Machine Learning in OCR Technology: Performance Analysis of Different OCR Methods for Slide-to-Text Conversion in Lecture Videos

Geeta S Hukkeri, R H Goudar, Prashant Janagond, Pooja S Patil
Department of CSE, VTU
Belagavi, India

***Abstract*—A significant percentage of a lecture video's content shown is text. Video text can therefore be a crucial source for automated video indexing. Researchers have recognised printed and handwritten text extracted from pictures using a variety of machine learning techniques and tools before digitising it. A machine learning technology called optical character recognition (OCR) enables us to recognise and retrieve text information from documents, converting it into searchable and editable data. This study primarily focuses on text extraction from lecture slides using Google Cloud Vision (GCV), Tesseract, Abbyy Finereader, and Transym OCR and compares the results to develop a lecture video indexing scheme for the non-linear steering in lecture videos to watch only the interesting points of topics. We have taken a total of 438 key-frames in 10 categories from seven different lecture videos that range in length. First, binary and greyscale versions of the input colour images are created. Before using the OCR APIs, the frames are additionally preprocessed to improve the image quality. The recognition accuracy demonstrated that the GCV OCR performs effectively, saving computing time by collecting image text with the highest accuracy of other tools, 96.7 percent.**

Keywords*—Video lectures; keyframes; Google cloud vision (GCV); Tesseract; Abbyy Finereader; Transym; text extraction

I. INTRODUCTION

A branch of machine learning known as optical character recognition (OCR) is focused on identifying characters in visuals such as scanned papers, printed books, or photographs. Despite being a promising technology, there are currently no OCR solutions that can reliably recognise every type of text. Machines can directly handle texts found in the current world thanks to optical character recognition [13]. Education, banking, government, and medical sectors are just a few of the industries where OCR is used. The pre-processed image is fed into the OCR Engine, which then extracts the text that has been written on it. Due to the different written and printed text formats, modern OCR methods use deep learning to increase accuracy. The issue of text recognition can be solved using a variety of conventional deep learning techniques. The most well-known ones include YOLO [1, 2], SSD [2], Mask RCNN [3], and Faster RCNN [2]. These designs may be trained to do character recognition and are essentially entity detectors. Region-based detectors use algorithms like Faster RCNN and Mask RCNN. This implies that the method first scans the image for objects (text) before classifying them (characters). It is slower but more accurate because of this two-step approach. Single Shot Detector (SSD) algorithms like YOLO and SSD simultaneously scan the items and classify them. They are quicker because of the single step procedure, but they do poorly with smaller items, like text in our example. These systems are trained on any of the aforementioned datasets, and the trained systems can be used to anticipate or identify the text in any given image. The goal of qualified neural network (NN) rule generation has spurred a variety of research efforts. The primary classification method for such algorithms is in the manner in which they generate rules. The decomposition method compares each hidden and production node separately, and a pattern is derived from it for precise word detection from images. In a feed-forward NN, each neuron's output is quantified as:

$$R_j = \left(\left(\sum_i W_{ij} \times A_i\right) + \vartheta_j\right) \quad (1)$$

where $A_i = \frac{1}{1+e^{-ax}}$

here, A is the level of activation of neuron i, W_{ij} represents the weight of the relationship between neuron i and j, and is the level of activation of neuron j that controls the gradient of the sigmoid. The breakdown method's most important feature is that almost all of neurons in the NN have either 0 or 1 activations. Binary inputs trigger this in the hidden layer's neurons.

Numerous artificial intelligence scholars have attempted to address the issue of OCR difficulty in order to develop effective OCR systems able to operate in an accurate and timely manner since the advent of computerised systems [28–30]. Even though there are a variety of OCR techniques and toolkits now accessible in the literature, we will be comparing four popular OCR toolkits: Google Cloud Vision (GCV) OCR [24], Tesseract [25], ABBYY FineReader [26], and Transym [27].

Due to the enormous amount of data that deep learning demands for model training, businesses like Google have an advantage in achieving promising outcomes with their OCR services. The specifics of Google Vision OCR are covered in this paper. Using a straightforward REST API interface, the GCV API [7] constructs highly complicated machine learning models focused on image recognition. It has a wide range of image recognition abilities. In this paper, we've concentrated on the OCR module, which scans an image for text before parsing it into data for our computers to use.

A. Objectives

The following are the objectives of this study:

- Data Acquisition by extracting key-frames from lecture videos
- Pre-process the raw input dataset to improve the image quality
- Appy OCR engines to extract text from the key-frames
- Compare the performance OCR engines to decide the best OCR

II. Literature Survey

Deep learning is used in computer vision to build NNs that direct image analysis and evaluation [23]. The OCR methods were mechanical machines, not computers, that could recognise characters at first, but the performance was extremely slow, and the results were less accurate. Although OCR is not a recent issue, its roots can be seen in methods used before the development of computers [12]. OCR has been applied in a wide range of fields. The Transym and Tesseract OCR technologies, for instance, were used by Patel and Patel to analyse car licence plates [17].

The paper [18] used the GCV API to analyse images in another scenario involving an autonomous vehicle to increase the accuracy of object identification and give tough-environment autonomous robots the capacity to recognise objects. Additionally, many industries employ this system to speed up data entry and decrease human error when removing information from document management systems [19], [20]. Additionally, such innovation has been used more and more in smart systems, cloud computing, IoT, and robots. Examples include IoT-based car verification systems [22] and road sign text interpretation [21].

On text in floor plan pictures, conventional and deep learning text detection techniques were contrasted [14]. Four approaches were compared in the study: EAST, Maximally Stable Extremal Regions (MSER), Connectionist Text Proposal Network (CTPN), Stroke Width Transform (SWT), Tesseract, and a normal image processing methodology are the first four options. The last option combines all four of the first three options. Extra sub images were employed for the CTPN approach at the border since CTPN had trouble reading text that was near to the picture borders [14]. The combined technique produces an output that depends on voting by comparing the outcomes from all three previous methods against one another. All approaches to combining particular text boxes into a single text item underwent post processing. Initially, the text was categorised according to the rules. Next, room characteristics were compared to a dictionary of acceptable terms, and the nearest keyword was substituted according to edit distance and term frequency. The proposed approaches were tested on datasets with different levels of quality. The noisy and low quality images were demonstrated to have substantially reduced efficiency with the CTPN approach. The combination technique had the best accuracy on the poor quality images, while the EAST approach seemed to have the greatest recall and F1-score. The efficiency of the detected text was not thoroughly examined, and none of the suggested algorithms could recognise slanted or curving text items. However, it was reported that Tesseract did not make correct estimates on the low resolution pictures.

For image analysis, the GCV API was utilised [8]. Their effort locates and recognises printed text hidden within images, as well as particular items and faces inside images. The adaptability of the GCV API to input noise is assessed in the paper. In particular, when noise is applied to a group of images, the API would be unable to identify the appropriate text or object since, when the noise is cleared, the output is equivalent to the original image. Noise filtering is available for the GCV API. A model that enables users to hear the image's main message in their own language has been proposed in [9]. Text is first taken from the picture and afterwards transformed into the person's native language speech. After being captured by the camera, the image is converted to text by the OCR engine. The gTTS is then used to translate text into speech [9]. A system that interprets words from a taken image has been suggested. Tesseract OCR is used to extract text from digital documents, and the text is then converted to voice. In order to reduce noise, the first acquired image is first transformed to grayscale. After using thresholding, the image is transformed to a binary format, cropped, imported into tesseract OCR for word recognition, and outputted as a text file that can be used as an input for E-speak to generate audio [10]. As of right now, any language's text can be manually entered and converted to any other language as necessary. A whole text book's images cannot be translated from one language to another. Some mobile applications that attempted to convert the above exhibited significant faults. The current system [11], which uses classic OCR, is unable to distinguish text from blurry or poor resolution, blurriness, high noise, and distorted images. The final product is distorted by the image noise. Consequently, consumers experience a challenge with comprehension.

This study primarily focuses on text extraction from lecture slides using Google Cloud Vision (GCV), Tesseract, Abbyy Finereader, and Transym OCR and compares the results to develop a lecture video indexing scheme for the non-linear steering in lecture videos to watch only the interesting points of topics. The dataset is total of 438 key-frames in 10 categories from seven different lecture videos that range in length. First, binary and greyscale versions of the input colour images are created. Before using the OCR APIs, the frames are additionally preprocessed to improve the image quality. The recognition accuracy demonstrated that the GCV OCR performs effectively, saving computing time by collecting image text with the highest accuracy of other tools.

III. Steps Involved in OCR

OCR is a programme that converts text into an appropriate machine-readable format [18, 19]. OCR technology is often used in businesses for automation and processing of written receipts [20]. Researchers now have access to a wide collection of electronic texts that can be analysed using just a few keywords thanks to the OCR technique. Fig. 1 depicts the general OCR approach for text extraction:

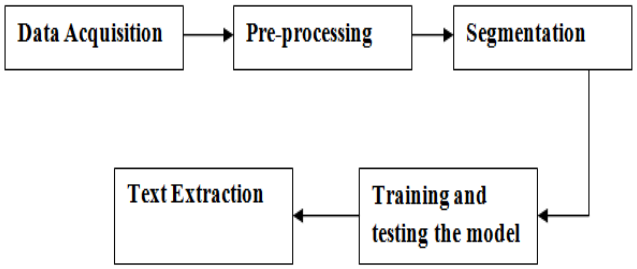

Fig. 1. General Text Extraction using OCR Model.

A. Data Acquisition

The data is a picture of text with straightforward or intricate layouts or backdrops in a scene or document from nature. We can get the text's visual representation via digital camera and handheld scanner [15]. There are several different types of text image databases that can be used for study. They are used to establish standards for processing speed, accuracy, and storage. A few of the datasets available for text extraction is given in [16].

B. Pre-Processing

Before using the OCR method, the raw input dataset must be cleaned up in this step to improve the image quality. The input image must be turned to grayscale and gaussian blur. The 1-dimensional and 2- dimensional Gaussian formula is given below in equations 2 and 3, respectively.

$$\text{GB (i)}=\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{i^2}{2\sigma^2}} \quad (2)$$

$$\text{GB (i, j)} = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{i^2+j^2}{2\sigma^2}} \quad (3)$$

where i and j are the horizontal and vertical axis's distance from the origin respectively, and σ is the Gaussian distribution's standard deviation.

C. Segmentation

The pre-processed images are divided into several sections during segmentation. This comprises scanning an image for clusters of pixels that contain character-containing elements; each of these elements has a class applied to it. Any thresholding method must be used in order to allow for additional analysis. In general, using the right settings makes adaptive thresholding operate best. The segmentation procedure is carried out as follows:

$$S_\sigma(I[l_m, l_n]) = \sqrt{\sum_{I=l_i}^{m} prbden(i) + \left(T - l_i\left(I[l_i, l_j]\right)\right)^2 \times p_l^{I[l_i,l_j]}} + \theta \quad (4)$$

where $prbden$ (probability density) is,

$$prbden(i) = \frac{Histogram\,(i)}{mn} + threshold\ + intensity\,(i) \quad (5)$$

for i=0,1,…..n-1

A feature for an immediate layer's adaptive pixel set is calculated as follows:

$$o_i = \mu p_{i(i)} + \delta(i,j) \quad for\ i = 1,2,\ldots\ldots\ldots m \quad (6)$$

$$o_i = \mu q_{i(j)} - 2(x) + \delta(i,j) \quad for\ j = 3,4,\ldots\ldots n \quad (7)$$

here, μ is the layer importance taken into account when organising the layers in a sequential manner for precise and distinctive text recognition. The created single layer has a fixed total and estimates the result as the sum of all the pixels which make up a set. A fresh layer is produced as:

$$o_i = \frac{\sum \overline{w}(i)h(i)+\theta}{\sum \overline{w}(i)} + \delta(j,k)f_i \qquad for\ k = 1,2,\ldots\ldots m-n \quad (8)$$

D. Training and Testing the Model

The crucial OCR phase is model training. Numerous hyperparameters are engaged in this situation. These have either been generated from the training data or have default values set. Following their definition, a model that creates a generic picture -> text modelling for the data processes the data in the training event. The below Fig. 2 depicts the training and testing phase of the model.

On the provided image, feature extraction is carried out using FEL to produce a feature map. CRGL uses a 3×3 hole convolutional and the anchor procedure to create basic areas in an image by clearing dublicate features. CASL uses Soft-Nonmaximum Suppression (NMS) to get the preliminary areas in the image. A Region of Interest (RoI) can be obtained from the image by using the ROI pooling method in TDL. The training model is established as:

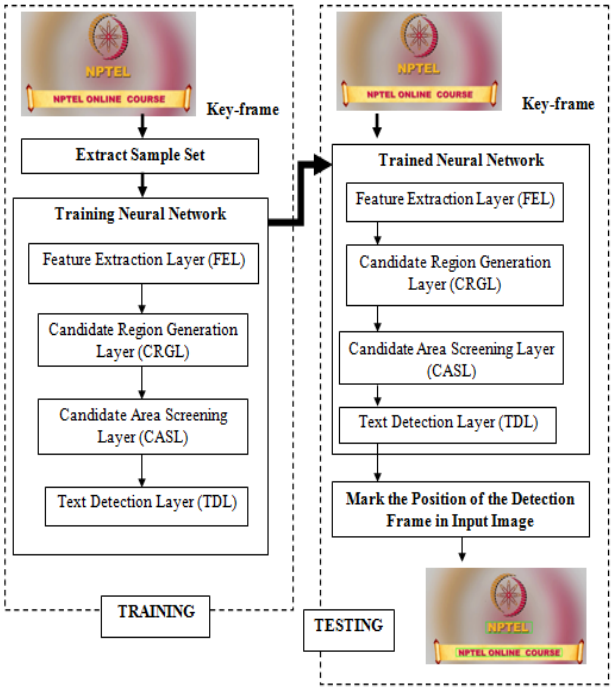

Fig. 2. Training and Testing Phase.

$$TR_1(B,A) = \sum_{i=0}^{m}\sum_{j=0}^{n}\left[\begin{array}{c}\sqrt{\frac{\mu_A^2(a_y)+\mu_B^2(b_y)}{2}} \\ +\sqrt{\frac{1}{m-1}\sum_{i=1}^{n}\left|p_{i-\mu}\right|^2} \\ +\sqrt{\frac{(1-\mu_A(a_y))^2+(1-\mu_B(b_y))^2}{2}}+\end{array}\right] \quad (9)$$

Since NN is used, the hidden layers are taken into account for precise text extraction. Each hidden layer's input is:

$$H(I(i,j)) = I_{i,j} + \sum F_i * W \quad (10)$$

where W and F are the weights between hidden layer and input, and the hidden layer's bias value respectively. As a result, the output of each hidden layer is determined using:

$$o(I(i,j)) = \frac{\sum_{j=1}^{n}(A_{ij}^{M}+B_{ij}^{N})x_j}{\sum_{i=1}^{m}(A_{ij}^{M}+B_{ij}^{N})} \quad (11)$$

E. Text Extraction

To increase the model's ability to extract text accurately, an analysis step is taken after processing through first four steps. The text that was retrieved from the image is given by the following pixels:

$$T(X,Y) = \sum_{(i,j)\in F_s, i\in W, j\in T} o(i) + W_i + \frac{(i*j)(i,j)}{(H_{i,j(P,Q)})} \quad (12)$$

An interface utilising Google OCR technology has been developed in order to provide users with a simple and practical method of text extraction from images. Additionally, this will automate a few processes through the use of the Google OCR engine. The goal of this work is to extract text from English-language lecture slides. The user can choose a language and start the text extraction process. For the purposes of OCR, the regions are separated into unoccupied and occupied regions. Following that, a machine learning model is used to scan the data before a number of processes, including area segmentation and extraction, creating the necessary line images for line segmentation inputs, ground truth output, and more.

IV. Proposed OCR in Slide-to-Text (STT) Conversion

With an emphasis on image recognition, the GCV API transforms extremely complicated machine learning models into a straightforward REST API interface. We concentrate on the OCR module in this work. A Python script was used to construct the Fig. 3 workflow in Tensorflow.

- We have considered seven different lecture videos (machine learning, network, DBMS, Algorithms, two cryptography, and data science for engineers) of variying duration.
- We have first extracted the key-frames (images) from each lecture videos [total 438 images of 10 categories, including: 1) Digital Images, 2) Machine-written characters, 3) Hand-written characters, 4) Machine-written digits, 5) Hand-written digits, 6) Multi-oriented text strings, 7) Black and white images, 8) Noisy images, 9) Skewed images, and 10) Blurred images].
- The obtained images are transformed from the colour to grayscale and binary images. The pre-processing procedures (sharpening, contrast adjustment, and brightness adjustment) are also used to improve the image quality before applying the OCR APIs.
- The processed images are then uploaded to Google Cloud Storage (GCS). Vision API and background processes are started by a GCS event to Create a transcription of the GCS-stored image.
- The converted images are yet again saved in GCS for use in the future. The Natural Language API is used to extract entities from the converted images. The tool initially segments the image's structure to determine where the text is located. The OCR module then does a text recognition on the proper area to generate the text after detecting the general location.
- In a post-processing step, errors are finally fixed by running the data through a language model. The convolutional neural network (CNN) used to do all of this merely connects each neuron to a portion of the neurons in each layer. CNN is designed to mimic the hierarchical organisation of our visual system in terms of object (characters) recognition.

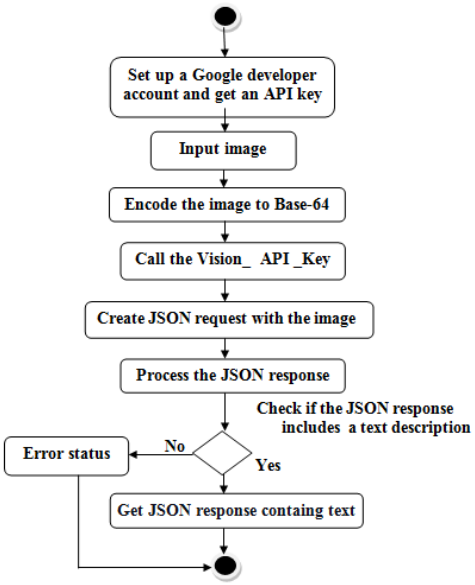

Fig. 3. Text Extraction using Google OCR.

V. Results and Discussion

This paper used a desktop computer with an i8 processor, 8 GB of RAM, 512 GB of storage, and an HDD (Hard Disk Drive). The text extraction results from each lecture video using the acquired key-frames demonstrated that GCV performed better than other OCR APIs in extracting text from the key-frames, with an average accuracy of 96.7 percent, as shown in Table I. Tesseract's is 92 percent, Abbyy Finereader's is 90.5 percent, and Transym's is 80.8 percent.

TABLE I. A COMPARISON OF OCR APIS

Dataset (key-frames of LV)	Method	Pr (%)	Re (%)	F1-Score (%)
38	Google OCR	97.2	94.7	**97.4**
	Tesseract	88.2	89.4	88.7
	Abbyy Finereader	87.8	86.8	87.2
	Transym	65.6	84.2	73.7
39	Google OCR	94.7	97.4	**96.0**
	Tesseract	86.1	92.3	89.0
	Abbyy Finereader	91.4	89.7	90.5
	Transym	72.7	84.6	78.1
38	Google OCR	97.2	97.3	**97.2**
	Tesseract	88.8	94.7	91.6
	Abbyy Finereader	88.2	89.4	88.7
	Transym	62.5	84.2	71.7
75	Google OCR	97.2	97.3	**97.2**
	Tesseract	91.5	94.6	93.0
	Abbyy Finereader	91.3	92.0	91.6
	Transym	83.3	88.0	85.5
72	Google OCR	98.5	98.6	**98.5**
	Tesseract	92.6	94.4	93.4
	Abbyy Finereader	95.5	93.0	94.2
	Transym	86.1	90.2	88.1
51	Google OCR	98.3	96.0	**97.1**
	Tesseract	93.4	90.1	91.7
	Abbyy Finereader	88.6	86.2	87.3
	Transym	75.6	80.3	77.8
125	Google OCR	89.4	98.4	**93.6**
	Tesseract	96.6	96.8	96.6
	Abbyy Finereader	94.1	95.2	94.6
	Transym	88.8	93.6	91.1

The GCV OCR's accuracy is much higher than that of other techniques while taking into account the file size and resolution. Additionally, the accuracy of the low-resolution or small-size images is the lowest. The three parameters listed below are used to evaluate performance.

$$Recall\ (Re) = \frac{Extracted\ text}{Total\ key-frames} \quad (13)$$

$$Precision(Pr) = \frac{Correctly\ extracted\ text}{Extracted\ text} \quad (14)$$

$$F1 - score = \frac{2 \times Re \times Pr}{Re + Pr} \quad (15)$$

The images were reduced to 720 x 480 pixels because the more pixels an image has, the longer OCR would take to process it into grayscale. To cut down on the amount of time needed for STT translation, all preprocessing stages were completed. Everything in the GCV OCR is contained within a RESTful API that provides a JSON structure with the text and bounding box (containing image text area with x and y co-ordinates). It takes about 15 seconds to translate a STT. The sample output of text extraction using GCV OCR is shown in Fig. 4. Precision, recall, and F-score of different OCR APIs is shown in Fig. 5.

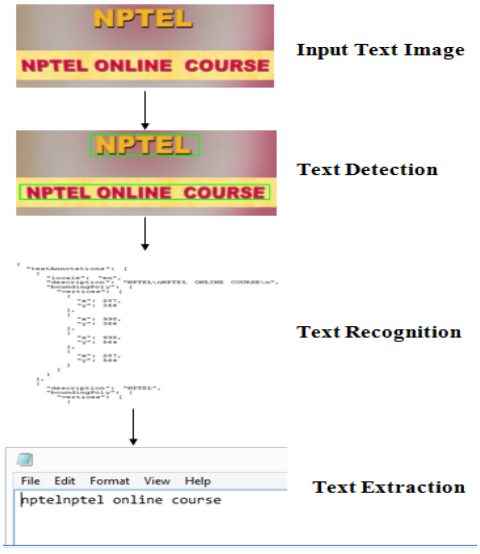

Fig. 4. Text Extraction from an Image.

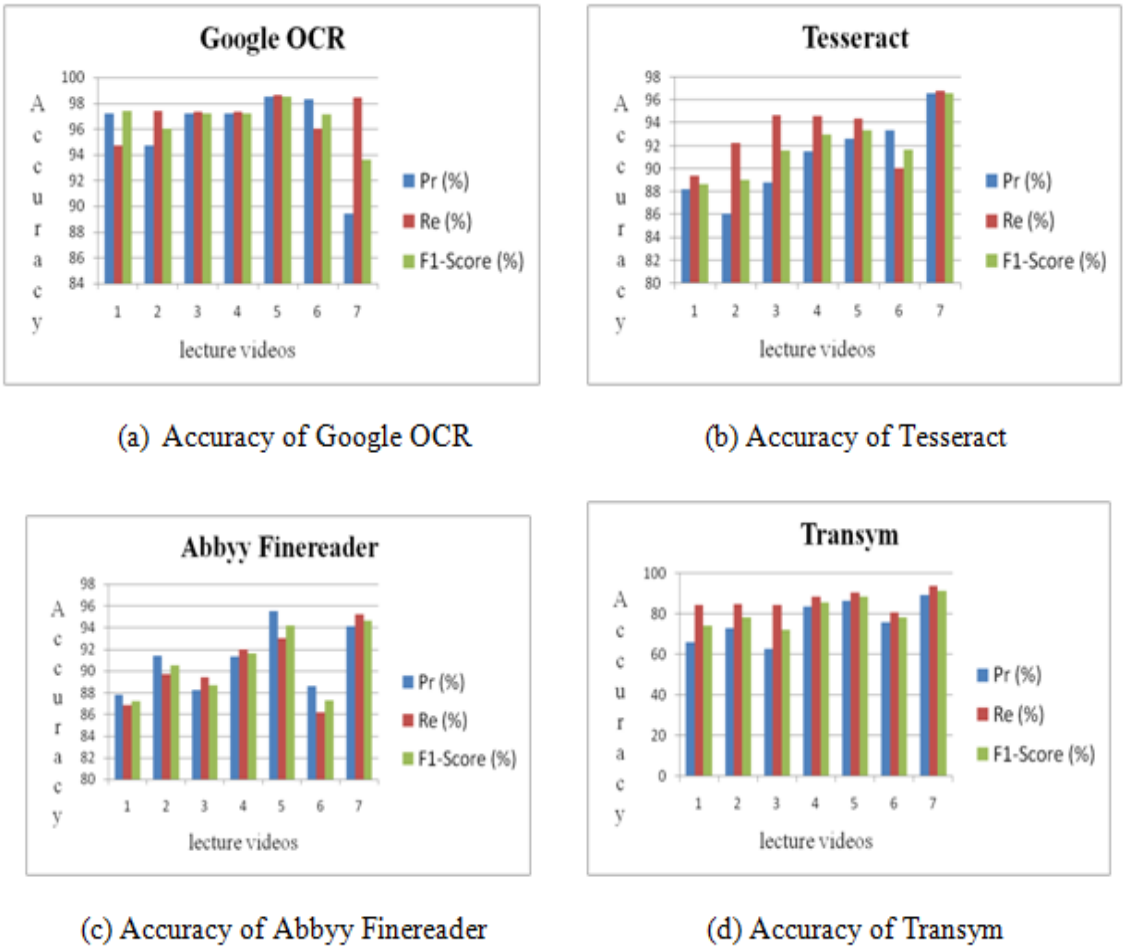

Fig. 5. Precision, Recall, and F-Score of Different OCR APIs.

From this result we can clearly say that the GCV OCR is much better than Tesseract, Abbyy Finereader, and Transym, with accuracies of 96.7%, 92.0%, 90.5%, and 80.8%, respectively, in STT conversion (shown in Fig. 6).

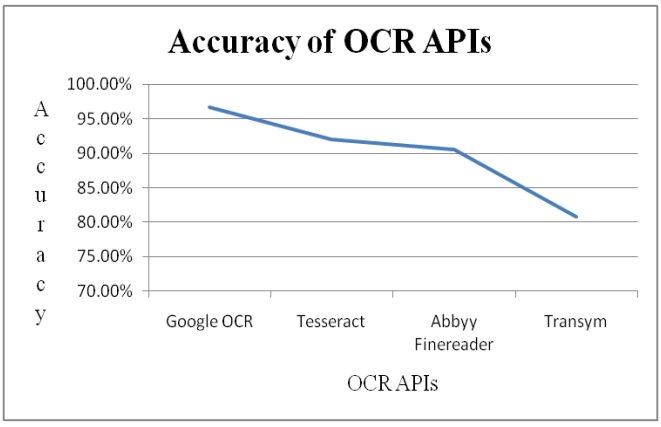

Fig. 6. Performance Comparison of Different OCR APIs.

A comparison of a number of quality criteria provided by the OCR systems is summarised in Table II.

A. Discussion

In order to make the tools more effective in identifying and processing information, this section addresses some noteworthy results, fascinating difficulties, and other usage domains or areas of study. In terms of size and image attributes, the GCV API is more accurate than competing APIs. In terms of additional factors, we discovered the following:

- All the tools were able to recognise English letters with comparable proficiency.
- Slightly elevated images could be detected by all the tools with a high degree of accuracy, while very small, distant, or blurry images could not be recognised by both the Abbyy Finereader and Transym tools.
- The supplied image's watermark background and grey-colored text significantly lower the text identification performance.

TABLE II. OCR SYSTEMS

OCR methods	Pros	Cons
Transym [27]	• Available as a SDK • Multilingual support • Support machine written characters	• Not available online • Not open source
ABBYY Finereader [6]	• Best for business users • Supports Automation • Batch processing • Support for 192 languages	• Not for general users • Not open access
Tesseract [4] [5][6]	• Quite powerful and accurate • Supports over 100 languages	• Not for business users
GCV API [4]	• Quick and easy OCR software for general users • Support for over 200 languages • Mobile app support • Available on almost all platforms • Quite powerful and accurate	• Not open access

The Tesseract and GCV APIs outperform the other two. Because Tesseract is open-source software that can be developed, customised, and managed according to particular needs, it is great software for developers. Tesseract, however, can be somewhat challenging to install and configure. Due to the availability of a variety of services, the GCV API performs better than Tesseract. It is also straightforward to connect to, configure, and use services on. The following are some potential strategies to enhance the functionality of OCR technologies to make them more effective at recognising and evaluating information:

- To cut down on extra reading material and prevent wrongly positioned images, the programmer should define the border, frame, or template matching.
- Before the recognition process, the programmer should make any necessary colour adjustments to the character, as well as remove the extra watermark backdrop.
- To aid with the understanding difficulties with the presentation slides, the programmer should create a programme that can connect models, enabling both printed and handwritten text recognition.
- The effectiveness of the post-processing outcomes can be increased by using natural language processing techniques.

VI. CONCLUSION

Extracting text from lecture slides is crucial for indexing the lecture video. This study evaluates the text extraction capabilities of the GCV OCR, Tesseract, Abbyy Finereader, and Transym in order to develop a lecture video indexing scheme for the non-linear steering in lecture videos so that viewers only watch the interesting points of topics. According to the test findings, Google Cloud Vision had accuracy rates of 96.7 percent, 92.0 percent, 90.5 percent, and 80.8 percent, which were higher than those of Tesseract, Abbyy Finereader, and Transym. The amount of time needed for processing an image grows as its resolution does. In order to reduce the time needed for STT translation, the images are first reduced to 740 x 480 pixels and then converted to grayscale. According to this study, resizing and preprocessing an image before performing OCR can greatly increase the OCR's accuracy. It takes about 15 seconds to translate an STT. This study gives an idea for the researchers who work on OCR.

Our future work will include an effort to assess additional OCR services utilising substantial datasets and more statistically significant analyses for their accuracy and durability. We will make use of cutting-edge image processing techniques and assess how they may be used to create OCR systems that are more precise and effective. In the future, we'll also work on turning the audio from the lecture into text and creating the index points using an effective ASR tool. The results of this study will help to generate index points.

ACKNOWLEDGMENT

We are very thankful to our parents, family, and friends for supporting to complete this work.

REFERENCES

[1] Shashidhar R, A S Manjunath, Santhosh kumar R, Roopa M, Puneeth S B. “Vehicle Number Plate Detection and Recognition using YOLO- V3 and OCR Method” 2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNWC) . pp. 1-6 (2021). https://doi.org/10.1109/ICMNWC52512.2021.9688407.

[2] Shrey Srivastava, Amit Vishvas Divekar, Chandu Anilkumar, Ishika Naik, Ved Kulkarni and V. Pattabiraman. “Comparative analysis of deep learning image detection algorithms.” Journal of big data. Pp. 1-27 (2021). https://doi.org/10.1186/s40537-021-00434-w.

[3] Canhui Xu, Cao Shi , Hengyue Bi , Chuanqi Liu, Yongfeng Yuan, Haoyan Guo, And Yinong Chen. “A Page Object Detection Method Based on Mask R-CNN.” IEEE Access. Pp. 143448- 143456 (2021). https://doi.org/10.1109/ACCESS.2021.3121152.

[4] Thomas Hegghammer. “OCR with Tesseract, Amazon Textract, and Google Document AI: a benchmarking experiment.” Journal of Computational Social Science (2022) 5:861–882. https://doi.org/10.1007/s42001-021-00149-1.

[5] Malathi T , Selvamuthukumaran D , Diwaan Chandar C S , Niranjan V , Swashthika A K. “An Experimental Performance Analysis on Robotics Process Automation (RPA) With Open Source OCR Engines: Microsoft Ocr And Google Tesseract OCR.” IOP Conf. Series: Materials Science and Engineering. Pp. 1-9 (2021). doi:10.1088/1757-899X/1059/1/012004.

[6] Abdulkarim Malkadi, Mohammad Alahmadi, Sonia Haiduc. “A Study on the Accuracy of OCR Engines for Source Code Transcription from Programming Screencasts.” 2020 Association for Computing Machinery. ACM. https://doi.org/10.1145/3379597.3387468.

[7] https://research.aimultiple.com/ocr-accuracy/

[8] Hossein Hosseini, Baicen Xiao and Radha Poovendran “Google’s Cloud Vision API Is Not Robust to Noise,” 16th IEEE International Conference on Machine Learning and Applications December 18-21, 2017.

[9] Rithika.H, B. Nithya santhoshi “Image Text to Speech Conversion in The Desired Language by Translating with Raspberry Pi, ” International Conference on Computational Intelligence and Computing Research 2016.

[10] Yasuhisa Fujii, "Optical Character Recognition Research at Google", IEEE 7th Global Conference on Consumer Electronics (GCCE), December 2018.

[11] Mr. Rajesh M., Ms. Bindhu K. Rajan Ajay Roy, Almaria Thomas K, Ancy Thomas, Bincy Tharakan T, Dinesh C “Text recognition and face detection aid for visually impaired person using raspberry pi” International Conference on circuits Power and Computing Technologies [ICCPCT] July 2017.

[12] Mr. Rishabh Dubey "Machine Learning in the Field of Optical Character Recognition (OCR)" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-4 | Issue-5, August 2020, pp.1664-1668, URL: https://www.ijtsrd.com/papers/ijtsrd33233.pdf

[13] Devices Yu Weng and Chunlei Xia. “A New Deep Learning-Based Handwritten Character Recognition System on Mobile Computing.” *Mobile Networks and Applications* volume 25, pages402–411 (2020). https://doi.org/10.1007/s11036-019-01243-5.

[14] J. Ravagli, Z. Ziran, and S. Marinai, “Text recognition and classification in floor plan images,” in 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), vol. 1, Sep. 2019, pp. 1–6.

[15] Liang, J., Doermann, D. and Li, H. (2015) “Camerabased analysis of text and documents: a survey”, International Journal on Document Analysis and Recognition, pp. 1-21.

[16] Lingqian Yang, Daji Ergu, Ying Cai, Fangyao Liu, Bo Ma. “A review of natural scene text detection methods.” The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 & 2021). Procedia Computer Science 199 (2022) 1458–1465. https://doi.org/10.1016/j.procs.2022.01.185

[17] C. Patel, A. Patel, and D. Patel, “Optical character recognition by open source OCR tool tesseract: a case study,” International Journal of Computer Applications, vol. 55, no. 10, pp. 50–56, Oct. 2012, doi: 10.5120/8794-2784.

[18] M. Sugadev, Yogesh, P. K. Sanghamreddy, and S. K. Samineni, “Rough terrain autonomous vehicle control using Google Cloud Vision API,” in 2019 2nd International Conference on Power and Embedded Drive Control (ICPEDC), Aug. 2019, pp. 244–248, doi: 10.1109/ICPEDC47771.2019.9036621.

[19] J. Sharma, G. S. Sindhu, S. Sejwal, J. Solanki, and R. Majumdar, “Intelligent vehicle registration certificate,” in 2019 Amity International Conference on Artificial Intelligence (AICAI), Feb. 2019, pp. 418–423, doi: 10.1109/AICAI.2019.8701286.

[20] F. Adamo, F. Attivissimo, A. Di Nisio, and M. Spadavecchia, “An automatic document processing system for medical data extraction,” Measurement, vol. 61, pp. 88–99, Feb. 2015, doi: 10.1016/j.measurement.2014.10.032.

[21] I. Kavati, G. K. Kumar, S. Kesagani, and K. S. Rao, “Signboard text translator: a guide to tourist,” International Journal of Electrical and Computer Engineering (IJECE), vol. 7, no. 5, pp. 2496–2501, Oct. 2017, doi: 10.11591/ijece.v7i5.pp2496-2501.

[22] A. J. Samuel and S. Sebastian, “An algorithm for IoT based vehicle verification system using RFID,” International Journal of Electrical and Computer Engineering (IJECE), vol. 9, no. 5, pp. 3751–3758, Oct. 2019, doi: 10.11591/ijece.v9i5.pp3751-3758.

[23] A. Brunetti, D. Buongiorno, G. F. Trotta, and V. Bevilacqua, “Computer vision and deep learning techniques for pedestrian detection and tracking: A survey,” Neurocomputing, vol. 300, pp. 17–33, Jul. 2018, doi: 10.1016/j.neucom.2018.01.092.

[24] Google drive (2012). http://drive.google.com

[25] S mith, R.: An overview of the tesseract OCR engine (2007)

[26] Abbyy OCR (2016). https://www.abbyy.com/

[27] Transym OCR. http:// www.transym.com/download.htm

[28] Patil, V.V., Sanap, R.V., Kharate, R.B. “Optical character recognition using artificial neural network.” Int. J. Eng. Res. Gen. Sci. 3(1), 7 (2015)

[29] Bautista, C.M., Dy, C.A., Ma˜nalac, M.I., Orbe, R.A., Cordel, M. “Convolutional neural network for vehicle detection in low resolution traffic videos.” In: 2016 IEEE Region 10 Symposium (TENSYMP), pp. 277–281. IEEE (2016)

[30] Ye, Q., Doermann, D. “Text detection and recognition in imagery: a survey.” IEEE Trans. Pattern Anal. Mach. Intell. 37(7), 1480–1500 (2015).