2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)

Note-ing Hill: A Note-Making Application

Vishnu Anand
Department of Computer Science
PES University
Bengaluru, India
vishnuanand2000@gmail.com

Chinmaya Ravi
Department of Computer science
PES University
Bengaluru, India
chinmayaravi@gmail.com

Aniket Acharya
Department of Computer Science
PES University
Bengaluru, India
aniket.ga8@gmail.com

Sahithya Papireddy
Department of Computer Science
PES University
Bengaluru, India
sahithya.papi@gmail.com

Prajwala T.R.
Department of Computer Science
PES University
Bengaluru, India
prajwalatr@pes.edu

2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE) | 978-1-6654-8316-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICDCECE53908.2022.9793320

***Abstract*—This paper provides an insight into the approaches taken for the development of the application - 'Note-ing Hill' that aims to optimize the task of taking down notes in a classroom and make it more efficient. This application involves the following components: speech-to-text transcription, image to text conversion and text summarization, collectively supported by a cloud back-end(AWS). For speech-to-text transcription AWS' transcribe feature was employed while image-to-text conversion (more formally, OCR - optical character recognition) was performed with the help of the Tesseract engine and finally, for text summarization, the extractive approach of the Text rank algorithm was used.**

Keywords—Optical Character Recognition (OCR), Automatic Speech Recognition (ASR), Extractive text summarization, Textrank, Pagerank, Pytesseract.

I. Introduction

This paper delves into the approach behind building a product - an Android application named 'Note-ing Hill' that primarily aims to serve problems faced in the classroom and during real-time learning.

The application combines three major functionalities: Speech-to-text transcription, image-to-text conversion and text summarization, each providing utility in the form of: transcribing audio (of a professor in class) into notes, converting images (of textbook pages) into text and finally summarizing these lengthy notes into concise summaries, respectively, that all prove to be immensely useful, especially for a student.

Thus a two-and-a-half-hour college lecture can be recorded and transcribed into written text, supplemented by pictures of textbook pages that can be converted into text documents and finally, the text summarization algorithm can be employed collectively to convert that text into concise, neat summaries.

The paper is divided into six sections: A literature survey regarding the individual components of the application, Current methods available, Methodology used for the implementation of the application, Results and Discussion, Conclusion and References.

II. Literature Survey

This section gives a brief summary into the different literature that referred to during the course of developing the application. This section is subdivided based on core concepts and the associated paper is summarized under the concept topic.

A. Text Summarization

Usha Manjari, Syed Rousha, Dasi Sumanth and Sirisha Devi in their paper, "Extractive Text Summarization from Web pages using Selenium and TF-IDF algorithm"[4], have demonstrated extractive text summarization from web pages using Selenium for scraping the data and TF-IDF for the summarization. This approach attempts to save the user the hassle of going through various web pages manually when they query a particular topic. In their approach, the user first enters a query to perform the search operation. Based on this query, the most popular web pages are obtained from a search engine like google and the data from these websites are automatically scraped using Selenium. The data obtained will be summarized using the TF-IDF (Term Frequency - Inverse Document Frequency) approach - initially the data is pre-processed by tokenizing the text and then the TF and the IDF values for each word in a paragraph is calculated. Their respective formulae are given below

TF term = (Frequency of the term in the document) / (Number of terms in the document)

IDF term = loge (Total number of documents /Number of documents containing the term)

The TF-IDF score is then calculated by multiplying a term's TF score with its corresponding IDF score. A sentence is now scored using this TF-IDF score:

Sentence score = (Sum of all TF-IDF values in a sentence/ Number of words in that sentence)

A threshold can then be set and all sentences which have a sentence score greater than the threshold are displayed, thus giving the user a concise summary from multiple web pages.

B. OCR – Optical Character Recognition

Sahil Thakare, Ajay Kamble, Vishal Thengne and U. R. Kamble have demonstrated Optical Character Recognition in their paper, "Document Segmentation and Language Translation Using Tesseract-OCR"[13]. Optical Character Recognition deals with the detection of textual lines, words and symbols in any document. Tesseract first tries to determine the position of text in the document. It tries to apply pattern recognition techniques to correctly identify a given character. It tries to detect fixed pitch and proportional text. The features of a character are compared to a character that is closely related.

Tesseract OCR can be used with documents that are consistent with character fonts and have a clear image. After

978-1-6654-8316-2/22/$31.00 ©2022 IEEE

the initial phase of inputting the image, Tesseract tries to detect blobs of text regions. After blob detection, a line finder detects lines by looking at vertically overlapping adjacent characters over a line. Then, the fixed pitch detector checks for character layout and performs word segmentation according to fixed pitches. The words are identified in two passes. In the first pass, successfully identified words are submitted to an adaptive classifier. The adaptive classifier uses the data as training data. Words that were not successfully identified in the first pass are parsed again, but this time, with the help of the training data available in the adaptive classifier.

C. Speech-to-text Transcription

A detailed insight into the implementation of speech recognition systems is presented by Bhuvaneshwari Jolad and RajashriKhanai in the paper - 'The art of speech recognition: a review' [2].

The entire speech recognition process itself can be broken down into three major steps:

Speech pre-processing where noise is filtered from the audio signal and distinct words are extracted from it; Feature extraction which involves representing the speech signal and encoding it; Classification that employs a mixture of AI algorithms in order to perform speech recognition.

There are several techniques to go about the feature extraction stage:

Principal component analysis and Independent component analysis, both of which are non-linear methods with the latter being an extension of the former, Zero crossing detection, Linear predictive coding - a speaker-specific method that breaks down when it comes to evaluating words with identical vowel pronunciation, Perceptual linear prediction and Relative spectral filtering - both of which function well for audio inputs with considerable background noise, Mel frequency Cepstral Coefficient - a popular method for feature extraction that does not perform very well when it has to process audio with background noise, in which case, the Relative spectral filtering method works better, Wavelet transform which can differentiate between the voiced and unvoiced parts of speech, Gaussian Mixture model, Hidden Markov model, Dynamic Time warping, Vector quantization, Neural networks and conventional neural networks. Similarly, there's also the classification stage which can be implemented in several ways: The K-Nearest neighbour classifier and Naive Bayes Classifier are simple algorithms but have a small scope. The support vector machine method is more robust but proves to be slow for large data sets. Artificial Neural networks show good performance, however, computation speed of the network comes down with a large number of hidden layers.

III. CURRENT METHODS.

A. Text Summarization

There are many approaches to text summarization available, which can all be broadly categorized as either extractive or abstractive [1][10].

Presented below is an overview of some of the popular and widely-used models and algorithms:

Under extractive approaches there are graph based approaches like LexRank and Textrank, that employ Google's popular page rank algorithm, in order to rank sentences to be included in the summary, Latent Semantic Analysis - an unsupervised learning algorithm that works on the principle of distributional hypothesis[8], LuhnSummariser which ranks sentences based on TF-IDF (Term Frequency-Inverse Document Frequency)[3][8], KL-Sum algorithm - a greedy optimization approach that works on decreasing the "KL-divergence criteria" in order to ensure that summary and input document are semantically similar[15].

Under abstractive approaches, there are models such as the T-5 transformer, which comes in 5 sizes and is an encoder-decoder model[14] that converts all tasks into a text-to-text form; the BART model which is of seq2seq architecture[6][9] and employs a bidirectional encoder and a left-to-right decoder as well as the GPT models - GPT-1, proposing the novel concept of training a generative language model with unlabeled data and then fine-tuning it to a specific task, GPT-2 and GPT-3, each improving upon its predecessor in terms of their language-modeling capabilities and the XLM transformer, an improved version of the BERT model that serves as a cross-lingual classification model that also improves upon language models.

B. Image to text conversion

There are a number of services and applications that have optical character recognition as a core principle such as OpenCV, Google MLKit Library and Tesseract-OCR, to name a few. Factors such as feasibility, ease of use and reliability of the method were used to narrow down on a specific method that can be used to implement OCR in the Note-ing Hill application:

The most popular method is to use the Google MLKit Library for Text Recognition in images. This is provided as a paid service API to which an application sends images and receives recognized text back. The main drawback of this service is that it is paid. Although free tiers are available, they were not producing accurate results.

Tesseract-OCR is another open source option. A python version of Tesseract-OCR, named "Pytesseract" can be used. Having a separate library that is modular is very beneficial. It can essentially be used as a plug and play module.

Hence, after considering all the pros and cons of the aforementioned methods, based on modularity, ease of use and feasibility, Pytesseract is the path chosen to implement OCR for our project.

C. Speech to text transcription

There are several tools [7] and software available in the market that perform speech recognition. Some of them run over their own unique speech recognition software like Otter.ai that uses its own software - Otter, to perform live speech to text transcriptions while most others employ one or the other of the following widely popular APIs or core technologies:

Google Cloud's speech to text API that works on Google's deep learning technologies - This tool has the added benefit of providing the user with several machine learning models to choose from that are suited for transcription of audio from specific domains. IBM's Watson speech to text feature whose significance lies in its robust AI that can understand context well. However, it falters in its limited language options. AWS' Transcribe that boasts of features ranging from customization of the output text to safety and privacy measures such as vocabulary filtering and redaction of sensitive information in the content.

D. Cloud Technologies

A vast number of cloud technologies are available [5] coupled with a large number of public cloud vendors to choose from, with the market leaders being Google Cloud Platform (GCP), Amazon Web Services (AWS) and Microsoft Azure.

Google's GCP suffers the drawbacks of being a latecomer to the cloud market and does not offer as many services as AWS. There is lesser control over the virtual machines as well. However, its Kubernetes container orchestration system makes it very useful for containerized applications.

Microsoft's Azure provides a great variety of virtual machines that run different versions of Windows - Microsoft's proprietary operating system. The choice of Linux machines is very limited. Azure requires users to have considerably more platform expertise compared to its competitors.

Amazon's AWS has the advantage of maturity, being the first major public cloud vendor. With over 200 services, AWS also has support for large enterprises - such as Netflix. It is highly flexible with a very cost-effective pricing model. It also allows for greater control over security and scalability.

IV. Methodology

A. Text Summarization

Textrank has been chosen as the extractive text summarization approach for the Note-ing Hill application. Textrank is an algorithm based on the popular Pagerank algorithm [11][15].

This approach has been presented in the paper, "TextRank algorithm by exploiting Wikipedia for short text keywords extraction" by Wengen Li and Jiabao Zhao [15]. In this paper, weights have been assigned to words by calculating their TF-IDF. For the Note-ing Hill application, word embedding has been used to give weight to the sentences, this has been extended to sentences. This happens to be a fairly common approach to perform text summarization using Textrank.

After preprocessing, the sentences are vectorized using GloVe (Global Vectors). A 100 dimension vector model and the word vectors are used to obtain the sentence vector value for the whole sentence.

To get the relationship between sentences, cosine similarity is used, as proposed by the paper. The cosine similarity is calculated for a sentence, with every other sentence and is converted into a similarity matrix where, similarity[i][j] represents the similarity between sentence i and j (similarity[i][i] is taken as 0).

The similarity matrix is then converted into a weighted directed graph, where each node represents a sentence and the weight of each edge is the similarity between the two sentences connected by the edge.

Traditionally, TextRank is a graph based algorithm, pw herein a node is connected to another node through an edge, the former casts a vote on the latter. So the importance of a node depends on the number of votes it has received and the importance of the nodes that have voted for it [12].

The Pagerank of a node 'p' is given by:

$$PR(p) = (1-d) + d\sum_{i=1}^{n} \frac{PR(T_i)}{C(T_i)} \quad (1)$$

PageRank is run on the cosine similarity graph obtained. 'd' is the Damping factor (usually taken as 0.85), the term inside the summation is the PageRank value that a page 'Ti' contributes to page p. In our approach, we have used the PageRank function from the network x library in python that takes a graph as an input and outputs the PageRank scores of every node. The number of incoming edges and the number of outgoing edges for every node is the same (Total number of sentences - 1, as there is no self-edge), hence, PageRank for every node will depend on the cosine similarities of every edge connected to it. Thus, the algorithm is modified to a weighted PageRank algorithm. A sentence becomes more important if it has a high cosine value and thus, a high semantic relationship with many other sentences. Sentences having a high score can then be included in the summary.

The students can choose their preferred summary length according to their use case, for example, based on the length or pattern of their examination.

B. Image-to-text conversion

As mentioned in the previous section, Tesseract OCR is the method proposed because of the following reasons:

- Tesseract-OCR is open source
- Modular code can be written with PyTesseract
- Image manipulation is easier
- Tesseract is more accurate than OpenCV and the free tier of the Google MLKit Text Recognition API.

C. Speech-to-text transcription

Amazon Transcribe was chosen as the appropriate base software, over other software, for several reasons:

The application is hosted on AWS's cloud that serves as a back-end. Hence, using Amazon's Transcribe Services can help maintain consistency throughout the implementation. Further, Amazon's Transcribe provided a good balance between the cost plan and features required. It also has an Indian-English option that makes it best suited for the targeted demographic of the Note-ing Hill application. Additionally, Amazon Transcribe showed good accuracy and demonstrated the use of punctuation throughout the text.

D. Cloud Technologies

The AWS cloud platform provided by Amazon has been chosen to host the back end for the application. The services utilised are described below:

- AWS Amplify - It provides a single point of access for the backend of the Android application, connecting it to the other AWS services.
- AWS S3 (Simple Storage Service) - An S3 storage bucket is used to store the lecture audio recordings, images, and summarized text files.
- AWS EC2 (Elastic Compute Cloud) - An EC2 Ubuntu Linux virtual machine instance is used to run the text summarization program as well as the image-to-text conversion program.

- AWS Lambda - This service provides server-less computing capabilities. The Lambda code is triggered whenever an object is uploaded to the S3 bucket. Based on the file type of the object, the corresponding program code is executed on the EC2 instance.
- AWS Transcribe - It takes the lecture audio recordings and converts them into text files for summarization.

V. Results And Discussion

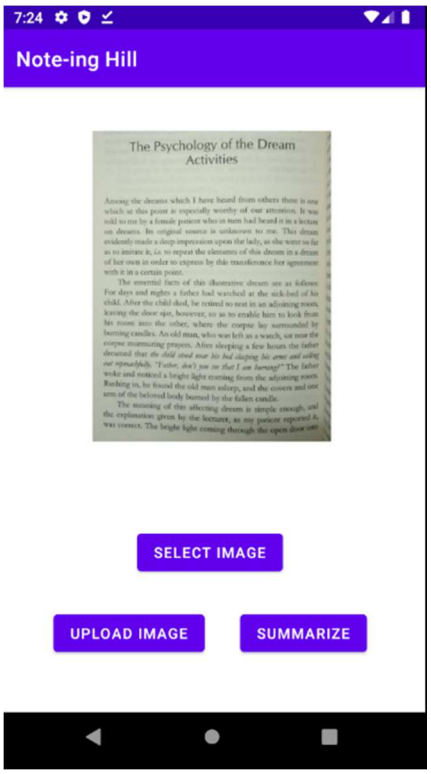

Fig. 1. Image Selection (Image-to-text)

```
The Psychology of the Dream
Activities

Among the dreams which I have heard from others there is one
which at this point is especially worthy of our attention. It was
told to me by a female patient who in turn had heard it in a lecture
on dreams. Its original source is unknown to me. This dream
evidently made a deep impression upon the lady, as she went so far
as to imitate it, Ze. to repeat the elements of this dream in a dream
of her own in order to express by this transference her agreement
with it in a certain point.

The essential facts of this illustrative dream are as follows:
For days and nights a father had watched at the sick-bed of his
child. After the child died, he retired to rest in an adjoining room,
leaving the door ajar, however, so as to enable him to look from
his room into the other, where the corpse lay surrounded by
burning candles. An old man, who was left as a watch, sat near the
corpse murmuring prayers. After sleeping a few hours the father
dreamed that the child stood near his bed clasping his arms and calling
out reproachfully, “Father, don’t you see that 1 am burning?” The father
woke and noticed a bright light coming from the adjoining ba
Rushing in, he found the old man asleep, and the covers 3
arm of the beloved body ‘

```

Fig. 2. Recognized Characters from the image (image-to-text)

```
The Psychology of the Dream
Activities

Among the dreams which I have heard from others there is one
which at this point is especially worthy of our attention.
It was
told to me by a female patient who in turn had heard it in a lecture
on dreams.
This dream
evidently made a deep impression upon the lady, as she went so far
as to imitate it, Ze.
The essential facts of this illustrative dream are as follows:
For days and nights a father had watched at the sick-bed of his
child.
After sleeping a few hours the father
dreamed that the child stood near his bed clasping his arms and calling
out reproachfully, “Father, don’t you see that 1 am burning?” The father
woke and noticed a bright light coming from the adjoining ba
Rushing in, he found the old man asleep, and the covers 3
arm of the beloved body
```

Fig. 3. Summary of the text recognized via image-to-text (Text-summarization)

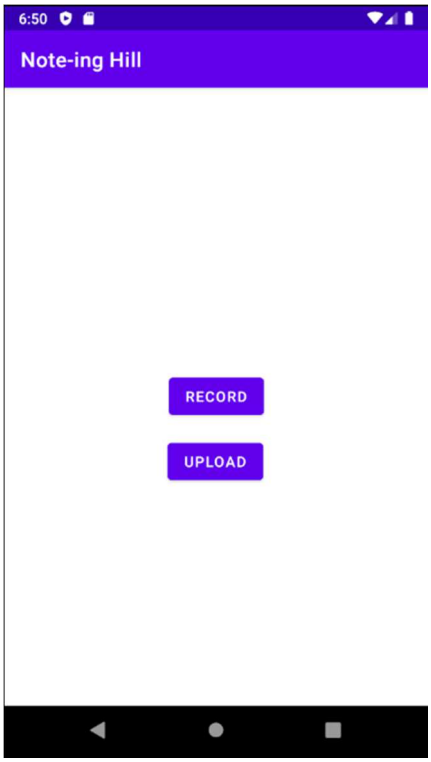

Fig. 4. Audio Recording Activity within the Android app (Speech-to-text)

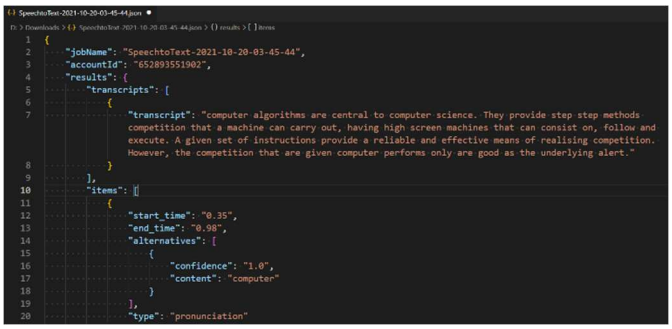

Fig. 5. JSON file of the recognized characters from audio (Speech-to-text)

The above figures 1-5 demonstrate the working of the application according to the implementation discussed in the Methodology section. As shown in the pictures that demonstrate the transcription of a sample text, a fairly good amount of parsing is achieved for image-to-text. Furthermore, if the original input sample text is compared to the summary produced at the end, it can be noticed that the main content of the input text has been encapsulated well in the summary.

Thus, the various methods and technologies used as part of the Note-ing Hill application's implementation have yielded satisfactory results for the purpose and scope of this application.

VI. LIMITATIONS

For the speech-to-text transcription module to function efficiently, the speech needs to be clear and loud. In addition to this, AWS Transcribe struggles to recognize technical terms with ease which is a drawback, considering that our application is primarily aimed for the education domain and lecturers tend to use technical terms while explaining lessons.

For the OCR module, a picture with clean segmentation of the foreground text from the background needs to be provided and the module is not yet capable of recognizing handwriting or interpreting diagrams.

As far as text summarization is concerned, there is scope to improve the effectiveness of the summaries produced.

VII. FUTURE SCOPE

As discussed in the Limitations section, there are several avenues of further development for the Note-ing Hill application such as extending the OCR module's functionality to interpret diagrams and handwriting, adding voice-enabled commands to the speech-to-text transcription module to provide flexibility in transcribing and including an in-built text editor functionality within the application to make any necessary updates.

VIII. CONCLUSION

The android application “Note-ing Hill” encompasses the domains of speech - to - text transcription, image - to - text conversion (OCR) and text summarization. It will streamline the process of making notes so that teachers can provide notes to students with ease and students can cope with their courses at their own pace.

REFERENCES

[1] A. T. Al-Taani, "Automatic text summarization approaches," 2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS), 2017, pp. 93-94, doi: 10.1109/ICTUS.2017.8285983.

[2] B. Jolad and R. Khanai, "An Art of Speech Recognition: A Review," 2019 2nd International Conference on Signal Processing and Communication (ICSPC), 2019, pp. 31-35, doi: 10.1109/ICSPC46172.2019.8976733.

[3] J. N. Madhuri and R. Ganesh Kumar, "Extractive Text Summarization Using Sentence Ranking," 2019 International Conference on Data Science and Communication (IconDSC), 2019, pp. 1-3, doi: 10.1109/IconDSC.2019.8817040.

[4] K. U. Manjari, S. Rousha, D. Sumanth and J. Sirisha Devi, "Extractive Text Summarization from Web pages using Selenium and TF-IDF algorithm," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), 2020, pp. 648-652, doi: 10.1109/ICOEI48184.2020.9142938.

[5] Kumar, Sandeep Tyagi, Manan Khanna, Abhirup Fore, Vivudh. (2018). A Survey of Mobile Computation Offloading: Applications, Approaches and Challenges. 51-58. 10.1109/ICACCE.2018.8441740.

[6] M. Ihori, A. Takashima and R. Masumura, "Large-Context Pointer-Generator Networks for Spoken-to-Written Style Conversion," ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 8189-8193, doi: 10.1109/ICASSP40776.2020.9053930.

[7] M. Ravanelli, T. Parcollet and Y. Bengio, "The Pytorch-kaldi Speech Recognition Toolkit,"ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 6465-6469, doi: 10.1109/ICASSP.2019.8683713

[8] N. Andhale and L. A. Bewoor, "An overview of Text Summarization techniques," 2016 International Conference on Computing Communication Control and automation (ICCUBEA), 2016, pp. 1-7, doi: 10.1109/ICCUBEA.2016.7860024.

[9] P. R. Dedhia, H. P. Pachgade, A. P. Malani, N. Raul and M. Naik, "Study on Abstractive Text Summarization Techniques," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1-8, doi: 10.1109/ic-ETITE47903.2020.087.

[10] S. Biswas, R. Rautray, R. Dash and R. Dash, "Text Summarization: A Review," 2018 2nd International Conference on Data Science and Business Analytics (ICDSBA), 2018, pp. 231-235, doi: 10.1109/ICDSBA.2018.00048.

[11] S. R. K. Harinatha, B. T. Tasara and N. N. Qomariyah, "Evaluating Extractive Summarization Techniques on News Articles," 2021 International Seminar on Intelligent Technology and Its Applications (ISITIA), 2021, pp. 88-94, doi: 10.1109/ISITIA52817.2021.9502230.

[12] S. R. Rahimi, A. T. Mozhdehi and M. Abdolahi, "An overview on extractive text summarization," 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), 2017, pp. 0054-0062, doi: 10.1109/KBEI.2017.8324874.

[13] Thakare, Sahil Kamble, Ajay Thengne, Vishal Kamble, U.R.. (2018). Document Segmentation and Language Translation Using Tesseract-OCR. 148-151. 10.1109/ICIINFS.2018.8721372.

[14] Tomer, M., Kumar, M. Improving Text Summarization using Ensembled Approach based on Fuzzy with LSTM. Arab J Sci Eng 45, 10743–10754 (2020). https://doi.org/10.1007/s13369-020-04827-6

[15] W. Li and J. Zhao, "TextRank Algorithm by Exploiting Wikipedia for Short Text Keywords Extraction," 2016 3rd International Conference on Information Science and Control Engineering (ICISCE), 2016, pp. 683-686, doi: 10.1109/ICISCE.2016.151.