16th April 2024

Task -1 Automatic Speech Recognition

# consdering Factors

Based on the specific demand of the project we can select the most appropriate Speech to Text model. For this purpose, we need to answer few questions such as what the priority for the project in terms of requirements of the text to speech model is do we want the fastest speech to text model, or most accurate speech to text model or is the price a consideration that is the priority. When we can answer these questions, we can go ahead and select the most appropriate model for this project.

## Factors affecting the selection of the model: -

### **Modality:** The number one consideration that we need to make for selecting a suitable Speech to Text model is either we require the product to analyze the data in a batch format or in a real time streaming format is the latter is the case we would require a Speech to Text model which can give us almost instant response as it would truly make it a conversational AI product. But if we require the model to give the output in a batch format, we can select a model which is not that instantaneous but is very precise and accurate in the transcriptions that it generates.

### **Accuracy:** It is the second most important factor to consider according to me as if the accuracy is not good for the speech to text model for our project, then it won’t provide any real value to the customers and the customer experience would be negative.

### **Speed:** If for the commercial purpose we require speed as the key ingredient for the product we can priorities it the most as some commercial applications require an instant output and fast processing of the audio input.

### **Customization:** There are a lot of different accents, vocabulary, or jargons as well as slangs and names of people, places and things that need to be processed accurately while also adapting to a lot of different other factors such as the background noise etc. This factor can be a very crucial consideration when selecting the Speech to Text model as then would we require a model which can be easily trained to understand all this factors if it comes across something that it might be transcribing incorrectly.

# Models consideration

Based on the above listed factors when selecting the appropriate Speech to Text model based on the research, we have OpenAI’s Whisper model as viable option with some modifications, and each modification has its own advantages.

**OpenAI Whisper**

This model has been widely used for research purposes and in various products because of the competitive pricing and it satisfies most of the needs of the industry in terms of generating fairly accurate transcripts in English language with a WER of 13.2 when using a batch processing which is an excellent number and is advantages. But as this model does not support real time audio transcriptions natively that can be a problem which can be tackled in the future using some other methods if we decide to use this model itself we can use Whisper Streaming methodology with Faster-Whisper model to work around this problem by audio chucking.

The issue with this model choice is that it has 230s latency when we use the large model settings with an audio of 1 hour length and for most of the use cases, we will have to use the large model as it is the only choice which generates satisfactory results for transcribing the audio files. I have found couple of studies to overcome this latency issue and I think that might work.

Firstly, we can use batch inference which can provide substantial results [[1]](https://github.com/openai/whisper/discussions/662). I have also found that there is a library named CTranslate2 which is a C++ and Python library which performs efficient inference with Transformer models [[3]](https://github.com/SYSTRAN/faster-whisper).

Further investigation of this library led me to discover a model which **faster-whisper** and it is basically a reimplementation of OpenAI's Whisper model which is using the CTranslate2 library, which is a fast inference engine for Transformer models.

This implementation of CTranslate2 library on the Whisper model is up to 4 times faster than the original OpenAI's Whisper model for the same accuracy while using lesser memory. Also, the efficiency can be further improved with a 8-bit quantization on both CPU and GPU [[4]](https://github.com/SYSTRAN/faster-whisper).

Also, we can further make up for the latency caused in this process when we process the text further in the Text to Text model where I have found yet another way to speed up the performance of LLaMa model [[2]](https://github.com/microsoft/DeepSpeed-MII?tab=readme-ov-file#supported-models).

# Experiments

**Upon experimenting with 3 types of whisper models following were the results and observation I have tested these models on my own voice audio\* which I recorded using my phone:**

* **Original OpenAI Whisper model**: This model performed relatively well and had a very low WER and CER upon testing on personal audio files recorded by me. But the main difficulty when using this specification of model is the latency issue for the audio, I used for this test got a latency of 4.60 seconds.
* **Whisper Streaming model**: This model performed relatively similar with a slight performance improvement in terms of latency it got me a latency of around 3 seconds.
* **Faster-Whisper model**: This model gave me a latency of 0.50 – 0.80 seconds which is the fastest results that I have gotten on the large version of the whisper model and it is very commendable performance which will help in getting a really quick response and in building a very robust and good conversational AI product. The Whisper Streaming model also uses the Faster-Whisper in its backend showcasing that it is so far one of the best reimplementation of whisper model because of the usage of Ctranslate2 which is a very powerful tool and can be used for task 2 as well as this particular library also work on Llama.

*\*The audio file is 1 minute and 31 seconds long and described about whisper model.*

***Note****: For all the testing I have utilized NVIDEA A100 80GBVRAM GPU.*

Based on these observations I can make a conclusion that using the **Faster-Whisper** model will be the most appropriate model for Task-1 of the project as having a quicker transcription will help make up for the further processes that are vital for the success of the overall project and to achieve the goal of making a very realistic conversational AI.

For further investigating the performance and robustness of the model I utilized the LIBRISPEECH dataset readily available in the torchaudio library and tested the model’s performance on 100 samples and recorded the metrics for them you can find them in the attached CSV. The results were similar to the previous test I did using my own audio data which proved the model’s performance is stable.

# Challenges

When trying to perform real time transcription by leveraging my laptops audio I am unable to do so on the university server as I don’t have access to the hardware directly and nor the server has access to my device for direct audio transcription and I don’t have permission to host my code on the local server for making a streamlit UI page which again doesn’t allow me to make a video my implementation and therefore I have just made a video of my code and how well is it performing for now. Also, when trying to load certain libraries for audio processing I am not allowed to do so for some libraries like pyaudio, etc. And when there is some problem in the libraries which can simply be solved by just going into one of the files of the libraries and changing the code of one “.py” file the permission to open that file is denied. “*bash: /opt/conda/lib/python3.10/site-packages/librosa/core/constantq.py: Permission denied*” this are some technical difficulties that I am facing, and your guidance and suggestions will help me navigate this I can use some cloud providers if possible or get an update from the IT team at Queen Mary University of London.

# Future work

I can focus on improving the performance of the transcription even more although I think it can perform in most of the situations. I can do these improvements by taking inspiration from the Whisper Streaming paper [[5]](https://aclanthology.org/2023.ijcnlp-demo.3.pdf). I can also fine tune the model through its various parameters and upon experimentation I can find the most optimal solution.

I also plan to focus on getting the feature of processing audio using computer microphone in real time I have found the proper libraries and preprocessing necessary for this task, and I have tested this on my personal computer using tiny whisper models due to computation constraints.

I plan on creating an UI using streamlit where people can easily either drop an audio file in any format and receive transcription or have flexibility of using their microphone to get an interactive real time transcription. This process is quite straightforward the only constraint I am having in completing this future works is the computational constraints I have mentioned in the above challenges section.