RAP-B

1. Introduction

This project aims to build a chat bot - RAP B(ot) that allows the user to upload documents and run queries against them. The two part project involves fine-tuning a Large Language Model (LLM) for sentiment analysis and using Retrieval Augmented Generation (RAG) to enable the user to 'Chat' with their files.

2. Literature and Research Questions (Contributions)

For the RAG architecture we plan to use LLAMA2 [1], an open source large language model that was released by Meta and can be used or further trained for other natural language tasks such as text generation, summarization etc. BERT[2] is another potential candidate to be fine-tuned for the sentiment analysis as despite its simplicity, a bit of fine-tuning would be enough for it to perform an entire range of Natural Language (NL) tasks. In [3], Zhang et al. conducted an extensive study comparing the performance of large language models like ChatGPT, Flan-UL2, and T5 in semantic analysis. Through this project, we hope to answer the following questions:

1. Given the large size of the lyrics, how can the data be segmented or chunked so that the model can ingest them without losing the contextual meaning?

2. Given that V-RAM is necessary for fine-tuning, which quantization methods can we implement such that both practical implementation and precision is balanced?

3. Can a model that is fine-tuned on the German language be effective for English?

4. Would a model that has been fine-tuned on a lyric's dataset work for other questions as well or would it be better to use a general one?

3. Methodology

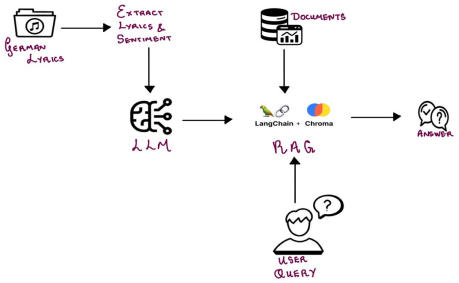
3.1 Dataset, Tools and Technologies

Table 1a shows the dataset, tools and technology that we plan to use in this project. The German dataset contains both a lyrics and sentiment column which would be used to fine-tune the LLMs, and the English dataset would be used for testing.

Dataset	German Rap[a], English Rap
LLMs	BERT, LLAMA2
Frontend	Angular
Backend	Django (REST API)
Database	Chroma / FAISS / DeepLake
Libraries	LangChain, Pandas, Transformers

(a) Dataset, Tools and Technologies

a. https://www.kaggle.com/datasets/mxgra93/german-rap-sentiment/data



(b) RAG Architecture

3.2 Implementation and Testing

Figure 1b shows the architecture of our project. The steps to implement and test the system are as follows:

1. Collect the dataset for training and testing
2. Identify, fine-tune, test and evaluate a LLM(s) with the collected data
3. Chain a vector database to the LLM and create a user interface for the user to upload files ask questions
4. Evaluate whether the RAG architecture works as expected

4. Example Screens and Conclusion

An example of the use we foresee with this work is as shown in Figure 2. The user just has to enter lyrics and ask the bot for the sentiment associated with it, and it would be returned. By the end of our work, we hope to have achieved the following contributions:

1. Combining sentiment analysis with RAG
2. Application of sentiment analysis to RAP lyrics in particular
3. Testing if a model that is familiar with data in one language would work as well for data in another language
4. Testing if a model that is fine-tuned for sentiment analysis on rap lyrics would suffer from 'catastrophic forgetting'.

References

[1] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023.

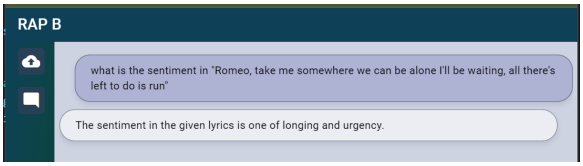


Figure 2: User Interface - Rap B

[2] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019.

[3] Wenxuan Zhang et al. *Sentiment Analysis in the Era of Large Language Models: A Reality Check*. 2023. arXiv: 2305.15005 [cs.CL].