Offline Analysis Of Document using LLM Workflow

Offline Document Analyzer

Faiza Iftikhar
iamfaiza@gmail.com
Department Of Computer Science
Institute of Space Technology
Islamabad, Pakistan

Eman Tahir
emantahirist@gmail.com
Department Of Computer Science
Institute of Space Technology
Islamabad, Pakistan

Shumail Inam
shumailinam@gmail.com
Department Of Computer Science
Institute of Space Technology
Islamabad, Pakistan

Faran Mahmood
faran.Mahmood@IST.edu.pk
Department Of Computer Science
Institute of Space Technology
Islamabad, Pakistan

ABSTRACT

This project involves the design and implementation of an Offline Document Analysis system using a Large Language Model (LLM) Workflow. The system operates offline and provides answers to queries related to documents in various formats such as JSON, CSV, TXT, and PDF. Upon uploading these documents, the system generates summaries and answers questions related to the content.

The LLM workflow is facilitated by the implementation of LangChain. The primary objective is to obtain accurate responses to user queries and generate comprehensive document summaries, with support for multiple document uploads. The implementation incorporates advanced embedding techniques, leveraging LLMs for contextual embeddings to enhance query responses and document summarization. Different embedding methods, including Word2Vec, GloVe, TF-IDF, FastText, Doc2Vec, and modern LLMs, have been explored to identify the most suitable approach for question-answering tasks. The system uses an offline model, ensuring efficient and secure processing without reliance on continuous internet access. Integration of vector storage with Chroma DB enables efficient retrieval and similarity search using cosine similarity, ensuring relevant context generation and accurate answer formulation. This project demonstrates significant advancements in offline document processing and highlights its practical applications and benefits in real-world scenarios. Our contributions include the development of an efficient offline document analysis workflow using LLMs, a comprehensive evaluation of various embedding techniques, and the successful deployment of an offline model for document summarization and query response.

KEYWORDS

Large Language Models (LLMs),Document Summarization, Lang Chain, Chroma DB, Cosine Similarity, Embedding Generation Context Generation .

INTRODUCTION

In today's information-rich environment, the efficient management and analysis of documents are crucial. This paper presents a solution for offline document analysis using a Large Language Model (LLM) workflow, capable of handling various document formats such as JSON, CSV, TXT, and PDF. The objective is to produce accurate summaries and answer queries related to these documents while ensuring data security and efficiency through offline functionality. LangChain has been utilized to manage the parsing of documents. Each format is processed using the best available loader to ensure accurate content extraction. The extracted text is then cleaned and prepared for analysis, making it consistent and suitable for further processing.

A key component of this system is the generation of embeddings, which aid in understanding and querying document content. Several embedding techniques were explored, including traditional methods like Word2Vec, GloVe, TF-IDF, FastText, and Doc2Vec. Despite their effectiveness, these methods have limitations in capturing contextual nuances. To address this, modern offline LLMs were also evaluated for their ability to generate more accurate and context-aware embeddings.

The advanced LLMs are used to generate embeddings, which are stored in Chroma DB, a vector database, for efficient retrieval. Cosine similarity is employed to match user queries with relevant document segments, ensuring that the answers provided are accurate and relevant.

A notable feature of this system is its offline capability. Ollama, an open-source tool, was integrated to enable LLMs to run locally, eliminating the need for continuous internet access. This approach ensures data privacy and security, making the solution ideal for environments where data confidentiality is critical.

This paper details the development and implementation of the offline document analysis system. The various embedding techniques evaluated, the use of LangChain for document parsing,

and Chroma DB for efficient query processing are discussed. The contributions highlight the creation of a reliable offline document analysis workflow and its practical applications.

1 LITERATUREREVIEW

Document analysis and retrieval have long been areas of interest within the fields of information retrieval and natural language processing (NLP). Traditional methods, such as TF-IDF (Term FrequencyInverse Document Frequency), have been widely used for text representation and document similarity measurement [11]. TF-IDF captures the importance of words within documents but lacks the ability to understand the semantic meaning and context of the text. To address these limitations, Word2Vec and GloVe (Global Vectors for Word Representation) were introduced, generating word embeddings based on co-occurrence statistics and neural networks [[8], [1]]. While these models improved upon TF-IDF by capturing word similarities and analogies, they still struggled with understanding the context of entire sentences and documents [7]. FastText, an extension of Word2Vec, included subword information, enhancing its ability to handle out-ofvocabulary words but still focusing on word-level embeddings [6]. Doc2Vec extended Word2Vec to generate embeddings for entire documents by considering word order and context [4]. This method improved document similarity tasks, topic modeling, and clustering. However, it still faced challenges in capturing complex semantic nuances compared to more recent transformer-based models. The advent of transformerbased models, such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), marked a significant advancement in NLP [[3], [13]]. These models leverage deep learning architectures to generate context-aware embeddings, excelling in various NLP tasks including text classification, sentiment analysis, and question answering. BERT, with its bidirectional training, captures context from both directions, making it highly effective for understanding text semantics [9]. GPT-3 and GPT-4, developed by OpenAI, further pushed the boundaries by generating high-quality, contextually relevant text and embeddings, though they require substantial computational resources and often depend on internet access [5]. The challenge of deploying such powerful models in offline environments has been addressed by the development of open-source LLMs that can run locally . Models such as Meta Llama, Codestral, Phi-3, Aya, and Mistral provide high performance while being capable of offline operation . Ollama, an open-source tool, facilitates the local deployment of these models, ensuring data privacy and security, crucial for sensitive document analysis tasks. Recent advancements in vector databases, like Chroma DB, have enhanced the efficiency of embedding storage and retrieval [11]. Vector databases support similarity search through techniques such as cosine similarity, enabling fast and accurate matching of user queries with relevant document segments. This paper builds upon these advancements by integrating traditional and modern embedding techniques with offline LLM capabilities, managed through LangChain. The workflow ensures efficient document parsing, embedding generation, and query processing, providing accurate summaries and answers in an offline environment. This approach addresses the limitations of earlier methods and offers a robust solution for secure and efficient document analysis.

2 METHODOLOGY

For Document Handling the document Formats Supported are JSON ,CSV , TXT , PDF etc then implemented Lang Chain to include documents of flexibly linked nature. Every format of document has its own best loader to parse the content in the best manner.[10] For Data Processing and Embedding converted the textual information within provided documents into an set of words then manipulated the text data to make it cleaned, and more appropriate for analysis to be carried out, as well as to make it consistent. For embedding generation Large Language Models (LLMs) are used to generate embeddings for the document text then employed Lang Chain to facilitate the embedding process. For Embeddings while considering models for embedding tasks in natural language processing (NLP) projects, traditional methods like Word2Vec and Glove,TF-IDF, FastText and Doc2Vec while resource-efficient and straightforward, fall short due to their static nature and inability to capture contextual nuances. Then the explored Modern large language models (LLMs) such as GPT-4 and BERT offer significant advancements by generating contextual embeddings and excelling in transfer learning, but they often require substantial computational resources and, in the case of Open AI GPT-4, continuous internet access and paid usage. Opensource models are nearly as accurate as BigScience and are readily available for deployment, although LLaMA and Mistral must work in offline environments; Mistral is better but requires GPU support. T5 by Google plans the training in a one-stop manner with the textto-text training, and RoBERTa offers better pre-training of BERT with similar downstream task performance, but both support offline training though are resource-intensive. Electra enables a proficient pre-training technique and is fathomable with high performance, combining fewer computations than most of its precedents. Last, Falcon developed by Eleuther AI is completely open source, no API limitations, fast and ready for offline use however as most of above it is demanding hardware for its best performance. As shown in above Table 1 there are various techniques for embeddings. For the use case of embedding for question/answering the embedding technique suitable is using LLM. We want to run our LLM locally in our device. For running LLM locally in our device we have install Ollama. Ollama allows open source large language model locally. [2] By research the large larguage models we have explored the most powerful model is "Palm" which is train on 540B parameters but we cannot locally run this model in our machine. Command R+ is train on 104B paramters and we can locally run this model in our machine followed by this also "Qwen" which is train on 110B parameters is 2nd most powerful local model in it. We will get the most accurate

embeddings with the most powerful model which is train on highest number of parameters. Now according to our space, capacity and processing power of our PC we will select which model we can run locally in our machine and after that we will integate the model in our application. After embeddings we use Vector Storage is used to store the generated embeddings in Chroma DB, a vector database, for efficient retrieval and similarity search. In Query Processing and Similarity Search we Embedded user queries using the same LLM used for document embeddings then store query embeddings in Chroma DB for consistency and efficient retrieval. In Similarity Search we applied cosine similarity to perform similarity search between user query embeddings and document embeddings. Retrieved the most relevant document segments based on the similarity scores. For Context Generation and Answer Formulation we used the results of the similarity search to generate context related to the user query. Provided this context to the LLM to ensure the generated answers which are relevant and accurate.

The cosine similarity between two vectors and is given by the formula:

$$\text{cosine\_similarity}(A, B) = \frac{A \cdot B}{\|A\| \, \|B\|} \quad (1)$$

where $A \cdot B$ is the dot product of vectors A and B, and $\|A\|$ and $\|B\|$ are the magnitudes (or lengths) of vectors A and B respectively. Cosine is employed in matching each user query with segmented documents. Here's how it fits into the workflow: Here's how it fits into the workflow: Embedding Generation: For the processing of queries and documents, the Large Language Models (LLMs) are used to map both the queries as well as the documents into the appropriate vector form. Vector Storage: These embeddings are stored in Chroma DB which is a vector database for efficient retrieval of vectors that have been colour coded. Similarity Search: When a submitted query is processed, compare the query embedding with other document embeddings using cosine similarity.[12] For Context Retrieval the extent of similarity between the document segments and the user query can be quantified using the cosine similarity index and selecting the most relevant context among them. In Answer Generation this context is then used by the LLM to pass back accurate and relative answers to the questions asked by the user. Leveraging cosine similarity, it guarantees that more important segments of the documents are included in the process of defining the reply and, by extension. After that, for answer generation, the LLM used the improved model to provide more accurate and relevant responses.

Table 1: Comparison of Embedding Techniques

Embedding Technique	Description	Typical Use Cases	Suitability for Q/A	Reasons
Word2Vec	Word2Vec generates word embeddings by training a neural network on a large corpus of text. word similarity tasks, language modeling.	Not Suitable	Generates embeddings at the word level, which makes it difficult to capture the meaning of entire sentences or documents.	Generates embeddings at theword level, which makes it difficult to capture the meaningmof entire sentences documents.
GloVe	GloVe (Global Vectors for Word Representation) generates word embeddings based on word co- occurrence statistics from a corpus.	Word similarity tasks, word analogy tasks, semantic analysis.	Not Suitable	Similar to Word2Vec, generates word-level embeddings, making it less effective for capturing the context of full sentences or documents.
TF-IDF	TF-IDF (Term FrequencyInverse Document Frequency) generates sparse representations based on word frequency and inverse document fre- quency.	Information retrieval, text classification, document similarity.	Partially Suitable	Captures the importance of words in documents but doesn't capture semantic meaning or context well.
FastText	An extension of Word2Vec that considers subword information and generates embeddings for words and phrases.	Similarity tasks, text classification, language modeling.	Not Suitable	Focuses on word-level embeddings, though it handles out-of-vocabulary words better. Still lacks sentencelevel context.
Doc2Vec	Doc2Vec extends Word2Vec to generate embeddings for entire documents by considering word order and context.	Document similarity, topic modeling, document clustering.	Suitable	Can generate embeddings for entire documents, but might not capture semantic nuances as effectively as transformer-based models like BERT or Sentence-BERT.

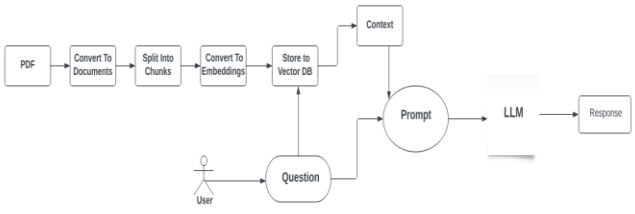

Figure 1: This block diagram shows LLM Conversation Flow

provided context, query and chat history to generate detailed and contextually appropriate answers to user query. In this section also we have to apply the most powerful LLM which we can to generate accurate answers.

techniques and offline functionality, the system processes the documents in various forms and provides accurate summaries as well as answers to the user queries. LangChain for document parsing enables efficient and accurate extraction of the content, while LLM enable comprehension and context analysis. These are achieved by storing embeddings in Chroma DB and use of cosine similarity for query processing. This solution can be used offline, which improves data protection and confidentiality. By incorporating Ollama, LLM can work offline, which is useful in areas where information security is an issue. In general, this research supports the evolution of offline document analysis workflows and showcases their applicability in practice. In this way, the integration of the new advanced LLM with efficient embedding techniques and offline capabilities ensures a safe and effective approach for document processing and analysis.

REFERENCES

[1] Piotr Bojanowski et al. “Enriching Word Vectors with Subword Information”. In: *Transactions of the Association for Computational Linguistics*. Vol. 5. 2017, pp. 135–146.

[2] Alexander Bonia. *Ollama and LangChain: Run LLMs Locally*. 2022. url: https://medium.com/@abonia/ollama-and-langchain-run-llms-locally-900931914a46 (visited on 02/29/2024).

[3] Tom Brown et al. “Language Models are Few-Shot Learners”. In: *arXiv preprint arXiv:2005.14165* (2020).

[4] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 4171–4186.

[5] Jeff Johnson, Matthijs Douze, and Hervé Jégou. “Billion-Scale Similarity Search with GPUs”. In: *IEEE Transactions on Big Data* (2019).

[6] Jey Han Lau and Timothy Baldwin. “An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation”. In: *Proceedings of the 1st Workshop on Representation Learning for NLP*. 2016, pp. 78–86.

[7] Quoc Le and Tomas Mikolov. “Distributed Representations of Sentences and Documents”. In: *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 2014, pp. 1188–1196.

[8] Tomas Mikolov et al. “Distributed Representations of Words and Phrases and their Compositionality”. In: *Advances in Neural Information Processing Systems*. Vol. 26. 2013.

[9] Hieu Pham et al. “Meta Llama: A Diverse Open-Source Set of Large Language Models”. In: *arXiv preprint arXiv:2104.05132* (2021).

[10] Varsha Rainer. *Document Loaders in LangChain*. 2022. url: https://medium.com/@varsha.rainer/document-loaders-in-langchain-7c2db9851123 (visited on 03/09/2024).

[11] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., 1986.

[12] Severalnines. *Vector Similarity Search with PostgreSQL’s pg_vector: A Deep Dive*. 2022. url: https://severalnines.com/blog/vector- similarity- searchwith-postgresqls- pgvector- a- deep-dive/#:~:text=Cosine%20similarity%20compares%20the%20angle,point%20in%20the%20same%20direction (visited on 03/20/2024).

3 CONCLUSION

In conclusion, this research paper provides an entirely unique approach to offline document analysis with the help of modern Large Language Models (LLMs). Due to various embedding

[13] Hugo Touvron et al. “LLaMA: Open and Efficient Foundation Language Models”.
In: *arXiv preprint arXiv:2302.13971* (2023).