PDF TO AUDIOBOOK CONVERSION USING MACHINE LEARNING

Saksham Doda[1], Roopam Garg[2], Dr. Neha Agrawal[3]

[1,2]Student, Maharaja Agrasen Institute of Technology, Dept. of IT, Delhi, India
[3]Associate Professor, Maharaja Agrasen Institute of Technology, Dept. of IT, Delhi, India

Abstract

Audiobooks *are audio recordings of a book's text that one can listen to instead of reading. "Unabridged" refers to a reading of the entire text, whereas "abridged" refers to a reading of a condensed form. It is beneficial to everyone who is always devout. It is just not possible to purchase and store them in a home bookshelf. It can also be a terrific way to unwind your eyes after a long day of staring at digital devices. They are used by some to save time and money. For instance, one can keep up with books and multitask. It can not only solve the problems for youth but can also be very beneficial tool for visually impaired person. The ability to transform any material into an audiobook is a true gift to society. Our technology can be put to use in the development of such tools.*

Keywords: *Audiobook, Google Cloud Services, Machine Learning, gTTS, PDF*

I. INTRODUCTION

Audiobooks are becoming an admired and challenging resource for creating synthetic expressive speech. The expressivity variation in audiobooks imposes various challenges, in the sense that although it is rich, it is back-breaking to handle. It is difficult to use all audiobook data in the standard unit selection technique unless the data has been previously categorized or gathered according to voice style, emotion, impersonated characters, or, in general, according to some level of expressivity.

As text extraction is a climacteric process for digitising physical text, which primarily consists of important research papers, for further downstream processes such as converting research papers into audio books, more efficient data annotations, and many more connotation processes execution in work. While critical, the procedure is more complex than others because it relies on a significant number of variables such as image variance, text style, text orientation, alignment, low contrast, noise, and complex background structure, among others.

As a result, the project's major goal is to aid in the resolution of common text-to-speech issues and the development of a simplified method for converting PDFs into more relevant and comprehensive audiobooks that handle data more efficiently.

II. TECHNOLOGY USED

- **Python** - Python is a high-level, general-purpose programming language that is interpreted. The use of considerable indentation in its design philosophy emphasizes code readability. Its language elements and object-oriented approach are aimed at assisting programmers in writing clear, logical code for both small and large-scale projects.

- **Tesseract OCR -** Optical Character Recognition (OCR) is an acronym for optical character recognition. Recognizing text within images, such as scanned papers and photos, is a common technology. OCR technology converts almost any image containing written text (typed, handwritten, or printed) into machine-readable text data. Converting printed paper is probably the most well-known application of OCR. Tesseract is an optical character recognition (OCR) engine that extracts text from a printed text image or a scanned document.

- **Machine Learning** – Machine Learning is the study of computer algorithms that can learn and develop on their own with experience and data. It is considered to be a component of artificial intelligence.

- **gTTS** - Google Text-to-Speech is the abbreviation for Google Text-to-Speech. It's a Python library and command-line tool for interacting with Google's text-to-speech API. It can be used to save spoken mp3 data to a file, create a file-like object (byte string) for further audio modification, or simply pre-generate Google Translate TTS request URLs to feed to another software.

III. METHODOLOGY

We build our PDF-to-audiobook converter in three main steps:

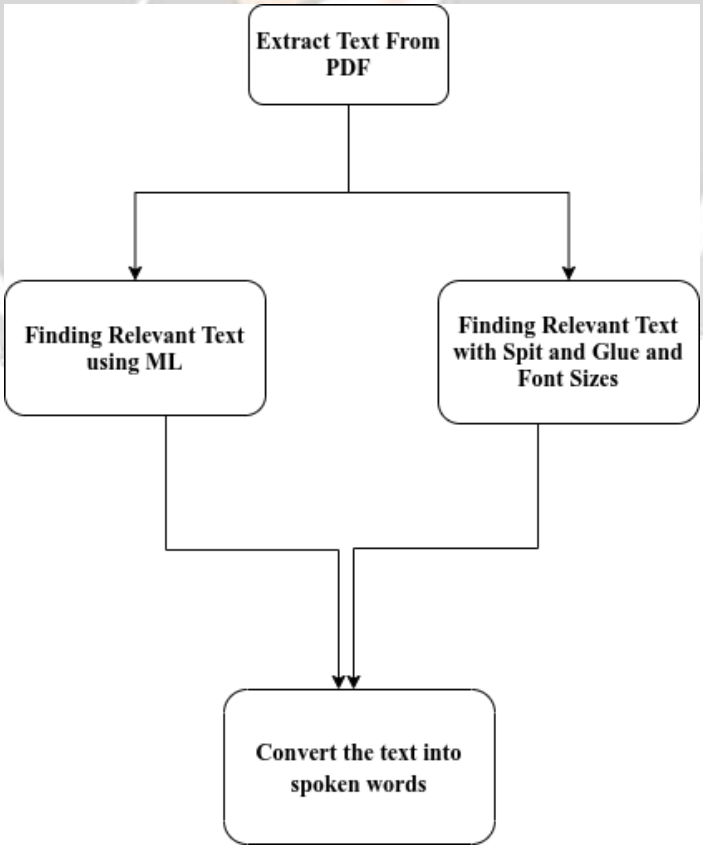

Fig. 5. Flow diagram of Approach

1) Extract text from PDFs (or images)

A PDF document can contain a variety of items, including pictures, vector graphics, and images. Text extraction is a necessary step for every pdf in order to continue processing the data, but it is a time-consuming and difficult work. Calamari is an open-source text recognition engine that was used to extract the text. Calamari is an OCR Engine written in Python 3 that is based on OCRopy and Kraken and is designed to be simple to use from the command line while also being flexible enough to be integrated and changed from other Python scripts.

The document is then given through the Google Cloud Vision AI API, which allows developers to easily add vision detection features such as picture labelling, face and landmark identification, optical character recognition (OCR), and explicit content tagging into their apps. The Vision API gives the page's raw text as well as the (x,y) position of each character. After that, we'll have all of the raw text, which will contain things like image captions, page numbers, document footers, and other things that aren't necessary and shouldn't be in the audiobook. As a result, we'll decide whether portions of raw text should be included in the audiobook in the next step.

2) Decide which parts of the text to include in the audiobook:

The major goal of our project will be to reduce the noise in the PDF in order to discover the vital information. Page numbers, reference numbers, headers, and footers are the most common sources of noise. In an audiobook, these noises are less beneficial.

Techniques to find relevant text:

i) Finding Relevant Text with Machine Learning:

When we look at a research paper, it's probably easy for us to gloss over the irrelevant bits just by noting the layout: titles are large and bolded; captions are small; body text is medium-sized and centered on the page.

We can also train a machine learning model to achieve this using spatial information about the layout of the text on the page. We offer the model a variety of body text, header text, and other types of text in the hopes that it will learn to recognise them.

The Google Cloud Vision API not only gives the page's content, but also its layout. It divides text into chunks (pages, blocks, paragraphs, words, and characters) and provides their page location.
In particular, for each word, it returns a bounding box that looks like this:

```
"boundingBox": {
    "normalizedVertices": [
                {"x": 0.9248292,"y": 0.06006006},
                {"x": 0.9384966,"y": 0.06006006},
            {"x": 0.9384966,"y": 0.067567565},
            {"x": 0.9248292,"y": 0.067567565}
      ]
}
```

The bounding box above describes where a word is located on the page, and how large it is. We can use this data to train a model.

We'll develop a set of features that describe each block of text: What was the total number of characters in the section of text? What size was it, and where on the page did it appear? What was the aspect ratio of the text box (a narrow box, for example, may just be a sidebar)?
After that, we'll gather and label a large number of documents, and then use Google Cloud AutoML Tables to train a machine learning model. It's a model-building tool that doesn't require any coding.

ii) Finding Relevant Text with Spit and Glue and Font Sizes:

We reasoned that by examining font size, we could learn a lot. The title of a document, for example, is most likely written in the highest text size. Meanwhile, body text is the most common text in a document. Using those observations, we will use this heuristic:

- Calculate the font size for all words.
- Compute the most common font size. Label every bit of text in that font size"body".
- Compute the largest font size. Label every bit of text in that font size as "title".

After assessing the feasibility and efficiency of both options, we will decide which strategy to employ to discover the relevant text from the extracted raw text and then proceed with it.

3) Convert the text into spoken words

The extracted text from the is then fed into a TTS Engine, which generates synthetic speech to turn the textual input into spoken output. Synthetic speech was generated using gTTS (Google Text-to-Speech). Google Text-to-Speech is powered by WaveNet, which uses machine learning to generate speech and then recreates human speech using data from a database of human speech. The final result includes voices with subtleties like accents and lip smacks, making it sound more natural and reducing the gap with human performance by 70%.

IV. CONCLUSION

Increased learning benefits were shown in a study of audiobook technology and its impact on reading comprehension and enjoyment. Research on literacy strategies for struggling adult readers is both essential and necessary. The audiobook intervention yielded such promising effects that it is suggested that more research in this area be conducted.

This project does an excellent job reading simple PDF text files. It enables visually challenged and time-pressed persons to quickly access information. Given the widespread use of audiobooks in literacy and library programs around the world, the success of this research effort is remarkable. However, if the text contains complex mathematical equations, this software will be unable to understand the equations as a human would. As a result, the project is suitable for simple text but not scientific publications, as it will struggle to understand complex equations. We want to improve our model by expanding its processing capabilities for files with complex equations and making it more user-friendly.

V. ACKNOWLEDGMENT

I would like to thank my mentor Dr. Neha Agrawal who guided me in doing this project. She provided me with invaluable advice and helped me in difficult periods. Her motivation and continuous support contributed tremendously to the successful completion of the project.

VI. REFERENCES

[1] Seaboyer J, Barnett T. New perspectives on reading and writing across the disciplines. HigherEducation Research Development, 2019, 38(1):1-10.

[2] J. Xu, W. Ding and H. Zhao, "Based on Improved Edge Detection Algorithm for English TextExtraction and Restoration From Color Images," in IEEE Sensors Journal, vol. 20, no. 20, pp. 11951-11958, 15 Oct.15, 2020, doi: 10.1109/JSEN.2020.2964939.

[3] R. Mittal and A. Garg, "Text extraction using OCR: A Systematic Review," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA),Coimbatore, India, 2020, pp. 357-362, doi: 10.1109/ICIRCA48905.2020.9183326.

[4] T.Rubesh Kumar, C.Purnima "Assistive System for Product Label Detection with Voice Output For Blind Users" International Journal of Research in Engineering Advanced Technology2014.

[5] Bo Li and Heiga Zen, "Multi-language multi-speaker acoustic modeling for LSTM-RNN based statistical parametric speech synthesis.," in INTERSPEECH, 2016.

[6] Mithe, R., Indalkar, S. and Divekar, N. (2013) 'Optical character recognition', International Journal of Recent Technology and Engineering (IJRTE), Vol. 2, pp.72–75.

[7] Rakshit, S. and Basu, S. (2010) Development of a Multi-User Handwriting Recognition System Using Tesseract Open Source OCR Engine, arXiv preprint arXiv:1003.5886.4.

[8] Kumar S, R Gupta, et al. Text extraction and document image segmentation using matched wavelets and MRF model, IEEE Trans Image Process, August 2007; 16:2117– 2128.

[9] M. Kumar, Y.C. Kim and G.S. Lee, Text Detection using Multilayer Separation in Real Scene Images, 10th IEEE International Conference on Computer and Information Technology, 2010, pp. 1413-1417.