Extracting textual data from different multimedia formats

COP290-Assignment2-Subtask1

Vedant Sameer Talegaonkar

2022CS11603

1 Introduction

In this subtask, I extracted textual data from different file formats using various libraries as follows:

- **Audio and Video:** OpenAI's *whisper*[1] Python library.
- **PDF:** Python library PyMuPDF*(fitz)*[2] for purely textual data, and Python library *pytesseract*[3] for Optical Character Recognition in images in the pdf.

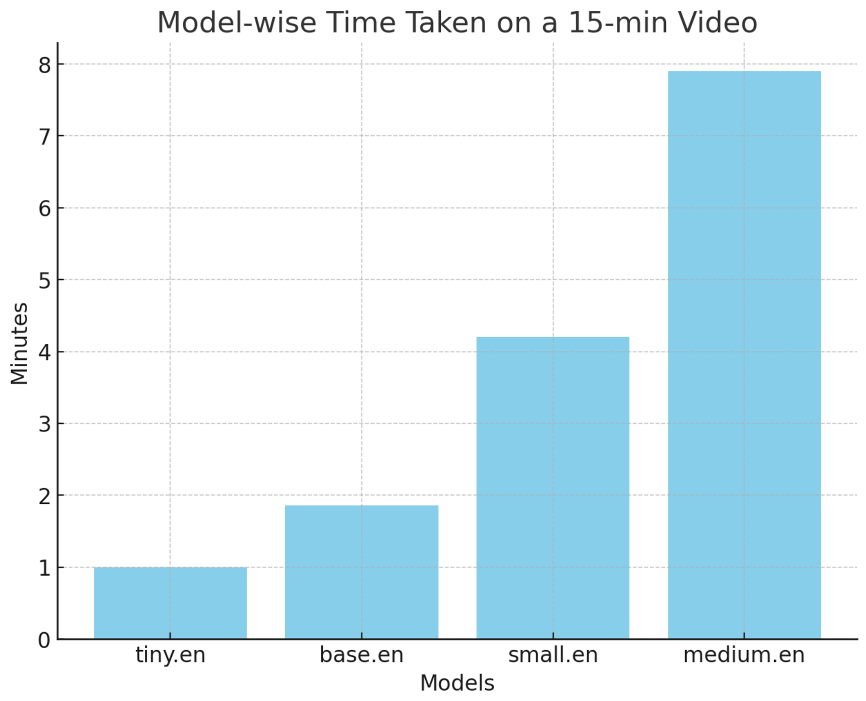
2 Design Choices and Experimentation

2.1 Audio

I tried using Python's *SpeechRecognition* library for extracting transcript from audio files. However, the quality of transcript was poor. Therefore, I had to switch to the OpenAI *whisper*.
Whisper is an open-source pre-trained neural network by OpenAI. It offers 5 levels of speech recognition models - tiny, base, small, medium, large - in increasing order of quality of transcript. I have used the English-only model in the base category - which is **base.en**, as the higher models take a lot of time to run on my system, while the lower model is comparatively inaccurate.

2.2 Video

As the video transcript depends only on its audio, I have used *whisper* model base.en for video transcription. This was also due to the reasons mentioned above.
Here is the analysis of model-wise time taken using *whisper*:



(1)

2.3 PDF

Initially, I used the python library *PyPDF2* for extracting text. However, it did not perform well, including unnecessary linebreaks and bad formatting. Then, I tried the python library *pdfplumber*. However, this turned out to be too slow for my usage.
Finally, I settled on python library *PyMuPDF*. This performed the best out of the three, with fast computation time along with great quality of text.
I am also extracting text data from the images in the pdf using Optical Character Recognition (OCR) in the python library *pytesseract* for OCR and *Pillow* for image handling. This has been done to handle scanned pdfs/pdfs with a lot of images, which would otherwise be ignored.

3 Issues Faced

- In the OCR incorporated in PDF text extraction, I faced an issue involving the format of images in the pdf. Eg- if the image format was .jpeg, the Pillow library was not compatible with this format, therefore I stuck to the approach of skipping the image (as most of the images encountered were of .png format).
- In the *whisper* library, the medium and large models are extremely computationally heavy, therefore, I could not try them on my system. The small model performs well for short videos/audios, but is pretty slow for large videos - therefore, I had to stick to using the base.en model.

4 Results on test files

Here is the link to the test files folder: Test files

4.1 Audio

I tested on two samples - one 3 min long, and one 10 min long. Any model above base.en in *whisper* was taking too much time, therefore, I stuck to using **base.en** for all files.

4.2 Video

I tested on videos of varying length - under 1 min, 3 min and then 15 min. **base.en** proved to be the most viable option for all of them considering the time taken to process and quality of text.

4.3 PDF

I tested my code on Courses of Study, and it took around 20 seconds to run, which is good considering the size of the document. The quality of text received was satisfactory, with the headings being properly shown, and the text formatting being readable. Since there are not many images in the COS with proper text, the image text extraction did not perform that well. I tested on a completely scanned pdf - first using only plain text extraction, which led to empty transcript - and then including image text extraction - which gave great results.

References

[1] *OpenAI Whisper*. `https://github.com/openai/whisper`.

[2] *PyMuPDF*. `https://pymupdf.readthedocs.io/en/latest/`.

[3] *PyTesseract*. `https://pypi.org/project/pytesseract/`.