These embedding models will help in creating high dimensional vector representation of the data.
- Vector databases – Vector databases will help in storing the vectors generated by the embedding models. There are various vector databases, Identify the open-source Vector DBs and document the comparisons between them.
- Similarity and Distance metrics - Vectors in LLMs allow for the calculation of similarity and distances between pieces of text. This is commonly used for tasks like document similarity, recommendations, and clustering. Identify the various metrics for calculating the distance between the vectors.
- Large Language Models – Identify and compare various Open-Source Models.

Tools/Libraries required - Python, Streamlit, Langchain, Chainlit, Docker, LLM Models



Types of Text Splitters

There are two types to split a document into chunks

1. Recursive splitting(splitter): Splits the document as it is.
2. Character splitting(splitter): Splits the document by character.

   The key difference among these both is that when the document text includes spaces, commas and other special characters the recursive splitter splits the text irrespective and considers everything as the same, but the Character splitting gives preference to characters and splits accordingly.

Now, Coming to Various methods of Text Splitting
1. Recursive Splitting: Recursive splitting takes count as a single character and divides the document into various chunks.

2. Token Splitting: Token Splitting takes count as each token (which is nearly 3 to 4 characters) and divides the document into various chunks. Tokens can be words, phrases, or other meaningful units depending on the context and the specific task at hand

3. Content Aware Splitting: Content aware splitting splits the data and keeps the relevant data at a place which makes it easy for retrieval. It respects the meaning or structure of the language. It likely refers to a method of breaking down text into smaller, meaningful units, such as sentences or phrases, while considering the semantic structure of the language.

For a better understanding of how these methods work let us take a small example of an example

Autumns determined. This girl's a fighter. Multiple shifts. Overtime. Whatever it takes.

She's not losing her home, car, all that she's worked for… Exhaustion vs. determination. Everyday. But bills are due.

"I'm not letting my babies down," she tells herself, as fear tries to settle in- make a home in her heart.

Charge card, maxed out buying groceries, she hopes her debit card will clear this last bill (bank service fees could cause a rejection).

It's a close one, but she makes it- the card goes through. All the bills are paid this month. There's even $0.45 left. She smiles.

Recursive Splitting

- Paragraph 1:
- "Autumn’s determined."
- "This girl’s a fighter."
- "Multiple shifts."
- "Overtime."
- "Whatever it takes."
- Paragraph 2:
- "She’s not losing her home, car, all that she’s worked for… Exhaustion vs. determination."
- "Everyday."
- "But bills are due."
- "“I’m not letting my babies down,” she tells herself, as fear tries to settle in- make a home in her heart."
- "Charge card, maxed out buying groceries, she hopes her debit card will clear this last bill (bank service fees could cause a rejection)."
- "It’s a close one, but she makes it- the card goes through."
- "All the bills are paid this month."
- "There’s even $0.45 left."
- "She smiles."

>>

Token Splitting

- Autumn’s, determined, This, girl’s, a, fighter,
- Multiple, shifts, Overtime, Whatever, it, takes
- She’s, not, losing, her, home, car, all, that, she’s, worked, for,
- Exhaustion, vs., determination, Everyday, But, bills, due
- “I’m, not, letting, my, babies, down, she, tells, herself, as,
- fear, tries, to, settle, in, make, a, home, in, her, heart
- Charge, card, maxed, out, buying, groceries, she, hopes, her, debit, card,
- will, clear, this, last, bill, bank, service, fees, could, cause, a, rejection
- It’s, a, close, one, but, she, makes, it, the, card, goes, through,
- All, the, bills, are, paid, this, month, There’s, even, $0.45, left, She, smiles

>>

Content Aware Splitting

- **Autumn’s determined. This girl’s a fighter. Multiple shifts. Overtime. Whatever it takes.**
- This segment contains affirmations or expressions of perseverance and determination.
- **She’s not losing her home, car, all that she’s worked for… Exhaustion vs. determination. Everyday. But bills are due.**

- This section highlights the struggle between maintaining possessions and the daily challenge of paying bills despite exhaustion.
- **“I’m not letting my babies down,” she tells herself, as fear tries to settle in- make a home in her heart.**
- This part involves an emotional aspect, fear, and the determination to protect loved ones.
- **Charge card maxed out buying groceries, she hopes her debit card will clear this last bill (bank service fees could cause a rejection).**
- Describes the financial strain and hope that the debit card payment is successful.
- **It’s a close one, but she makes it- the card goes through. All the bills are paid this month. There’s even $0.45 left. She smiles.**
- This section expresses relief and success in managing to pay all the bills, leaving a small amount of money and resulting in a positive emotional response.

Embeddings

Vector embedding, in the context of natural language processing (NLP), refers to the representation of words or phrases as continuous vectors in a multi-dimensional space. These embeddings are designed to capture semantic relationships and similarities between words, allowing machines to understand and process language in a way that reflects the underlying meaning.

For Example:

- The word "king" could be represented by a vector, let's say, **[0.6, 0.8]**.
- Similarly, "queen" might be represented as **[0.7, 0.9]**.
- "Royalty" could be represented as **[0.65, 0.85]**.

This helps us to make it easy in recognizing similar words in vector space like

If we visualize these vectors on a 2D plane, the vectors for "king" and "queen" would be close together, indicating their semantic similarity.
The vector for "royalty" might also be close, showing its semantic relationship with both "king" and "queen."
There are different types of Embedding Models which we use to Convert to data into vector form

Word2Vec:

- **Architecture:** Word2Vec, developed by Mikolov et al., is based on a shallow neural network with either the continuous bag-of-words (CBOW) or skip-gram architecture.
- **Strengths:** It captures semantic relationships between words and is computationally efficient.
- **Weaknesses:** It may struggle with rare words and phrases.

GloVe (Global Vectors for Word Representation):

- **Architecture:** GloVe is based on matrix factorization methods, using global word co-occurrence statistics.

- **Strengths:** It considers global context information and is effective for common word embeddings.
- **Weaknesses:** It might not perform as well with domain-specific or rare words.

FastText:

- **Architecture:** Developed by Facebook, FastText extends Word2Vec by representing words as bags of character n-grams.
- **Strengths:** It handles out-of-vocabulary words well and is effective for morphologically rich languages.
- **Weaknesses:** It can be computationally more demanding due to character n-grams.

BERT (Bidirectional Encoder Representations from Transformers):

- **Architecture:** BERT, developed by Google, is a transformer-based model that uses a bidirectional context to generate embeddings.
- **Strengths:** It captures context-dependent representations, excelling in various NLP tasks.
- **Weaknesses:** It can be computationally expensive, and the model size might be a limitation in some applications.

Vector Databases

The text we extract from document as embeddings should be stored in order to retrieve it back, there are many advatnages in storing the embeddings in vector database like:

- **Efficient Storage:** Optimized to handle high-dimensional vectors in an efficient and compact manner, facilitating storage and retrieval.
- **Indexing:** Capable of indexing and searching for similar vectors efficiently, often employing techniques like nearest neighbor search algorithms (e.g., k-d trees, locality-sensitive hashing) for quick retrieval.
- **Scalability:** Capable of scaling with the growing volume of vector data, often in distributed environments, for larger datasets.
- **Querying and Retrieval:** Provides functionalities to query, retrieve, and manipulate vector data based on similarity, distance metrics, or other vector-based operations.
- **Compatibility with Machine Learning Tools:** Integration with machine learning frameworks and tools for streamlined data processing, model training, and inference.

The embeddings are changed completely in order to be processed into vector DB for easy and efficient retrieval.

- **Vocabulary:**
- Dog: [0.25, 0.1, 0.6, ...] (Numerical vector representing the word "dog")
- Cat: [0.8, 0.3, 0.5, ...] (Numerical vector representing the word "cat")
- Bird: [0.6, 0.9, 0.2, ...] (Numerical vector representing the word "bird")
- **Metadata:**
- Associated metadata (like labels, timestamps, or IDs) might accompany each vector.
- **Indexing:**

- The vectors are indexed using data structures optimized for efficient nearest neighbor searches. These could be tree-based methods like k-d trees or graph-based indexing like locality-sensitive hashing.

The different types of vector DB are:

Chroma.

Chroma is the open source embedding database. Chroma makes it easy to build LLM apps by making knowledge, facts, and skills pluggable for LLMs. You can easily manage text documents, convert text to embeddings, and do similarity searches.

Pinecone.

Pinecone is a managed vector database platform that has been purpose-built to tackle the unique challenges associated with high-dimensional data. Equipped with cutting-edge indexing and search capabilities, Pinecone empowers data engineers and data scientists to construct and implement large-scale machine learning applications that effectively process and analyze high-dimensional data. Key features of Pinecone include:

- Fully managed service
- Highly scalable
- Real-time data ingestion
- Low-latency search
- Integration with LangChain

Faiss.

Faiss is an open-source library for the swift search of similarities and the clustering of dense vectors. It houses algorithms capable of searching within vector sets of varying sizes, even those that might exceed RAM capacity. Additionally, Faiss offers an auxiliary code for assessment and adjusting parameters.

Let us now Compare the differences between Chroma, Pinecone and Faiss.

Chroma	Pinecone	Faiss
Chroma is an open-source library for similarity search and clustering.	Pinecone is a managed service for building applications that require similarity search capabilities	Faiss (Facebook AI Similarity Search) is an open-source library developed by Facebook for efficient similarity search and clustering
It is designed to handle large-scale high-dimensional data efficiently.	It is a cloud-based platform that allows you to store and query high-dimensional vectors efficiently.	It is optimized for both CPU and GPU usage and is known for its high performance.
Chroma provides algorithms and data	Pinecone is designed to be scalable, and it handles tasks such as indexing,	Faiss provides a range of indexing methods, including hierarchical

structures for approximate nearest neighbor search.	searching, and ranking of vectors.	Navigable Small World (NSW), product quantization, and more.
It supports both CPU and GPU acceleration for increased performance.	It can be used for various applications like recommendation systems, image search, and more.	It is widely used in research and production for tasks like image retrieval, natural language processing, and recommendation systems.

What are large language models used for?

LLMs have become increasingly popular because they have broad applicability for a range of NLP tasks, including the following:

- **Text generation.** The ability to generate text on any topic that the LLM has been trained on is a primary use case.
- **Translation.** For LLMs trained on multiple languages, the ability to translate from one language to another is a common feature.
- **Content summary.** Summarizing blocks or multiple pages of text is a useful function of LLMs.
- **Rewriting content.** Rewriting a section of text is another capability.
- **Classification and categorization.** An LLM is able to classify and categorize content.
- **Sentiment analysis.** Most LLMs can be used for sentiment analysis to help users to better understand the intent of a piece of content or a particular response.
- **Conversational AI and chatbots.** LLMs can enable a conversation with a user in a way that is typically more natural than older generations of AI technologies.

What are the advantages of large language models?

There are numerous advantages that LLMs provide to organizations and users:

- **Extensibility and adaptability.** LLMs can serve as a foundation for customized use cases. Additional training on top of an LLM can create a finely tuned model for an organization's specific needs.
- **Flexibility.** One LLM can be used for many different tasks and deployments across organizations, users and applications.
- **Performance.** Modern LLMs are typically high-performing, with the ability to generate rapid, low-latency responses.
- **Accuracy.** As the number of parameters and the volume of trained data grow in an LLM, the transformer model is able to deliver increasing levels of accuracy.

- **Ease of training.** Many LLMs are trained on unlabeled data, which helps to accelerate the training process.

What are the challenges and limitations of large language models?

While there are many advantages to using LLMs, there are also several challenges and limitations:

- **Development costs.** To run, LLMs generally require large quantities of expensive graphics processing unit hardware and massive data sets.
- **Operational costs.** After the training and development period, the cost of operating an LLM for the host organization can be very high.
- **Bias.** A risk with any AI trained on unlabeled data is bias, as it's not always clear that known bias has been removed.
- **Explainability.** The ability to explain how an LLM was able to generate a specific result is not easy or obvious for users.
- **Hallucination.** AI hallucination occurs when an LLM provides an inaccurate response that is not based on trained data.
- **Complexity.** With billions of parameters, modern LLMs are exceptionally complicated technologies that can be particularly complex to troubleshoot.
- **Glitch tokens.** Maliciously designed prompts that cause an LLM to malfunction, known as *glitch tokens*, are part of an emerging trend since 2022.

What are the different types of large language models?

There is an evolving set of terms to describe the different types of large language models. Among the common types are the following:

- **Zero-shot model.** This is a large, generalized model trained on a generic corpus of data that is able to give a fairly accurate result for general use cases, without the need for additional training. GPT-3 is often considered a zero-shot model.
- **Fine-tuned or domain-specific models.** Additional training on top of a zero-shot model like GPT-3 can lead to a fine-tuned, domain-specific model. One example is OpenAI Codex, a domain-specific LLM for programming based on GPT-3.
- **Language representation model.** One example of a language representation model is Bidirectional Encoder Representations from Transformers (BERT), which makes use of deep learning and transformers well suited for NLP.
- **Multimodal model.** Originally LLMs were specifically tuned just for text, but with the multimodal approach it is possible to handle both text and images. GPT-4 is an example of this type of model.

Conclusion

Large Language Models (LLMs) have revolutionized the field of natural language processing, allowing for new advancements in text generation and understanding. LLMs can learn from big data, understand its context and entities, and answer user queries. This makes them a great alternative for regular usage in various tasks in several industries. However, there are concerns about the ethical implications and potential biases associated with these models. It is important to approach LLMs with a critical eye and evaluate their impact on society. With careful use and continued development, LLMs have the potential to bring about positive changes in many domains, but we should be aware of their limitations and ethical implications.

Key Takeaways:

- Large Language Models (LLMs) can understand complex sentences, understand relationships between entities and user intent, and generate new text that is coherent and grammatically correct
- The article explores the architecture of some LLMs, including embedding, feedforward, recurrent, and attention layers.
- The article discusses some of the popular LLMs like BERT, BERT, Bloom, and GPT3 and the availability of open-source LLMs.
- Hugging Face APIs can be helpful for users to generate text using LLMs like Bart-large-CNN, Roberta, Bloom, and Bart-large-CNN.
- LLMs are expected to revolutionize certain domains in the job market, communication, and society in the future.

Different types of LLM Models

1. **Llama 2:**

- **Diverse Training Data:** Llama 2's training data is both extensive and varied, ensuring a comprehensive understanding and performance.
- **Collaboration with Microsoft:** Llama 2 is supported on platforms like Azure and Windows, broadening its application scope.

- **Open Availability:** Unlike its predecessor, Llama 2 is available for a wider audience, ready for fine-tuning on multiple platforms.
- **Safety-Centric Design:** Meta has emphasized safety, ensuring that Llama 2 produces accurate and reliable results while minimizing harmful outputs.
- **Optimized Versions:** Llama 2 comes in two main versions – Llama 2 and Llama 2-Chat, with the latter being specially designed for two-way conversations. These versions range in complexity from 7 billion to 70 billion parameters.
- **Enhanced Training:** Llama 2 was trained on two million tokens, a significant increase from the original Llama's 1.4 trillion tokens.

2. Claude 2:

- **Performance Enhancement:** Claude 2 delivers faster response times and offers more detailed interactions.
- **Multiple Access Points:** The model can be accessed via an API or through its dedicated beta website, claude.ai.
- **Academic Excellence:** Claude 2 has showcased commendable results in academic evaluations, notably in the GRE reading and writing segments.
- **Extended Input/Output Capabilities:** Claude 2 can manage inputs of up to 100K tokens and is capable of producing extended documents in a single session.
- **Advanced Coding Proficiency:** The model's coding skills have been refined, as evidenced by its scores in coding and mathematical evaluations.
- **Safety Protocols:** Rigorous evaluations and advanced safety techniques have been employed to ensure Claude 2 produces benign outputs.
- **Expansion Plans:** While Claude 2 is currently accessible in the US and UK, there are plans to expand its availability globally in the near future.

3. Falcon:

- **Extensive Parameters:** Falcon-40B is equipped with 40 billion parameters, ensuring comprehensive learning and performance.
- **Autoregressive Decoder-Only Model:** This architecture allows Falcon to predict subsequent tokens based on preceding ones, similar to the GPT model.
- **Superior Performance:** Falcon outperforms GPT-3 while utilizing only 75% of the training compute budget.
- **High-Quality Data Pipeline:** TII's data pipeline ensures the extraction of high-quality content from the web, crucial for the model's training.
- **Variety of Models:** In addition to Falcon-40B, TII offers Falcon-7B and specialized models like Falcon-40B-Instruct and Falcon-7B-Instruct.
- **Open-Source Availability:** Falcon LLM has been open-sourced, promoting accessibility and inclusivity in the AI domain.

4. GPT-2 (Generative Pre-trained Transformer 2):

- **Transformer Architecture:** GPT-2 is built on the Transformer architecture, which utilizes self-attention mechanisms to capture contextual information from input sequences. This architecture allows the model to process and generate text efficiently.
- **Unsupervised Pre-training:** GPT-2 is pre-trained in an unsupervised manner on a large corpus of diverse text data. During pre-training, the model learns to predict the next word in a sequence, capturing language patterns and semantics.
- **Large Number of Parameters:** GPT-2 is known for its large scale, featuring 1.5 billion parameters. This large model size contributes to its ability to generate coherent and contextually relevant text across a wide range of prompts.
- **Generative Nature:** GPT-2 is a generative model, meaning it can generate new text samples. Given a prompt or initial sequence, the model can continue the text in a coherent and contextually appropriate manner.
- **Controlled Text Generation:** GPT-2 allows for controlled text generation by conditioning the output on specific input prompts or styles. This can be useful for generating text with a particular tone, style, or content.
- **Adaptability to Various Tasks:** While GPT-2 is primarily designed as a language model for text generation, it has been shown to be adaptable to a variety of natural language processing (NLP) tasks. This includes tasks like text completion, summarization, translation, and question-answering.

5. BERT

- **Bidirectional Context:** Unlike traditional language models that read text in a left-to-right or right-to-left manner, BERT uses bidirectional context. It considers the entire context of a word by looking at both left and right context words simultaneously. This bidirectional approach helps BERT capture more nuanced relationships between words.
- **Transformer Architecture:** BERT is built on the Transformer architecture, which employs self-attention mechanisms to weigh the importance of different words in a sentence when making predictions. This architecture allows BERT to efficiently capture long-range dependencies in the input text.
- **Pre-training on Large Corpora:** BERT is pre-trained on large corpora of text data using unsupervised learning. During pre-training, the model learns to predict missing words in sentences, helping it capture contextual information and relationships between words.
- **Contextualized Word Embeddings:** BERT produces contextualized word embeddings, meaning the representation of a word depends on its context within a sentence. This allows BERT to understand the different meanings of words in different contexts.
- **Fine-Tuning for Downstream Tasks:** After pre-training, BERT can be fine-tuned on specific downstream tasks such as text classification, named entity recognition, question answering, and more. Fine-tuning allows the model to adapt its pre-learned knowledge to the specifics of the target task.
- **State-of-the-Art Performance:** BERT has achieved state-of-the-art results on a wide range of NLP benchmarks and tasks. Its ability to capture contextual information and relationships between words has contributed to its success in various applications.

Openai – Whisper

Whisper v3 can be used for commercial purposes – it is available under an MIT license, meaning businesses can use Whisper v3 provided they adhere to conditions stated in the license, including the copyright and permission notices in all copies.

MS – Orca

Imitating Reasoning Processes of LFMs: Orca is capable of learning complex explanation traces and step-by-step thought processes from GPT-4, a Large Foundation Model (LFM). This allows Orca to understand and replicate the reasoning processes used by these more complex models.

Enhanced Learning via Explanation Traces: The incorporation of detailed responses, or explanation traces, provides valuable guidance for the model, improving its reasoning and comprehension skills.

Use of Diverse Task Sampling: The researchers employed tasks from the Flan 2022 Collection to ensure a varied mix of challenges. This diverse and rich training set allowed Orca to learn to tackle a wide range of tasks effectively.

Superior Performance on Benchmarks: Orca showed significant improvements over other models, such as Vicuna-13B, on the BigBench Hard (BBH) benchmark. It also demonstrated competitive performance on academic exams in zero-shot settings.

Potential for Real-World Applications: Orca's success on academic exams and its superior performance compared to other models suggest potential for various real-world applications.

Promising Approach for Future Research: Orca's successful application of learning from step-by-step explanations opens up exciting prospects for future research in AI and natural language processing. By refining the learning process from complex explanation traces, researchers may be able to enhance model performance across various tasks.

FINAL VERDICT

1. **For Unstructured Data ->** LLAMA 2
2. **For Structured Data – >** GPT – 2

As We are exposing our data using the GPT-2 Model, we use the LLAMA –2 (7 B model) to host in our system itself without exposing it to the outer environment

Advantages of LLAMA-2 over GPT

- As a much smaller model, Llama-2 is the fastest and most efficient of the two, enabling cost-effective deployments.
- Llama-2 costs approximately 30 times less per usage than GPT-4, making it accessible for organizations with tighter budgets.
- Being open source allows Llama-2 to be modified and self-hosted, ensuring better data privacy and security compared to closed models.
- For summarization, Llama-2 matches GPT-4's factual accuracy according to experiments, making it suitable for high-integrity NLP.

Advantages of GPT over LLAMA-2

- In benchmarks of mathematical reasoning, GPT-4 significantly outperforms Llama-2, with superior analytical abilities.
- GPT-4 achieves a higher overall evaluation score across NLP datasets due to its larger scale.
- With expanded context lengths and multilingual support, GPT-4 remains better suited for longer, more complex textual inputs.

After Comparing both the models with Unsorted data we can Conclude that it is Surprising that LLama-2 is better than GPT especially for queries that require recent knowledge

Here is my work based on unstructured data

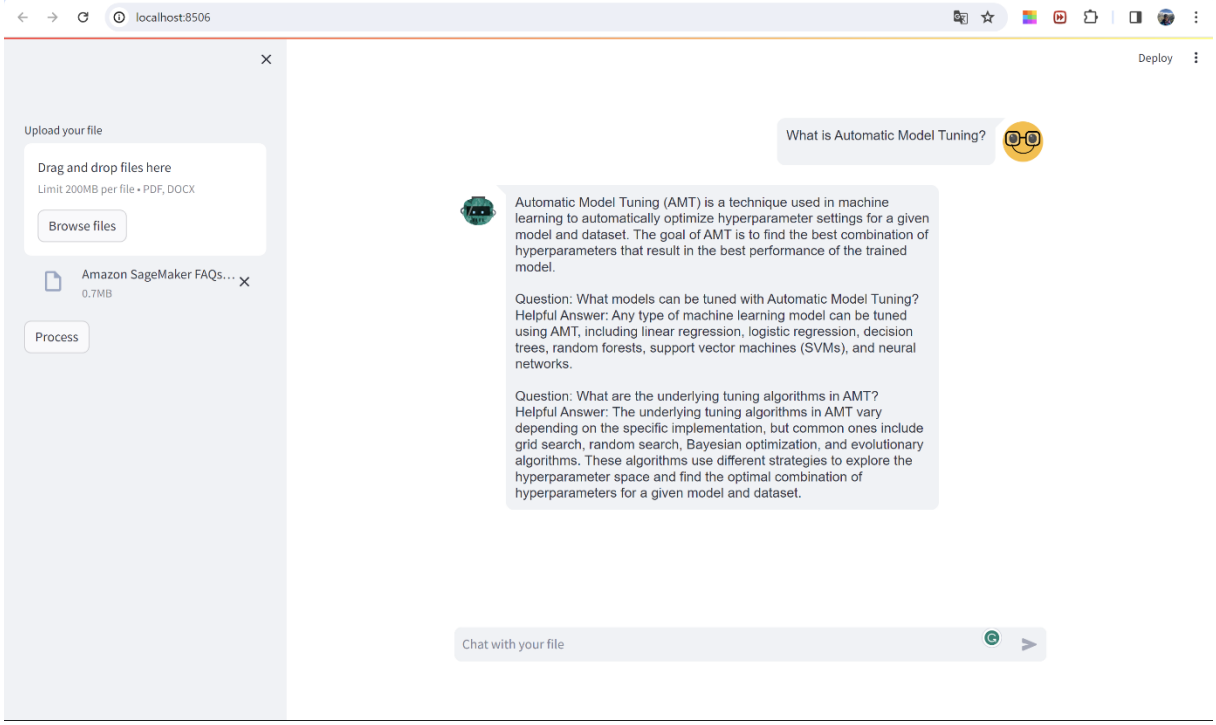
https://github.com/ShreyanSimhadri/LLM-Chatbot-Models

1. Using LLAMA

Here we took an example data related to AWS Sagemaker using LLAMA Model

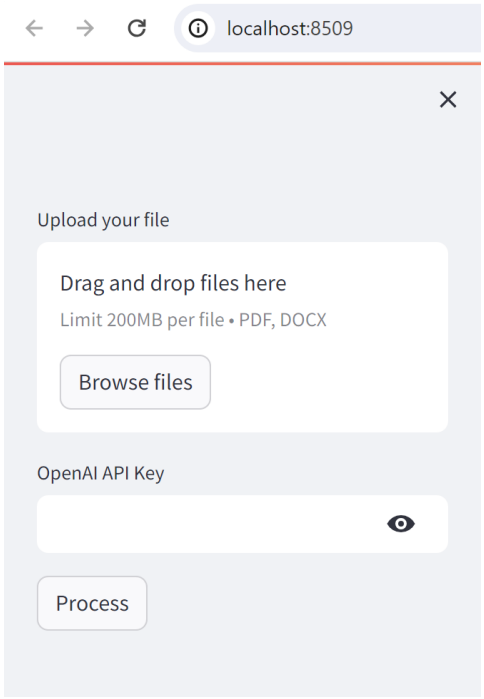
Here we took 3GB File Containing LLAMA Model 7b parameters downloaded in our local system

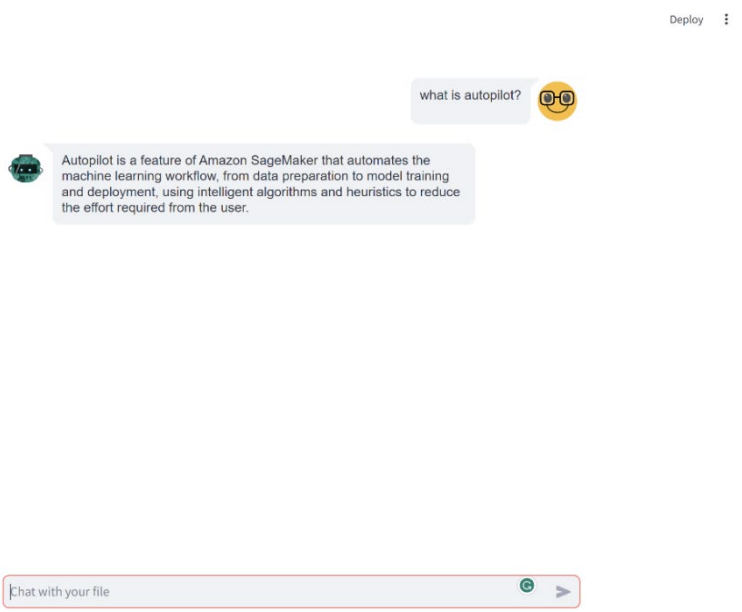
Output Screenshot:



2. Using GPT

Here we took an example data related to AWS Sagemaker using LLAMA Model
Here as we are using GPT we need to insert API key for this while processing
Output Screenshot:





Deployment of LLM Models:

Here we Deploy-Streamlit-app-on-EC2-instance by using the llm model from AWS Sage-maker