CS8811 PROJECT WORK

ENHANCED EXTRACTION OF TEXTUAL CHARACTERS FROM MULTIMEDIA USING NATURAL LANGUAGE PROCESSING

A PROJECT REPORT

Submitted by

DINESH K **(111719104044)**

BHARATH T **(111719104022)**

BUDHARAJU SUMANTH **(111719104030)**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

R.M.K. ENGINEERING COLLEGE

(An Autonomous Institution)

R.S.M. Nagar, Kavaraipettai-601 206

ANNA UNIVERSITY: CHENNAI 600 025

MARCH 2023

R.M.K. ENGINEERING COLLEGE

(An Autonomous Institution)

R.S.M. Nagar, Kavaraipettai-601 206

BONAFIDE CERTIFICATE

Certified that the project report **"ENHANCED EXTRACTION OF TEXTUAL CHARACTERS FROM MULTIMEDIA USING NATURAL LANGUAGE PROCESSING"** is the bonafide work of **"DINESH K (111719104044), BHARATH T (111719104022), BUDHARAJU SUMANTH (111719104030)"** who carried out the project under my supervision.

SIGNATURE

Dr. T. SETHUKARASI,
HEAD OF THE DEPARTMENT
Department of Computer Science
and Engineering,
R.M.K. Engineering College,
R.S.M. Nagar,Kavaraipettai,
Tiruvallur District - 601206.

SIGNATURE

Dr. P. KAVITHA
SUPERVISOR
Associate Professor
Department of Computer Science
and Engineering,
R.M.K. Engineering College,
R.S.M. Nagar, Kavaraipettai
Tiruvallur District-601206.

Submitted for the project viva voce held on …….

at R.M.K. Engineering College, Kavaraipettai, Tiruvallur District -601 206.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We earnestly portray our sincere gratitude and regard to our beloved **Chairman Shri. R. S. Munirathinam, our Vice Chairman, Mr. R. M. Kishore** and **our Director, Shri. R. Jyothi Naidu,** for the interest and affection shown towards us throughout the course.

We convey our sincere thanks to our Principal, **Dr. K. A. Mohamed Junaid,** for being the source of inspiration in this college.

We reveal our sincere thanks to our **Professor** & **Head of the Department**, Computer Science and Engineering**, Dr. T. Sethukarasi**, for her commendable support and encouragement for the completion of our project.

We would like to express our sincere gratitude for our **Project Coordinator Dr**. **A. Thilagavathy**, Associate Professor, for her valuable suggestions towards the successful completion for this project in a global manner.

We convey our deep gratitude and we are very much indebted to our versatile **Project Guide** , **Dr. P. Kavitha**, Associate Professor for her valuable suggestions and spontaneous guidance to complete our project.

We take this opportunity to extend our thanks to all faculties of Department of Computer Science and Engineering, parents and friends for all that they meant to us during the crucial times of the completion of our project.

ABSTRACT

Enhanced extraction of textual characters represents a comprehensive framework for detection and recognition of textual content in video frames as well as in images. Detection of text from document images enables Natural Language Processing algorithms to decipher the text and make sense of what the document conveys. The existing scene text detection techniques are unable to detect text accurately from the videos having low contrast, complex background or excessively small fonts. Furthermore, the text can be easily translated into multiple languages, and summarizes the text in to meaningful phrases using the transformers and pipelines in NLP text summarization. Then converts the summarizer text to an audio using Text-To- Speech library. NLP, also to recognize text from supermarket product names, traffic signs, and even from billboards, making them an effective translator and interpreter. The development of this application which will be great help to people with visual impairment and those who don't know other language.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	iv
	LIST OF FIGURES	viii
	LIST OF TABLES	ix
	LIST OF ABBREVATIONS	ix
1	**INTRODUCTION**	1
	1.1 Problem Statement	2
	1.2 Literature Survey	3
	1.3 System Requirements	10
	1.3.1 Hardware Requirements	10
	1.3.2 Software Requirements	10
	1.4 Feasibility Study	10
	1.4.1 Technical Feasibility	11
	1.4.2 Behavioral Feasibility	11
	1.4.3 Economical Feasibility	12
	1.4.4 Social Feasibility	12
2	**SYSTEM ANALYSIS**	
	2.1 Existing System	13
	2.1.1 Disadvantages of Existing System	13
	2.2 Proposed System	14
	2.2.1. Advantages of Proposed System	14

3	**SYSTEM DESIGN**	
	3.1 System Architecture	15
	3.2 UML diagrams	
	3.2.1 Use Case diagram	16
	3.2.2 Class diagram	17
	3.2.3 Sequence diagram	18
	3.2.4 Activity diagram	19
	3.2.5 ER diagram	20
4	**SYSTEM IMPLEMENTATION**	
	4.1 Modules	21
	4.2 Module Description	
	4.2.1 Accessing the Webcam	21
	4.2.2 Text Detection and Recognition	22
	4.2.3 Text Summarization	23
	4.2.4 Text Translation	23
	4.2.5 Speech Representation	24
	4.3 Prediction Models	
	4.3.1 To Extract the text from the scene image	25
	4.3.2 To Recognize the text from the bounded region of image	26
	4.3.2 To Tokenize the recognized text to the specified amount	27
	4.4 Testing	
	4.4.1 Testing Objectives	28
	4.4.2 Test cases	29

5	**RESULTS**	30
	5.1 Performance Metrics	
	5.1.1 Accuracy	33
	5.1.2 Precision	33
	5.1.3 Recall	33
	5.1.4 F-Score	33
	5.1.5 Confusion Matrix	34
6	**CONCLUSION**	35
	REFERENCES	36
	APPENDIX I – SOURCE CODE	38
	APPENDIX II - SCREENSHOTS	42

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
3.1	System Architecture Diagram	13
3.2.1	Use Case Diagram	14
3.2.2	Class Diagram	15
3.2.3	Sequence Diagram	16
3.2.4	Activity Diagram	17
3.2.5	ER Diagram	18
4.2.2	Detection and Recognition of Text	20
4.2.3	Summarization of Text	21
4.2.4	Text Translation	22
4.3.2	LSTM Architecture	24
4.3.2.1	Training validation data for LSTM model	27
4.3.3	Tokenization of words using NegEx	28
5.1	Graph analysis of LSTM model with Tesseract	32
5.2	Graph analysis of NegEx algorithm with other modules	33
5.3	Comparison of Machine Translation with Human Translation	34

LIST OF TABLES

TABLE NO	TITLE	PAGE NO
4.4	Test Cases	29
5.1	Performance Metrices	34

LIST OF ABBREVATIONS

OCR - Optical Character Recognition

NLP - Natural Language Processing

CTPN - Connectionist Text Proposal Network

MSER - Maximally Stable External regions

NEGX - Negate With Extend

CRAFT - Computerized Relative Allocation of Facilities Technique

CNN - Convolutional Neural Networks

CRNN - Convolutional Recurrent Neural Networks

LSTM - Long Short-term Memory Networks

UML - Unified Modelling Language

DSP - Digital Signal Processing

MDS - Mutual Direction Symmetry

MMS - Mutual Direction Symmetry

GVS - Gradient Vector Symmetry

DCT - Discrete Cosine Transform

SVM - Support Vector Machine

DBMS - Database Management System

CHAPTER 1

INTRODUCTION

In the recent years, there has been a tremendous increase in the amount of digital multimedia data, especially the video content, both in the form of video archives and live streams. According to statistics, 300 h of video is being uploaded every minute on the YouTube. With such huge collections of data, there is a need to have efficient as well as effective retrieval techniques allowing users retrieve the desired content. The term content may refer to the visual content (for example, objects or persons in the video), audio content (the spoken keywords for instance), or the textual content (News tickers, anchor names, score cards, etc.). The textual content in video can be categorized into two broad classes, scene text, and caption text. Examples of scene text include advertisement banners, sign boards, and text on a T-shirt. Scene text is commonly employed for applications like robot navigation and assistance systems for the visually impaired. The key components of a textual content based indexing and retrieval system include detection of text regions, extraction of text (segmentation from background), identification of script (for multi-script videos), and finally recognition of text (video OCR). The key highlights of this study are outlined in the following. Development of a comprehensive dataset of video frames with ground truth information supporting evaluation of detection and recognition tasks. Investigation of state-of-the-art deep learning CRAFT Algorithm based object detectors, fine-tuning them to detection of textual content. Validation of proposed techniques using a comprehensive series of experiments.

1.1 Problem Statement

These days there is a huge demand for storing information in a summarized version to a computer storage disk from the data available in images or video frames or handwritten documents to later re-utilize this information by means of computers. One simple way to store information to computer system from these paper documents could be to first scan the documents and then store them as image files. But to re-utilize this information, it would very difficult to read or query text or other information from these image files. Therefore a technique to automatically retrieve and store information, in particular text, from image files is needed. Similarly, our goal is to develop a machine learning model for image or video frames to text and audio representation, to potentially replace the updatable supervised machine learning classification models by predicting results in the form of best accuracy by comparing supervised algorithms. Our model evaluation techniques used to build the text extraction from the image. After that the text can be converted into summarization and it can translate the content to the language we want and it can also given the output in the audio format.

1.2. Literature Survey

The study in [1] presented a system for extraction and script identification of multilingual artificial text appearing in video images. As opposed to most of the existing text extraction systems which target textual occurrences in a particular script or language, [1] proposed a generic multilingual text extraction system that relies on a combination of unsupervised and supervised techniques. The unsupervised approach is based on application of image analysis techniques which exploit the contrast, alignment and geometrical properties of text and identify candidate text regions in an image. Potential text regions are then validated by an Artificial Neural Network (ANN) using a set of features computed from Gray Level Co-occurrence Matrices (GLCM). The script of the extracted text is finally identified using texture features based on Local Binary Patterns (LBP). [1] evaluated on video images containing textual occurrences in five different languages including English, Urdu, Hindi, Chinese and Arabic. The promising results of [1] experimental evaluations validate the effectiveness of the proposed system for text extraction and script identification.

The study in [2] described a system that automatically identifies the script used in documents stored electronically in image form. The system can learn to distinguish any number of scripts. It develops a set of representative symbols (templates) for each script by clustering textual symbols from a set of training documents and representing each cluster by its centroid. "Textual symbols" include discrete characters in scripts such as Cyrillic, as well as adjoined characters, character fragments, and whole words in connected scripts such as Arabic. To identify a new document's script, the system compares a subset of symbols from the document to each script's templates, screening out rare or unreliable templates, and choosing the script whose templates provide the best match. In [2] system, trained on 13 scripts, correctly identifies all test documents except those printed in fonts that differ markedly from fonts in the training set.

The study in [3] developed a techniques for distinguishing which language is represented in an image of text. This work is restricted to a small but important subset of the world's languages. The method first classifies the script into two broad classes: Han and Latin-based. This classification is based on the spatial relationships of features related to the upward concavities in character structures. Language identification within the Han script class (Chinese, Japanese, Korean) is performed by analysis of the distribution of optical density in the text images. They handle 23 Latin-based languages using a technique based on character shape codes, a representation of Latin text that is inexpensive to compute.

The study in [4], a document page may contain several script forms. For optical character recognition (OCR) of such a document page, it is necessary to separate the scripts before feeding them to their individual OCR systems. An automatic technique for the identification of printed Roman, Chinese, Arabic, Devnagari and Bangla text lines from a single document is proposed. Shape based features, statistical features and some features obtained from the concept of a water reservoir are used for script identification. The proposed scheme has an accuracy of about 97.33%.

The study in [5] proposed a new text detection method using the texture feature derived from text strokes. The model [5] consists of four steps: wavelet multiresolution decomposition, thresholding and pixel labeling, text detection using texture features from strokes, and refinement of mask image. Experiment results show that our method is effective.

The study in [6] proposed a video text extraction scheme using key text points (KTPs). A KTP is defined as the point that has strong textural structure in multi-directions simultaneously. First, a KTP can be acquired by the three high-frequency sub bands obtained from the wavelet transform. Second, the difference at the KTPs between neighboring frames is calculated as the similarity measure in text tracking, which can reduce the influence of dramatic background variation. Finally, the total number of the KTPs in each connected component is calculated to remove the background interference and to extract texts for text segmentation. Experimental results show that the proposed text detection and localization algorithm is robust to the font size, style, color and alignment of texts. Text tracking significantly improves the efficiency of text detection, and the similarity measure based on KTPs decreases the influence of scene changes and motions. The proposed [6] text segmentation has a promising performance when processing video scenes with complex backgrounds and low contrast between texts and backgrounds.

The study in [7] presented a study of some pre-processing techniques and features for word-wise script identification from video frames. Video frames are mostly colored and suffer from low resolution, blur, background noise, to mention a few. In [7] , an attempt has been made to explore whether the traditional script identification techniques can be useful in video frames. Three feature extraction techniques, namely Zernike moments, Gabor and gradient features, and SVM classifiers were considered for Multiple Languages. Experiments show that the super resolution technique with gradient features has performed well, and an accuracy of 87.5% was achieved when testing on 896 words from three different scripts. [7] also reveals that the use of proper pre-processing approaches can be helpful in applying traditional script identification techniques to video frames.

The study in [8] presented a detailed review of current script and language identification techniques. The main criticism of the existing techniques is that most of them rely on either connected component analysis or character segmentation. [8] went on to present a new method based on texture analysis for script identification which does not require character segmentation. Multiple channel (Gabor) filters and grey level co-occurrence matrices are used in independent experiments in order to extract texture features. Classification of test documents is made based on the features of training documents using the K-NN classifier. Initial results of over 95% accuracy on the classification of 105 rest decrements from 7 scripts are very promising. In this method shows robustness with respect to noise, the presence of foreign characters or numerals, and can be applied to very small amounts of text.

The study in [9] employed gradient angle segmentation on words from video text lines. In [9] presented new Gradient-Angular-Features (GAF) for video script identification, namely, Arabic, Chinese, English, Japanese, Korean and Tamil. [9] work enables us to select an appropriate OCR when the frame has words of multi-scripts. They employ gradient directional features for segmenting words from video text lines. The skeleton of the text candidates is analyzed to identify Potential Text Candidates (PTC) by filtering out unwanted text candidates. They proposed novel GAF for the PTC to study the structure of the components in the form of cursive ness and softness. The histogram operation on the GAF is performed in different ways to obtain discriminative features. The method is evaluated on 760 words of six scripts having low contrast, complex background, different font sizes, etc. in terms of the classification rate and is compared with an existing method to show the effectiveness of the method. The project in [9] achieve 88.2% average classification rate.

The study in [10] presented a study of various combinations of features and classifiers to explore whether the traditional script identification techniques can be applied to video frames. A texture based feature namely, Local Binary Pattern (LBP), Gradient based features namely, Histogram of Oriented Gradient (HoG) and Gradient Local Auto-Correlation (GLAC) were used in the study. Combination of the features with SVMs and ANNs where used for classification. Three popular scripts, namely English, Bengali and Hindi were considered in the present study. Experiments show that the GLAC feature has performed better than the other features, and an accuracy of 94.25% was achieved when testing on 1271 words from three different scripts. The study in [10] also reveals that gradient features are more suitable for script identification than the texture features when using traditional script identification techniques on video frames.

The study in [11] presented the final results of the ICDAR 2015 Competition on Video Script Identification. A description and performance of the participating systems in the competition are reported. The general objective of the competition is to evaluate and benchmark the available methods on word-wise video script identification. It also provides a platform for researchers around the globe to particularly address the video script identification problem and video text recognition in general. In total, six systems were received from five participants over the tasks offered. This report [11] details the competition dataset specifications, evaluation criteria, summary of the participating systems and their performance across different tasks.

The study in [12] described a novel deep neural network structure that efficiently identifies scripts of images. In [12] design, they exploit two important factors, namely the image representation, and the spatial dependencies within text lines.

To this end, they bring together a Convolutional Neural Network (CNN) and a Recurrent Neural Network (RNN) into one end-to-end trainable network. The former generates rich image representations, while the latter effectively analyzes long-term spatial dependencies. Besides, on top of the structure, we adopt an average pooling structure in order to deal with input images of arbitrary sizes. Experiments on several datasets, including SIW-13 and CVSI2015, demonstrate that our approach achieves superior performance, compared with previous approaches.

The study in [13] presented a comprehensive framework for detection and recognition of textual content in video frames. More specifically, they target cursive scripts taking Urdu text as a case study. Detection of textual regions in video frames is carried out by fine-tuning deep neural networks based object detectors for the specific case of text detection. Script of the detected textual content is identified using convolutional neural networks (CNNs), while for recognition, [13] proposed a UrduNet, a combination of CNNs and long short- term memory (LSTM) networks. A benchmark dataset containing cursive text with more than 13,000 video frame is also developed. A comprehensive series of experiments is carried out reporting an F-measure of 88.3% for detection while a recognition rate of 87%.

The study in [14] focused on the problem of script identification in unconstrained scenarios. Script identification is an important prerequisite to recognition, and an indispensable condition for automatic text understanding systems designed for multi-language environments. Although widely studied for document images and handwritten documents, it remains an almost unexplored territory for scene text images. They detailed a novel method for script identification in natural images that combines convolutional features and the Naive-Bayes Nearest Neighbor classifier.

The proposed framework efficiently exploits the discriminative power of small stroke-parts, in a fine-grained classification framework. Experiments done in this new dataset demonstrate that the proposed method yields state of the art results, while it generalizes well to different datasets and variable number of scripts. The evidence provided in [14] shows that multi-lingual scene text recognition in the wild is a viable proposition. Source code of the proposed method is made available online.

The study in [15] used transfer learning and fine-tuning in document analysis. Indeed, they deal with the scene script identification quantitatively by comparing the performances of transfer learning and learning from scratch. In [15], they evaluate two CNN architectures trained on natural images: AlexNet and VGG-16. Experimental results on several benchmark datasets namely, SIW-13, MLe2e and CVSI2015, demonstrate that the approach outperforms previous approaches and full training.

The study in [16] focused on the problem of script identification in scene text images. Facing this problem with state of the art CNN classifiers is not straightforward, as they fail to address a key characteristic of scene text instances: their extremely variable aspect ratio. Instead of resizing input images to a fixed aspect ratio as in the typical use of holistic CNN classifiers, they propose here a patch-based classification framework in order to preserve discriminative parts of the image that are characteristic of its class. In [16], they described a novel method based on the use of ensembles of conjoined networks to jointly learn discriminative stroke-parts representations and their relative importance in a patch-based classification scheme. In addition, they proposed a new public benchmark dataset for the evaluation of multi-lingual scene text end-to-end reading systems. Experiments done in this dataset demonstrate the key role of script identification in a complete end-to-end system that combines our script identification method with a previously published text detector and an off-the-shelf OCR engine.

1.3. System Requirements

1.3.1. Hardware Used

- Processor : Any Processor above 500 MHz.
- Ram : 4 GB
- Hard Disk : 4 GB
- Input device : 4 GB
- Output device : VGA and High Resolution

1.3.2. Software Used

- Operating System :Windows 7 or higher
- Programming : Python 3.6
- Machine Learning : Matplotlib, Scikit Learn, OpenCV
- Modules / Libraries : Easyocr, spacy, googletrans, pyttsx3
- Tools : Anaconda with Jupyter Note, VS Code

1.4 Feasibility Study

A feasibility study is an analysis conducted to determine whether a proposed project or idea is viable, practical, and profitable. The study examines various factors such as technical, behavioral, economical and social aspects of the project to determine whether it is feasible. The study typically involves a thorough analysis of various aspects of the project, including the technical feasibility of implementing it, the economic viability of the project, and its social, legal, and environmental impact. The feasibility study also takes into consideration any risks associated with the project and outlines potential solutions to mitigate these risks.

1.4.1. Technical Feasibility

The technical feasibility of text detection from images depends on several factors, including the quality and complexity of the image, the accuracy and speed of the text detection algorithm, and the availability of computing resources. Text detection algorithms typically rely on image processing techniques such as edge detection, thresholding, and morphological operations to identify regions of an image that are likely to contain text. These regions are then analyzed further to extract the actual text. One important consideration for text detection from images is the quality and complexity of the image. Images with high resolution, good lighting, and clear contrast between the text and background are generally easier to process than images with low resolution, poor lighting, or complex backgrounds. There are many different algorithms and techniques that can be used for text detection, each with its own strengths and weaknesses. Some algorithms may be better suited for certain types of images or text, while others may be faster or more accurate overall.

1.4.2. Behavioral Feasibility

User acceptance is an important factor in the behavioral feasibility of text detection from images. If users perceive the technology as invasive or threatening, they may resist its use, which could limit its adoption and effectiveness. On the other hand, if users see the benefits of text detection from images, such as increased efficiency and accuracy in data entry or analysis, they may be more likely to embrace it. Ethical implications are also a key consideration for behavioral feasibility. Text detection from images can potentially be used for nefarious purposes, such as invading individuals privacy or engaging in surveillance. Additionally, there is the possibility of bias and discrimination if the algorithm used to detect text is not sufficiently sensitive to different types of text or backgrounds.

1.4.3. Economical Feasibility

The cost of developing and implementing text detection from images can vary widely depending on the specific technology used, the complexity of the image processing algorithm, and the level of integration required with existing systems. For example, developing a highly accurate and efficient text detection algorithm may require significant investment in research and development. Similarly, integrating the technology with existing systems such as document management software may require additional resources. However, there are also potential cost savings that can be achieved through text detection from images. For example, automating data entry through text detection can reduce the need for manual data entry, which can be time-consuming and error-prone. This can lead to increased efficiency and reduced labor costs.

1.4.4. Social Feasibility

One important factor in social feasibility is privacy. Text detection from images may involve capturing and processing images that contain sensitive or personal information. This could potentially lead to privacy concerns, particularly if individuals are not aware that their images are being processed. Therefore, it is important to ensure that appropriate privacy safeguards are in place, such as obtaining informed consent and implementing appropriate security measures to protect data. Accessibility is another key consideration for social feasibility. Text detection from images should be accessible to all users, regardless of their level of technical expertise or ability. This means ensuring that the technology is easy to use, does not rely on specialized hardware or software, and meets accessibility standards for users with disabilities.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

A Text-to-speech synthesizer is an application that converts text into spoken word, by analysing and processing the text using Natural Language Processing (NLP) and then using Digital Signal Processing (DSP) technology to convert this processed text into synthesized speech representation of the text. Here, we developed a useful text-to-speech synthesizer in the form of a simple application that converts inputted text into synthesized speech and reads out to the user which can then be saved as an mp3.file. The development of a text to speech synthesizer will be of great help to people with visual impairment and make making through large volume of text easier.

2.1.1. Disadvantages of Existing System

- They are not summarising the whole paragraph from an image for easy understanding.
- Advanced NLP concepts not used.

2.2 PROPOSED SYSTEM

The proposed model is to built an application for video/Image-To-Text-Speech. . It also detect the text when webcam placed in any textual regions. It is an important for the people with visual impairment. The application is built to read any form of text images in any languages which is then converted to a English text using a translator package in python. The converted text is summarised into phrases for better understanding the whole meaning in the paragraph using transformers and pipeline Natural language processing. The summarized text is converted into an audio using Text-To-Speech library of NLP. It also extracts the text from the running video and the summarizes the text in to meaningful phrases and convert it to an audio format for the people with visual impairment.

2.2.1 Advantages Of Proposed System

- Advanced NLP concepts used.
- User friendly.
- This system achieves better F1-score, Precision, Recall, and Accuracy.
- This system is helpful in summarizing the whole paragraph.

CHAPTER 3

SYSTEM DESIGN

3.1 System Architecture

The below figure 3.1 represents the system architecture of the proposed system in which we represented all modules including text detection, text summarization, text translation and speech representation and prediction modules.

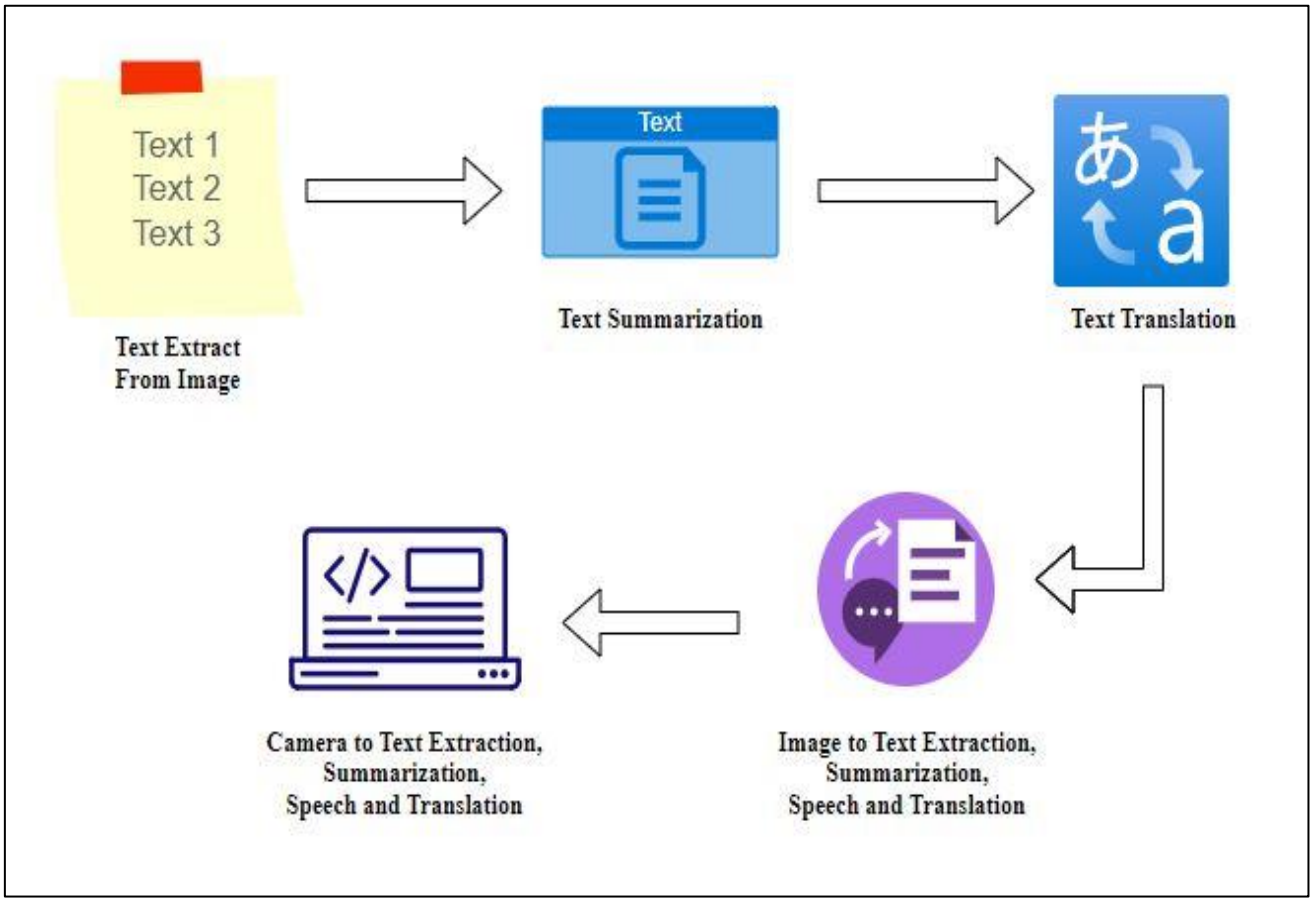


Figure 3.1 System Architecture Diagram

3.2 UML DIAGRAM

3.2.1 USECASE DIAGRAM

Use case diagram shows a set of use cases and actors (a special kind of class) and their relationship. Use case diagrams address the static use case view of our system. These diagrams are especially important in organizing and modeling the behavior of a system; It defines all the process by involving with the server and client.

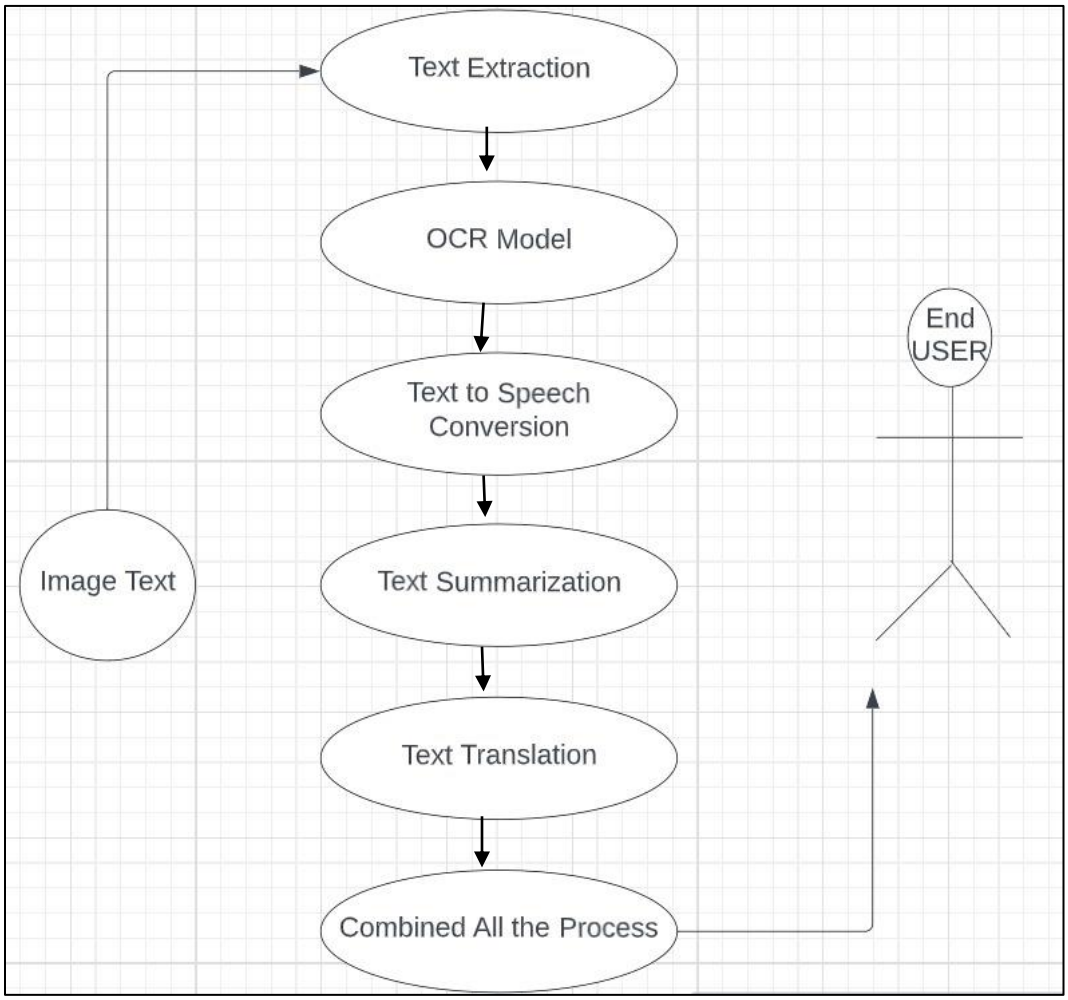


Figure 3.2.1 Use Case Diagram

3.2.2 CLASS DIAGRAM

Class diagrams area unit the foremost common diagrams employed in UML. Class diagrams represent the thing orientation of a system. It contains the main modules and application of that modules under each classes. Therefore, it's usually used for development purposes. This can be the foremost wide used diagram at the time of system construction. Our model contains 8 number of classes include: Data need to be find, Input information, classification, test data, summarize module, translate module, audio module and output.

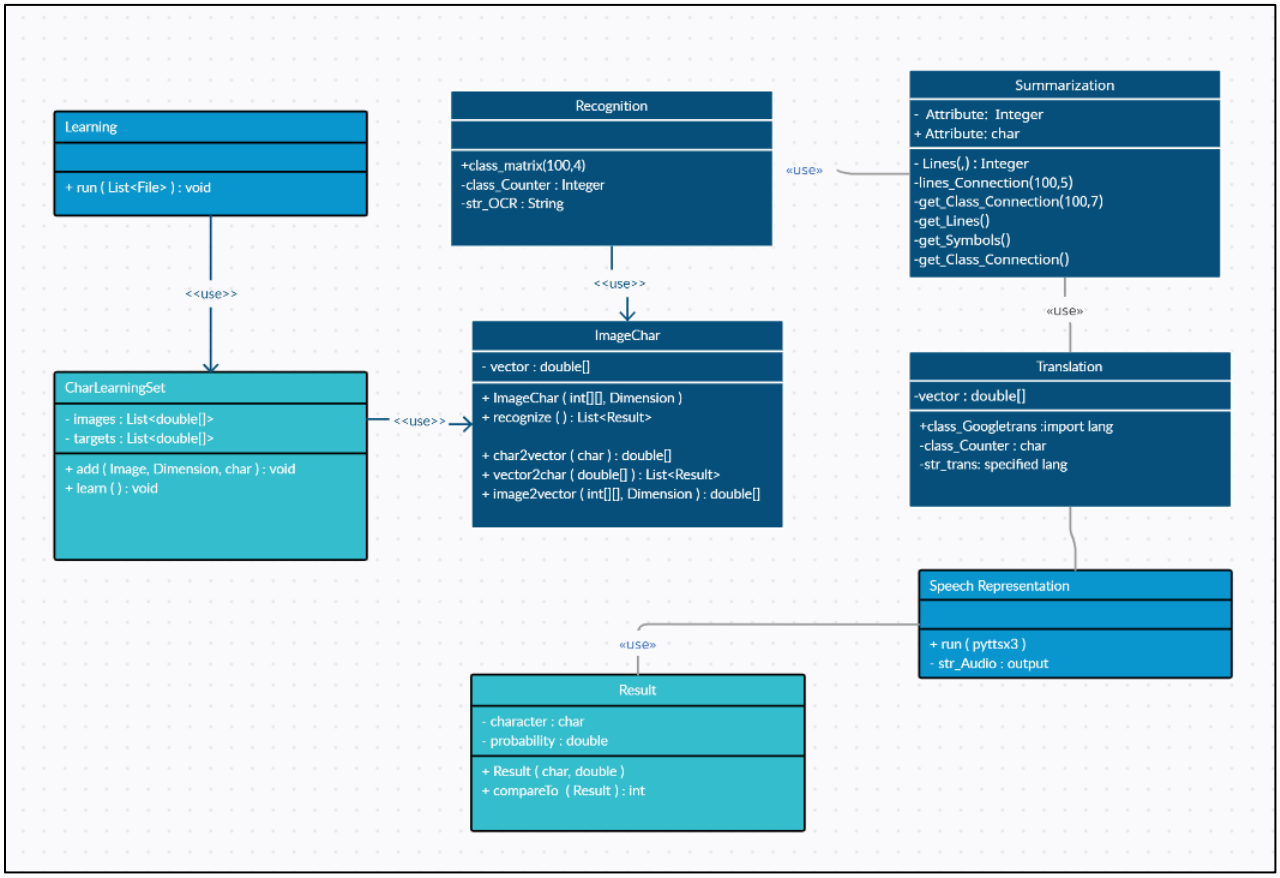


Figure 3.2.2 Class Diagram

3.2.3 SEQUENCE DIAGRAM

A Sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It manages the main application between user and the system (the input we are giving through the responsible path). It is a construct of a Message Sequence Chart. It is given as the image input from the user and the 4 modules representing detection, summarization, translation and speech representation.

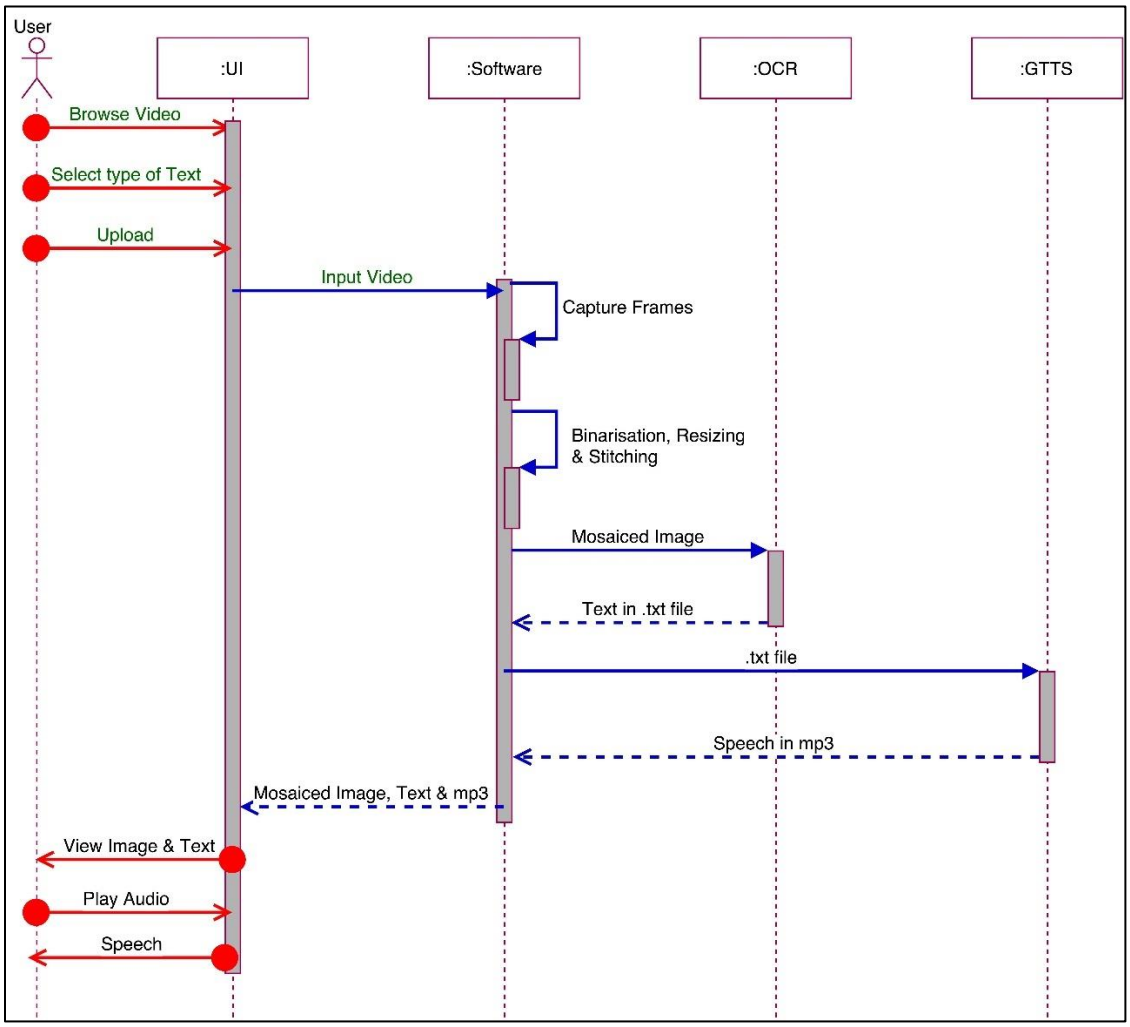


Figure 3.2.3 Sequence Diagram

3.2.4 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to show the process made by the each segmentation and input related queries. An activity diagram shows the overall flow of control.

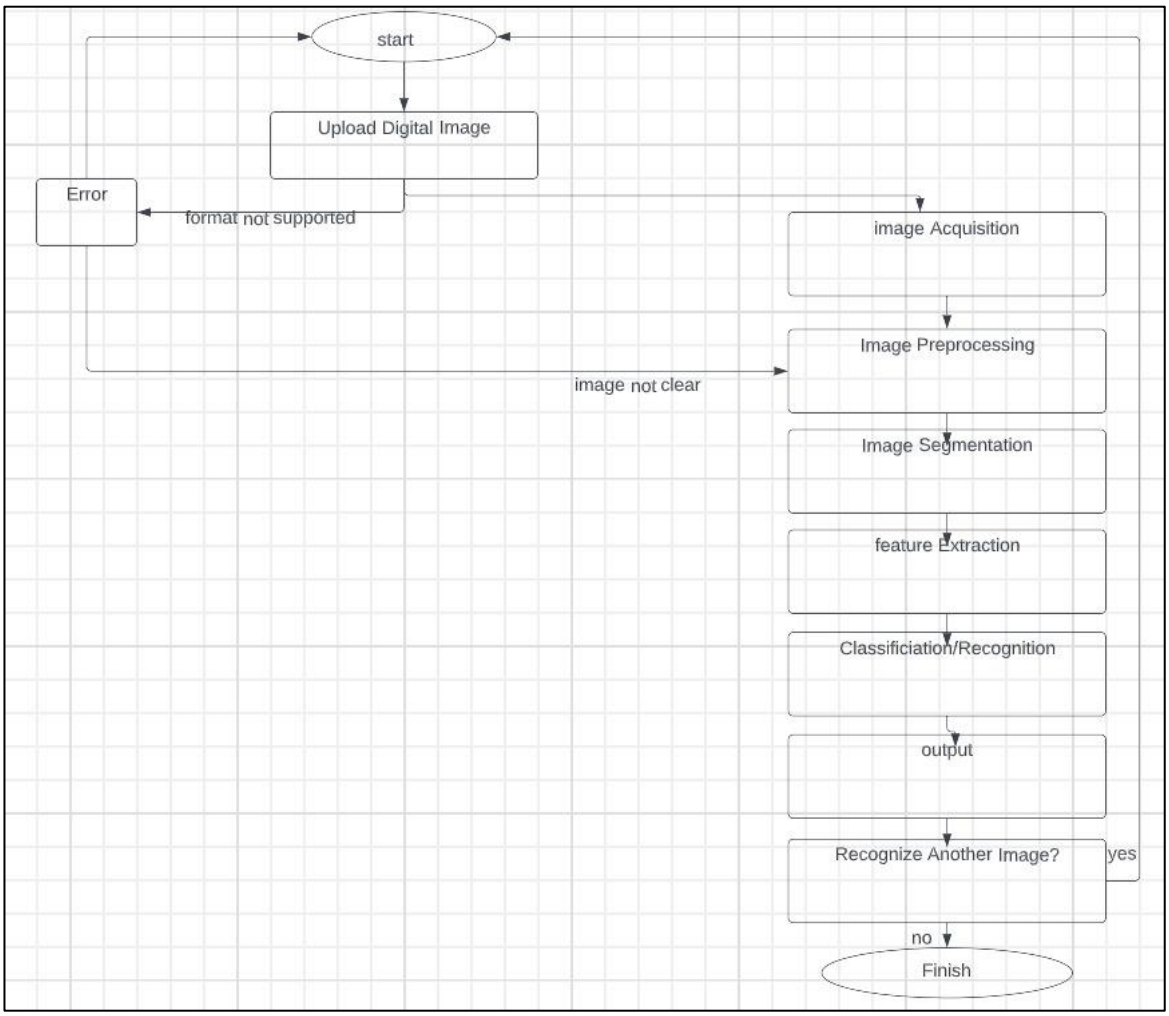


Figure 3.2.4 Activity Diagram

3.2.5 ER DIAGRAM

An Entity Relationship Diagram is a diagram that represents relationships among entities in a database. It is commonly known as an ER Diagram. An ER Diagram in DBMS plays a crucial role in designing the data presented in each database. It flow over the each process and combines with final results that is given to the end user.

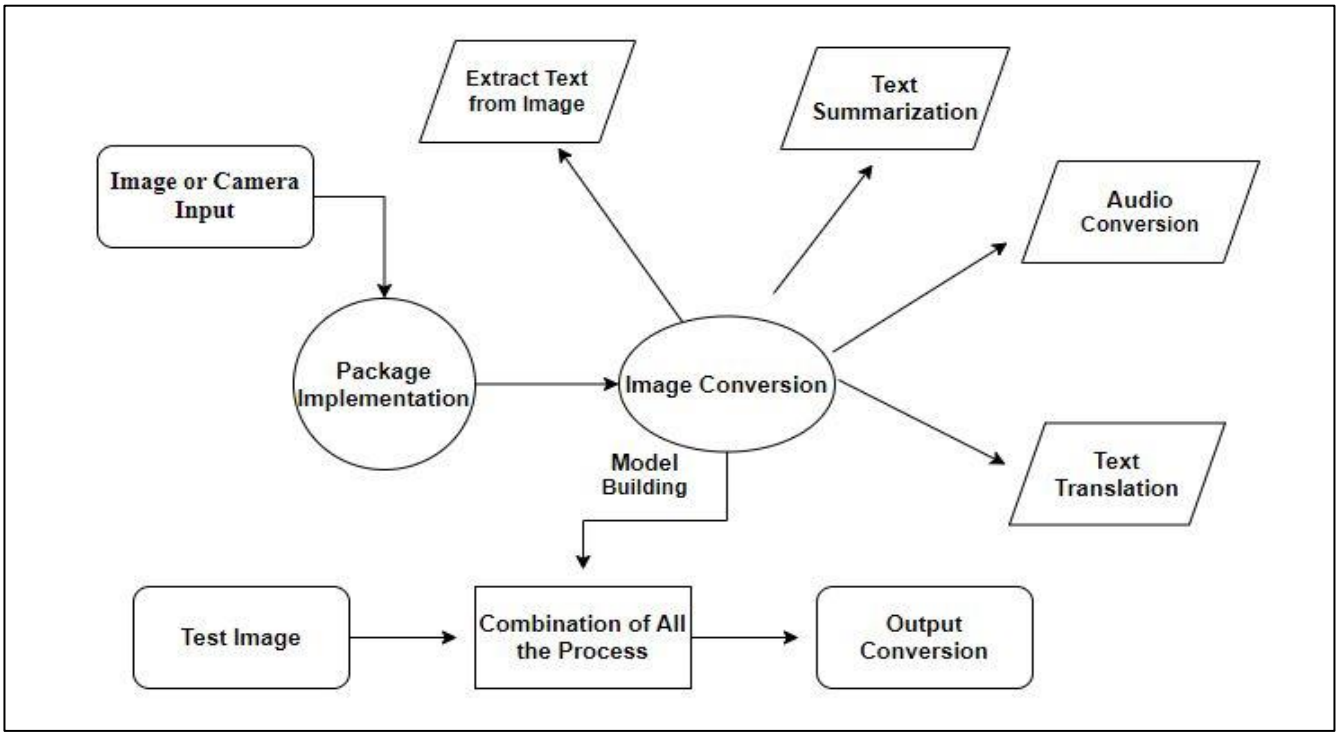


Figure 3.2.5 ER Diagram

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 MODULES

We start by Accessing images on the webcam or system images. Second, we process the data by detecting texts that are presented in the given image. we develop a variety of algorithms for each module and compute performance measures like accuracy, recall, precision, F1 score, etc. Next we summarize the textual characters using some negation techniques . Then, in order to compare and select the language for translation. Using python audio module, we tend the system to speech the text that feed it into the system. The following are the modules used in this system.

4.2 MODULE DESCRIPTION

Each module starting with accessing image, text detection, text summarization are detected by using different supervised ML algorithms and the translation and speech representation are predicted by using default library involved in python.

4.2.1 Accessing the Webcam

We are using the module named OpenCV to read the image from the given input and the color space of the image is first changed and stored in a variable. For color conversion we use the function cv2.cvtColor(input_image, flag). The second parameter flag determines the type of conversion. We can chose among cv2.COLOR_BGR2GRAY and cv2.COLOR_BGR2HSV**.** Here, we use cv2.COLOR_BGR2GRAY that helps us to convert an RGB image to gray scale image. A threshold is applied to the converted image using cv2.threshold function.

The basic Thresholding technique is Binary Thresholding. For every pixel, the same threshold value is applied. If the pixel value is smaller than the threshold, it is set to 0, otherwise, it is set to a maximum value.

4.2.2 Text Detection and Recognition

Using Python And The PyTorch Library, EasyOCR Is Implemented. The Underlying PyTorch Deep Learning Library Can Accelerate Your Text Detection EasyOCR Is Able To Write OCR Texts In 70+ Languages Including English, Hindi, Russian, Chinese, And More. EasyOCR Is Good For Clean Document Scanning And Would Result In Greater Accuracy And Support For LSTM. The detection part is using the CRAFT algorithm and the Recognition model is CRNN. Steps Involved in Detection and Recognition part of Video frames :

1. Import the Easy OCR and thus the necessary libraries to open an image and use it for recognition.
2. Select the language in which you want to extract the text.
3. Read and open the image you want to extract the text from.
4. Calculate the accuracy of box bounds for the text in the image
5. Draw the box bounds for the text in the image.

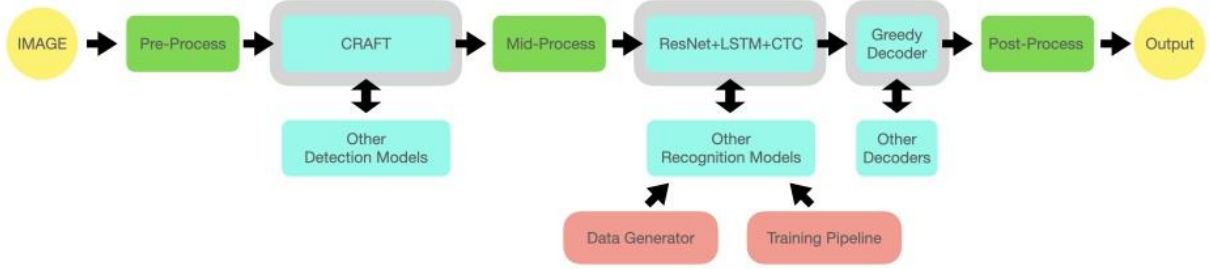


Figure 4.2.2 Detection and Recognition of Text

4.2.3 Text Summarization

We have used SpaCy to import a pre-trained NLP pipeline to help interpret the grammatical structure of the text. This will allow us to identify the most common words that are often useful to filter out (i.e. *STOP_WORDS*) as well as the punctuation (i.e. *punctuation*). We'll also use the *n largest* function to extract a percentage of the most important sentences.

Our algorithm will use the following steps:

- Tokenize the text with the SpaCy pipeline. This segments the text into words, punctuation, and so on, using grammatical rules specific to the English language.
- Count the number of times a word is used (not including stop words or punctuation), then normalize the count. A word that's used more frequently has a higher normalized count.
- Calculate the sum of the normalized count for each sentence.
- Extract a percentage of the highest ranked sentences. These serve as our summary.

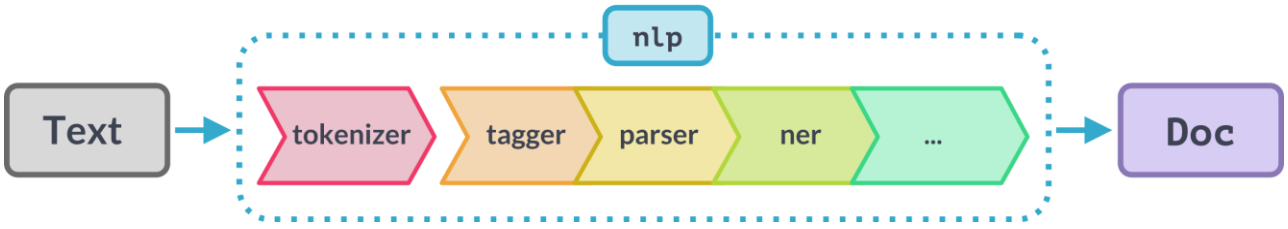


Figure 4.2.3 Summarization of Text

4.2.4 Text Translation

We imported the required package. we created an instance of Translator class and assigned it to translator variable. we performed the translation of the text. *(If the source language is not given, google translate attempts to detect the source language. If the target language is not given, it will translate your text to English by default.)*

we printed the translated text. we translated the text in *Korean* to *English* by specifying the destination, or target language, as en. we printed the translated text. Likewise we can also translate the English text to Tamil and any other languages by specifying destination as the language we want.

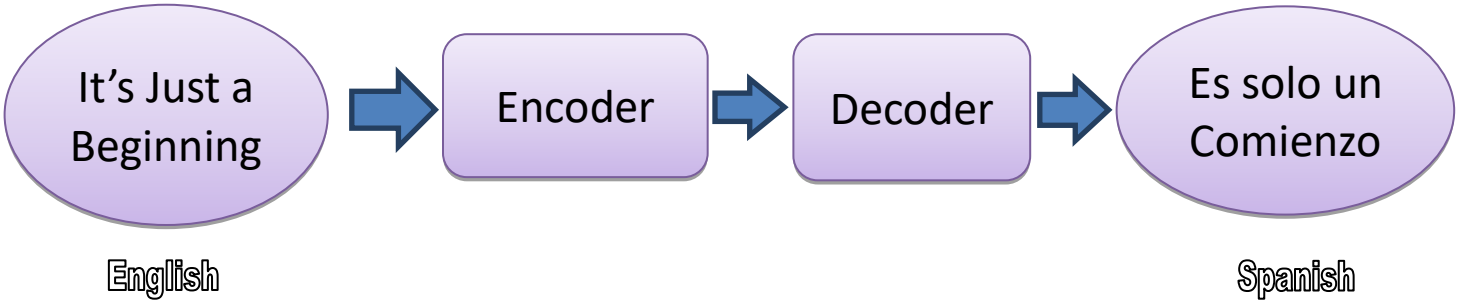


Figure 4.2.4 Text Translation

4.2.5 Speech Representation

First of all, we're going to import the pyttsx3 package that we've installed using the pip. *Pyttsx3 Engine Initialization engine = pyttsx3.init()

The above code initializes the pyttsx3 package. The Instance of the initialized pyttsx3 package is stored in the engine variable. We are calling the variable engine as it works as the engine and converts Text-To-Speech whenever execute the functions from the package.

Say Function in pyttsx3

*engine.say("This is Text-To-Speech Engine Pyttsx3")

There is a built-in say() function in the pyttsx3 package that takes a string value and speaks it out.

*runAndWait Function

*engine.runAndWait()

This function keeps track when the engine starts converting text to speech and waits for that much time, and do not allow the engine to close. After all the processes are over, we shut down the engine by calling the *stop() function.

4.3 PREDICTION MODELS

4.3.1 To Extract the text from the scene image :

Many deep learning models in computer vision have been used for the scene text detection task over the past few years. However, the performance of these models is not up to the mark when the text in the image is skewed or curved. The CRAFT model performs well on even curved, long and deformed texts in addition to normal text. The CRAFT text detection model uses a convolutional neural network model to calculate region scores and affinity scores. The region score is used to localize character regions while the affinity score is used to group the characters into text regions. CRAFT uses a fully convolutional neural network based on the VGG16 model. The inference is done by providing word-level bounding boxes. The CRAFT text detection model works well at various scales, from large to small texts and is shown to be effective on unseen datasets as well. The time taken by CTPN, EAST and MSER text detection methods is lower compared to the CRAFT text detection engine. However, CRAFT is more accurate and the bounding boxes are more precise when the text is long, curved, rotated or deformed.

4.3.2 To Recognize the text from the bounded region of image:

After the text detection step, regions, where the text is present, are cropped and sent through convolutional layers to get the features from the image. Later these features are fed to many-to-many LSTM architecture, which outputs soft max probabilities over the vocabulary. These outputs from different time steps are fed to the CTC

decoder to finally get the raw text from images. One can relate this to training any LSTM model with word embeddings like word2vec, Glove, fast Text. In the modelling, we are making a sequential model. The first layer of the model is the embedding layer which uses the 32 length vector, and the next layer is the LSTM layer which has 100 neurons whichwill work as the memory unit of the model.

After LSTM, the dense layer which is an output layer with sigmoid function, sigmoid function helps in providing the labels.

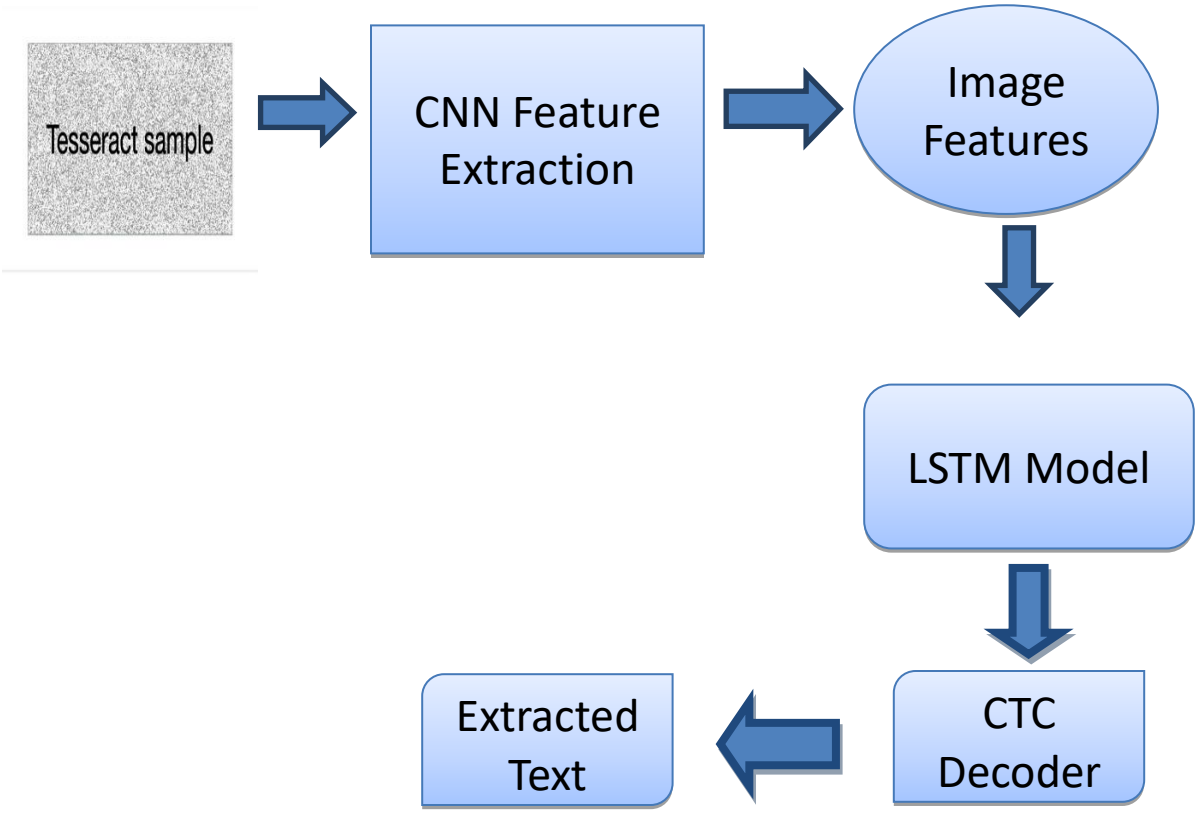


Figure 4.3.2 LSTM Architecture

```
 lstm_1 (LSTM)               (None, 50, 50)            20200

 lstm_2 (LSTM)               (None, 50)                20200

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 50,851
Trainable params: 50,851
Non-trainable params: 0
_________________________________________________________________
```

```
In [22]: history = model.fit(X_train,y_train,validation_data=(X_test,ytest),epochs=5,batch_size=32,verbose=1)

Epoch 1/5
230/230 [==============================] - 21s 92ms/step - loss: 0.1296 - accuracy: 0.8294 - val_loss: 0.1468 - val_accuracy:
0.8089
Epoch 2/5
230/230 [==============================] - 19s 82ms/step - loss: 0.1283 - accuracy: 0.8340 - val_loss: 0.1408 - val_accuracy:
0.8180
Epoch 3/5
230/230 [==============================] - 18s 80ms/step - loss: 0.1273 - accuracy: 0.8344 - val_loss: 0.1423 - val_accuracy:
0.8167
Epoch 4/5
230/230 [==============================] - 19s 81ms/step - loss: 0.1271 - accuracy: 0.8365 - val_loss: 0.1414 - val_accuracy:
0.8188
Epoch 5/5
230/230 [==============================] - 19s 84ms/step - loss: 0.1270 - accuracy: 0.8355 - val_loss: 0.1402 - val_accuracy:
0.8193
```

```
In [23]: print(f'\n Accuracy is: {model.evaluate(X_train, y_train)[1]*100}')

230/230 [==============================] - 6s 24ms/step - loss: 0.1259 - accuracy: 0.8372

 Accuracy is: 83.71712565422058
```

Figure 4.3.2.1 Training validation data for LSTM model

4.3.3 To Tokenize the recognized text to the specified amount :

The NegEx algorithm relies on four types of lexical cues: negation triggers that indicate a negation (e.g., “denies”), pseudo-negation triggers that contain negation triggers but do not negate the clinical condition (e.g., “no increase”), and termination terms that stop the scope of the negation trigger (e.g., “but”). Any clinical condition within the scope of a negation trigger is negated. All NegEx lexical items have an action—negation and pseudo-negation triggers can modify information to the right of the term (i.e., forward in the sentence) or to the left of the term (i.e., backward in the sentence). Termination terms have an action to terminate scope, which otherwise terminates at the end of the sentence. Pseudo negation triggers attempt to compensate for NegEx’s lack of syntax by listing exceptions to the occurrence of negation triggers such as “no” in phrases like “no previous”. Because the lexicon is the keystone of the algorithm, adaptation of Neg Ex to other languages relies mainly on translation of the lexical cues.

```
In [15]: max_frequency = max(word_frequencies.values())
         max_frequency

Out[15]: 15

In [16]: for word in word_frequencies.keys():
             word_frequencies[word] = word_frequencies[word]/max_frequency

         print(word_frequencies)
```

```
{'fact': 0.06666666666666667, 'rated': 0.06666666666666667, 'translation': 0.6, 'tools': 0.06666666666666667, 'comes': 0.066666
66666666667, 'accuracy': 0.4, 'exact': 0.06666666666666667, 'depend': 0.06666666666666667, 'language': 0.13333333333333333, 'pa
irs': 0.06666666666666667, 'chosen': 0.06666666666666667, ' ': 0.8666666666666667, 'said': 0.06666666666666667, '100': 0.066666
66666666667, 'perfect': 0.06666666666666667, 'actually': 0.2, 'accurate': 0.06666666666666667, 'service': 0.06666666666666667,
'want': 0.06666666666666667, 'consider': 0.06666666666666667, 'situations': 0.06666666666666667, 'tradeoffs': 0.066666666666666
67, 'alternative': 0.06666666666666667, 'Google': 1.0, 'Translate': 1.0, 'probably': 0.06666666666666667, 'going': 0.0666666666
6666667, 'best': 0.06666666666666667, 'option': 0.06666666666666667, 'people': 0.06666666666666667, 'Want': 0.066666666666666
7, 'details': 0.2, 'reading': 0.06666666666666667, 'post': 0.06666666666666667, 'learn': 0.06666666666666667, 'gory': 0.0666666
6666666667, 'Work': 0.06666666666666667, 'talk': 0.06666666666666667, 'reliable': 0.06666666666666667, 'let': 0.066666666666666
67, 'quickly': 0.06666666666666667, 'run': 0.06666666666666667, 'basic': 0.06666666666666667, 'works': 0.06666666666666667, 'sc
enes': 0.06666666666666667, 'important': 0.13333333333333333, 'talking': 0.06666666666666667, 'evolution': 0.06666666666666667,
'way': 0.06666666666666667, 'performs': 0.06666666666666667, 'translations': 0.13333333333333333, 'insight': 0.0666666666666666
7, 'gotten': 0.06666666666666667, 'lot': 0.13333333333333333, 'better': 0.06666666666666667, 'late': 0.06666666666666667, '201
6': 0.13333333333333333, 'backend': 0.06666666666666667, 'understand': 0.06666666666666667, 'different': 0.06666666666666667,
'options': 0.06666666666666667, 'user': 0.06666666666666667, 'Old': 0.06666666666666667, 'Approach': 0.06666666666666667, 'pas
t': 0.13333333333333333, 'fairly': 0.06666666666666667, 'simple': 0.06666666666666667, 'model': 0.13333333333333333, 'struggle
d': 0.06666666666666667, 'old': 0.13333333333333333, 'system': 0.13333333333333333, 'worked': 0.06666666666666667, 'entered':
0.06666666666666667, 'text': 0.2, 'translate': 0.13333333333333333, 'searched': 0.06666666666666667, 'huge': 0.0666666666666666
7, 'database': 0.06666666666666667, 'find': 0.06666666666666667, 'human': 0.06666666666666667, 'translated': 0.1333333333333333
3, 'documents': 0.06666666666666667, 'selected': 0.06666666666666667, 'pair': 0.06666666666666667, 'detected': 0.06666666666666
667, 'frequently': 0.06666666666666667, 'version': 0.06666666666666667, 'word': 0.3333333333333333, 'phrase': 0.2, 'supplied':
0.06666666666666667, 'needed': 0.06666666666666667, 'repeated': 0.06666666666666667, 'fully': 0.06666666666666667, 'content':
0.06666666666666667, 'called': 0.06666666666666667, 'statistical': 0.06666666666666667, 'machine': 0.06666666666666667, 'SMT':
```

Figure 4.3.3 Tokenization of Words using NegEX

4.4 TESTING

Software Testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding, Testing presents an interesting anomaly for the software engineer. The benefits of testing include preventing bugs, reducing development costs and improving performance. A variety of tests can be performed and they will have been designed to find as many errors or bugs as possible.

4.4.1 Testing Objectives

1. Testing is a process of executing a program with the intent of finding an error
2. A good test case is one that has a probability of finding an as yet undiscovered error.
3. A successful test is one that uncovers an undiscovered error.

4.4.2 Test Cases

S. NO	Test Description	Test Procedure	Test Input	Expected Result	Actual Result
1	To Detect the text from the images or video frames	User need to import the image from the directory or webcam	Input image from actual directory or cam	System can find the text from the specified image or webcam	System found the texts from the input image
2	To summarize the detected text to minimum amount we want	User need to concise the amount they need using the percentage	Detected text from the previous module	System can summarize the content into small amount	Make a little content from the detected part of texts
3	To Translate the minimized part to the flexible language we known	User need to input the symbol of the language we want in the code	Summarized text from the previous module	System can translate the content to the lang	System translated the content to the specified language
4	To represent the translated part in an audio format	User need to input the content that we want to listen	Translated content from the previous module	System can speech the content of the translated part	System addressed the content of the translated part

Table 4.4 Test cases

CHAPTER 5

RESULTS

In this project we have evaluated different machine learning modules for the detection and summarization of text from the video frames. Our focus is to find out the most suitable module that can predict the textual characters and summarize and translate that predicted characters more effectively .

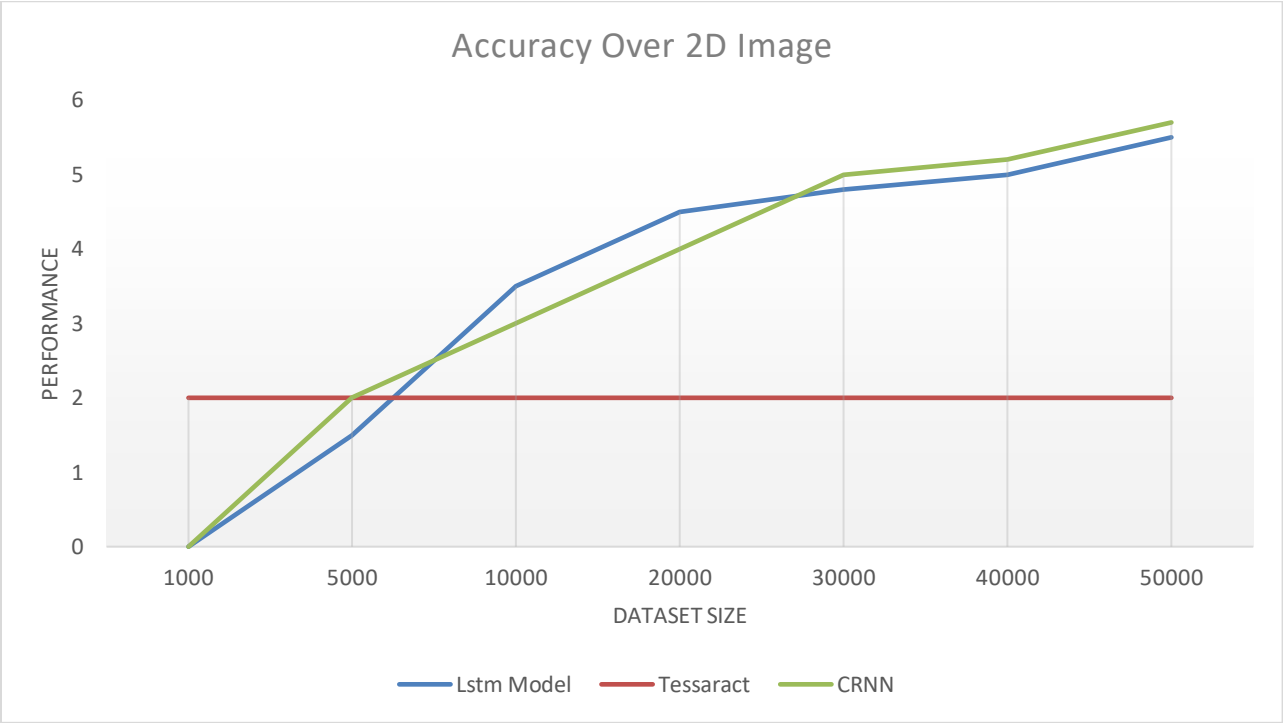


Figure 5.1 graph analysis of LSTM model with Tesseract

Text detection in natural scenes is a complex task, requires multi-orientation datasets to represent real-world scenarios. It requires text orientation prediction along with box co-ordinates. CRAFT is very accurate and fast enough: 8.6 FPS. It is multi-lingual. CRAFT is available in OpenCV, usage, and deployment is easy.

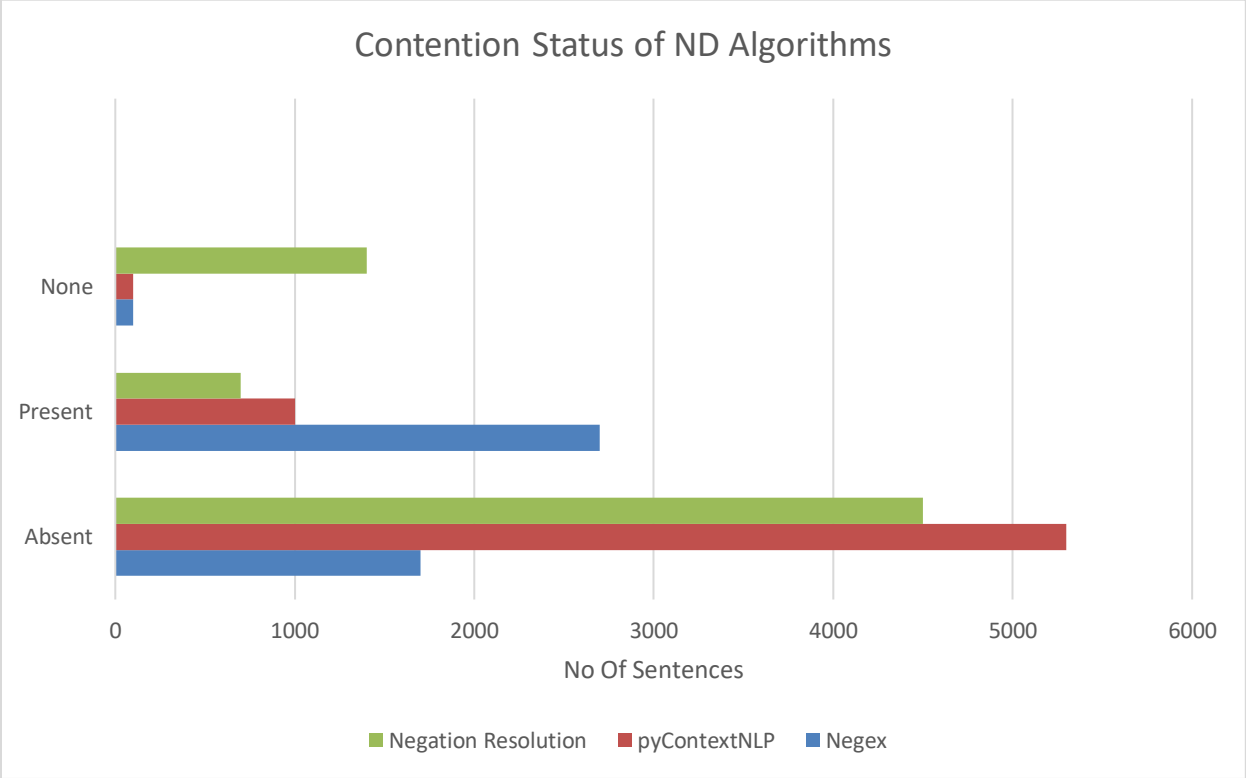


Figure 5.2 graph analysis of NegEx algorithm with other modules

NegEx has been recently ported and evaluated on clinical texts in Swedish [6] and French [7] and has shown good performance (recall 82%; precision 75%) for Swedish assessment sections of the Stockholm EPR corpus and better performance (recall 85%; precision 89%) for French cardiology notes. In both studies, the adapted NegEx systems achieved comparable recall to the English NegEx with observable differences in precision (differences of−9.3% and 4.4%, respectively).

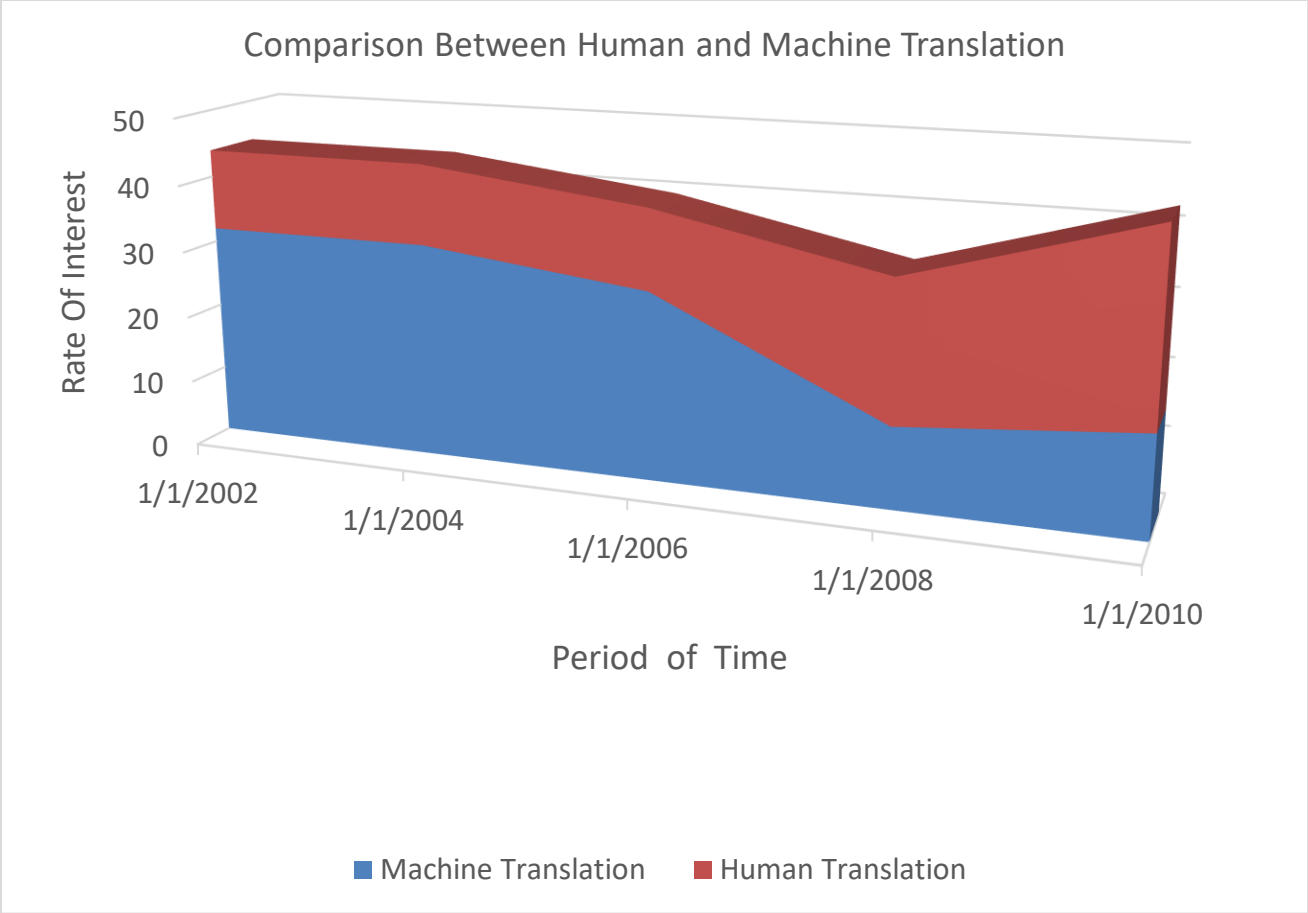


Figure 5.3 Comparison of Machine Translation with Human Translation

While Google Translate is available in more than 130 languages—making it a translation tool with the broadest range of support—it also varies in terms of accuracy rate. For instance, since Spanish is one of its most popular languages, its translation accuracy is typically over 90%. In fact, a 2014 study found Google Translate to have only 57.7% accuracy when used to translate complex medical phrases. A 2021 study conducted by the UCLA Medical Center found that Google Translate preserved the overall meaning for 82.5% of the translations. But the accuracy between languages spanned 55% to 94%.

5.1 PERFORMANCE METRICES

In order to evaluate the implemented models, we used as metrics the Accuracy, Precision, Recall and F1- Score. These metrics are based on the assessment of True positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) values.

5.1.1 Accuracy

The ratio of number of correct predictions to the total number of predictions.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

5.1.2 Precision

The precision metric is used to overcome the limitation of Accuracy. The precision determines the proportion of positive prediction that was actually correct.

$$\text{Precision} = \frac{TP}{TP + FP}$$

5.1.3 Recall

It is also similar to the Precision metric; however, it aims to calculate the proportion of actual positive that was identified incorrectly.

$$\text{Recall} = \frac{TP}{TP + FN}$$

5.1.4 F-Scores

F-score or F1 Score is a metric to evaluate a binary classification model on the basis of predictions that are made for the positive class. It is calculated with the help of Precision and Recall.

$$F1\ Score = \frac{2 * precision * recall}{precision + recall}$$

5.1.5 Confusion Matrix

A confusion matrix is a tabular representation of prediction outcomes of any binary classifier, which is used to describe the performance of the classification model on a set of test data when true values are known.

Table 5.1 Performance Metrices

S.NO	ALGORITHM	PRECISION	RECALL	F_SCORE	ACCURACY (%)
1	CRAFT	0.90	0.92	0.91	91
2	LSTM	0.98	0.94	0.96	95
3	NegEX	0.87	0.90	0.89	88
4	GNMT	0.92	0.93	0.96	95

The accuracy of the three algorithms was compared in order to determine which would be most appropriate for the detection and summarization. Craft model, LSTM model and Negation X all provided accuracy that are, respectively, 91%, 95%, 88%, and the default GNMT picked from already existing model and giving accuracy of 95%. So, using the above algorithms and the features from this dataset, text can be easily recognized and summarized with almost pinpoint accuracy.

CHAPTER 6

CONCLUSION

In this project, we presented a comprehensive framework for text detection and recognition in video frames containing textual occurrences in English and other languages. A number of contributions are made in the presented study, We developed a comprehensive dataset of video frames with ground truth information allowing evaluation of detection and recognition tasks. For detection of textual regions, we employed state-of-the-art deep learning-based object detectors and fine-tuned them to detect text in multiple scripts. Script of the detected textual regions is identified using CNNs in a classification framework. A combination of CNN and bidirectional LSTMs which reports high recognition rates for the challenging video text in English. For summarization of detected text, we employed a negation method using NegEx which avoiding unwanted phrase, extra symbols and repeated words. Implementation of summarising the text is helpful in understand the whole context of the text or paragraph. For translation process, we earned a process of google translation which have already been worked in a highest accuracy. For the speech Representation, we just developed a python code to given in an audio format.

In our further work on this problem, we intend to develop a complete indexing and retrieval system that can be queried for keywords. The system will be optimized to work on live streams in addition to archived videos. This will in turn allow development of keyword based alert generation systems. Furthermore, the dataset is also planned to be enhanced and made available publicly. The study can also be extended to include additional scripts by integrating their respective OCRs. The development of this application which will be great help to people with visual impairment and those who don’t know other language.

REFERENCES

[1] Jamil, A. Batool, Z. Malik, A. Mirza, I. Siddiqi, “Multilingual artificial text extraction and script identification from video images.” Int. J. Adv. Comput. Sci. Appl. 1(7), 529–539 (2016)

[2] J. Hochberg, L. Kerns, P. Kelly, T. Thomas, in Document Analysis and Recognition, 1995., Proceedings of the Third International Conference On, vol. 1. “Automatic script identification from images using cluster-based templates” (IEEE, Montreal, 1995), pp. 378–381

[3] A. L. Spitz, “Determination of the script and language content of document images.” IEEE Trans. Patt. Anal. Mach. Intell. 19(3), 235–245 (1997)

[4] U. Pal, B. Chaudhuri, in Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference On. “Automatic identification of English, Chinese, Arabic, Devnagari and Bangla script line” (IEEE, Washington, 2001), pp. 790–794

[5] C. Zhu, W. Wang, Q. Ning, in Advances in Multimedia Information Processing-PCM 2006. “Text detection in images using texture feature from strokes” (Springer, Hangzhou, 2006), pp. 295–301

[6] Z. Li, G. Liu, X. Qian, D. Guo, H. Jiang, “Effective and efficient video text extraction using key text points. IET Image Process.” 5(8), 671–683 (2011)

[7] N. Sharma, S. Chanda, U. Pal, M. Blumenstein, in Document Analysis and Recognition (ICDAR), 2013 12th International Conference On. “Word-wise script identification from video frames” (IEEE, Washington, 2013), pp. 867–871

[8] G. Peake, T. Tan, in Document Image Analysis, 1997.(DIA’97) Proceedings., Workshop On. “Script and language identification from document images” (IEEE, San Juan, 1997), pp. 10–17

[9] P. Shivakumara, N. Sharma, U. Pal, M. Blumenstein, C. L. Tan, in Pattern Recognition (ICPR), 2014 22nd International Conference On. “Gradient-angular features for word-wise video script identification” (IEEE, Stockholm, 2014), pp. 3098–3103

[10] N. Sharma, U. Pal, M. Blumenstein, in Neural Networks (IJCNN), 2014 International Joint Conference On. “A study on word-level multi-script identification from video frames” (IEEE, Beijing, 2014), pp. 1827–1833

[11] N. Sharma, R. Mandal, R. Sharma, U. Pal, M. Blumenstein, in Document Analysis and Recognition (ICDAR), 2015 13th International Conference On. Icdar2015 “competition on video script identification” (CVSI 2015) (IEEE, Nancy, 2015), pp. 1196–1200

[12] J. Mei, L. Dai, B. Shi, X. Bai, in 2016 23rd International Conference on Pattern Recognition (ICPR). “Scene text script identification with convolutional recurrent neural networks” (IEEE, Cancún, 2016), pp. 4053–4058

[13] A. K. Singh, A. Mishra, P. Dabral, C. Jawahar, in Document Analysis Systems (DAS), 2016 12th IAPR Workshop On. “A simple and effective solution for script identification in the wild” (IEEE, Santorini, 2016), pp. 428–433

[14] L. Gomez, D. Karatzas, in Document Analysis Systems (DAS), 2016 12th IAPR Workshop On. “A fine-grained approach to scene text script identification” (IEEE, Santorini, 2016), pp. 192–197

[15] M. Tounsi, I. Moalla, F. Lebourgeois, A. M. Alimi, in International Conference on Neural Information Processing. “CNN based transfer learning for scene script identification” (Springer, California, 2017), pp. 702–711

[16] L. Gomez, A. Nicolaou, D. Karatzas, “Improving patch-based scene text script identification with ensembles of conjoined networks.” Patt. Recogn. 67, 85–96 (2017)

APPENDIX I

SOURCE CODE

```
# Image to Text Detection.
import matplotlib.pyplot as plt
import cv2
import easyocr
import pyttsx3
text=''

for i in range(len(result)):
    text=text+' '+result[i][1]

print(text)
print(result[0][1])
top_left=tuple(result[0][0][0])
bottom_right=tuple(result[0][0][2])
text=result[0][1]
font=cv2.FONT_HERSHEY_SIMPLEX
img=cv2.imread('image1.jpg')
img=cv2.rectangle(img, top_left, bottom_right, (0,255,0), 5)
img=cv2.putText(img, text, top_left,font, 1, (0,0,255),2, cv2.LINE_AA)
plt.imshow(img)
plt.show()
# Handling Multiple lines

img=cv2.imread('image1.jpg')
for detection in result:
    top_left=tuple([int(val) for val in detection[0][0]])
    bottom_right=tuple([int(val) for val in detection[0][2]])
    text=detection[1]
    print(text)
    font=cv2.FONT_HERSHEY_SIMPLEX
```

```
    img=cv2.rectangle(img, top_left, bottom_right, (0,255,0), 5)
    img=cv2.putText(img, text, top_left,font, 1, (0,0,255),2, cv2.LINE_AA)

plt.figure(figsize=(10,10))
plt.imshow(img)
plt.show()
# Reading the text and converting into scentense

# Handling Multiple lines
import cv2
img=cv2.imread('image1.jpg')
text=''
for detection in result:
    top_left=tuple([int(val) for val in detection[0][0]])
    bottom_right=tuple([int(val) for val in detection[0][2]])
    text=text+' '+detection[1]
print(text)

# Summarization
import spacy
from spacy.lang.en.stop_words import STOP_WORDS
from string import punctuation
nlp=spacy.load("en_core_web_sm")
tokens = [token.text for token in doc]
punctuation = punctuation + '\n'
punctuation
word_frequencies = {}
for word in doc:
    if word.text.lower() not in stopwords:
        if word.text.lower() not in punctuation:
            if word.text not in word_frequencies.keys():
                word_frequencies[word.text] = 1
            else:
                word_frequencies[word.text] += 1
```

```
print(word_frequencies)
for word in word_frequencies.keys():
    word_frequencies[word] = word_frequencies[word]/max_frequency

print(word_frequencies)
sentence_scores = {}
for sent in sentence_tokens:
    for word in sent:
        if word.text.lower() in word_frequencies.keys():
            if sent not in sentence_scores.keys():
                print(f'sent{sent} ')
                sentence_scores[sent] = word_frequencies[word.text.lower()]
            else:
                sentence_scores[sent] += word_frequencies[word.text.lower()]

sentence_scores
for i in range(100,1,-1):
    z=1/i
    select_length = int(len(sentence_tokens)*z)
    summary = nlargest(select_length, sentence_scores, key = sentence_scores.get)
    final_summary = [word.text for word in summary]
    summary = ' '.join(final_summary)
    if len(summary)>1:
        b=z
        break
# select_length = int(len(sentence_tokens)*b)
select_length = int(len(sentence_tokens)*0.2)
select_length
summary = nlargest(select_length, sentence_scores, key = sentence_scores.get)
final_summary = [word.text for word in summary]
summary = ' '.join(final_summary)
final_summary = [word.text for word in summary]
summary = ' '.join(final_summary)
```

```
# Translation
import pyttsx3
from googletrans import Translator

translator = Translator()

while True:
    sent = str(input("Enter the text : "))
    sent = sent.lower()
    if sent == 'stop':
        break
    else:
        out = translator.translate(sent, dest='hi')
        val = out.text
        print(val)

from googletrans import Translator



while True:
    sent = str(input("Enter the text : "))
    sent = sent.lower()
    if sent == 'stop':
        break
    else:
        out = translator.translate(sent, dest='ta')
        val = out.text
        print(val)
# Speech Representation
import pyttsx3
engine = pyttsx3.init()
engine.setProperty('rate', 130)
engine.say(text)
engine.runAndWait()
```

APPENDIX 2

SCREENSHOTS

Text Detection From the image :

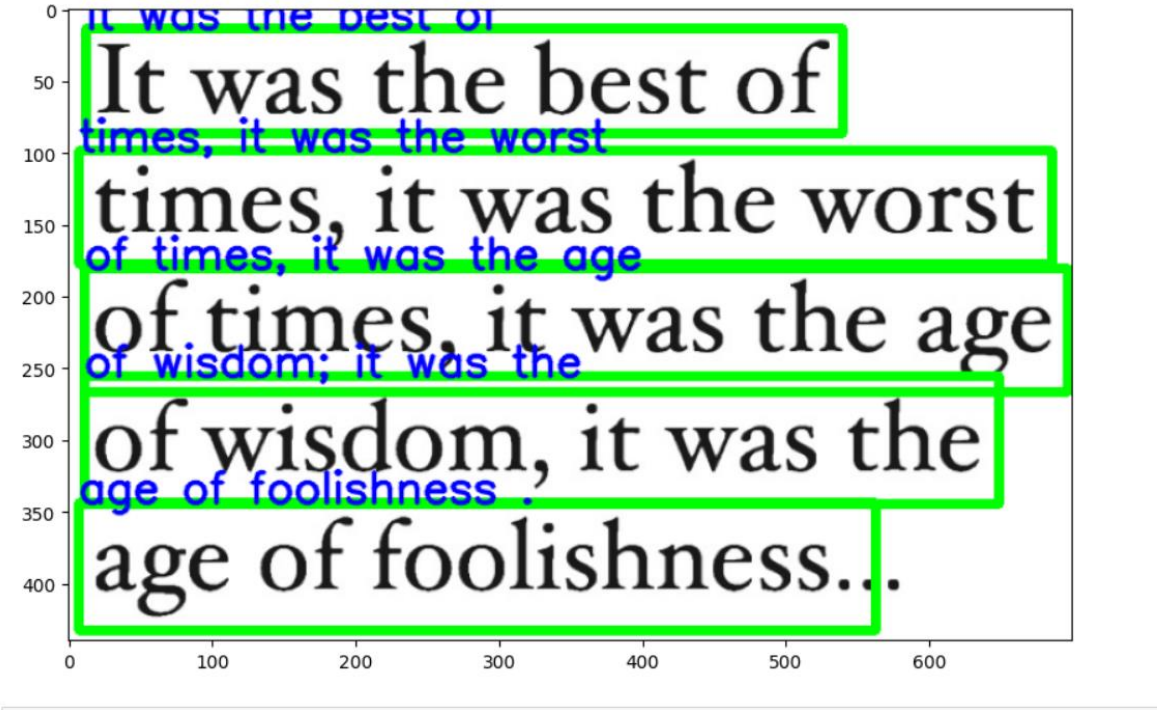


Figure (a)

```
The human brain
can
process entire
images that the eye
sees for as little as
13 milliseconds.
Source: Trafton; Anne-
"In the Blink of an Eye
'MIT News; 16 Jan: 2014, news mit edu/2014/in-the-blink-of-an-eye-0116.
```

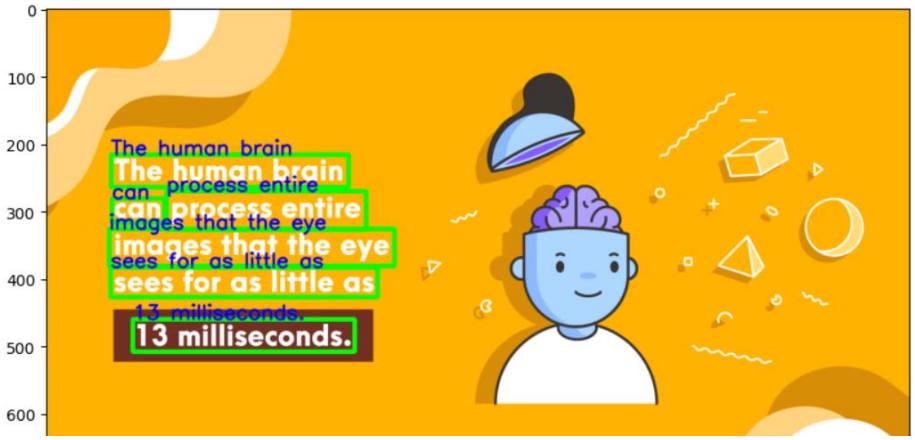


Figure (b)

```
In [50]: print(text)
```

```
Assuming leadership of the Indian National Congress in 1921, Gandhi led nationwide campaigns for easing poverty, expanding wome
n's rights, building religious and ethnic amity, ending untouchability, and, above all, achieving swaraj or self-rule. Gandhi a
dopted the short dhoti woven with hand-spun yarn as a mark of identification with India's rural poor. He began to live in a sel
f-sufficient residential community, to eat simple food, and undertake long fasts as a means of both introspection and political
protest. Bringing anti-colonial nationalism to the common Indians, Gandhi led them in challenging the British-imposed salt tax
with the 400 km (250 mi) Dandi Salt March in 1930 and in calling for the British to quit India in 1942. He was imprisoned many
times and for many years in both South Africa and India.  Gandhi's vision of an independent India based on religious pluralism
was challenged in the early 1940s by a Muslim nationalism which demanded a separate homeland for Muslims within British India.
[10] In August 1947, Britain granted independence, but the British Indian Empire[10] was partitioned into two dominions, a Hind
u-majority India and a Muslim-majority Pakistan.[11] As many displaced Hindus, Muslims, and Sikhs made their way to their new l
ands, religious violence broke out, especially in the Punjab and Bengal. Abstaining from the official celebration of independen
ce, Gandhi visited the affected areas, attempting to alleviate distress. In the months following, he undertook several hunger s
trikes to stop the religious violence. The last of these, begun in Delhi on 12 January 1948 when he was 78,[12][13][14][15] als
o had the indirect goal of pressuring India to pay out some cash assets owed to Pakistan,[16][17][18][19] which the Indian gove
rnment,[20] had been resisting.[21][22][23][24][25] Although the Government of India relented,[26] as did the religious rioter
s, the belief that Gandhi had been too resolute in his defence of both Pakistan and Indian Muslims, spread among some Hindus in
India.[27][28][15] Among these was Nathuram Godse,[29] a militant Hindu nationalist from Pune, western India,[30][31] who assas
sinated Gandhi by firing three bullets into his chest at an interfaith prayer meeting in Delhi on 30 January 1948
```

```
In [51]: print(summary)
```

```
Assuming leadership of the Indian National Congress in 1921, Gandhi led nationwide campaigns for easing poverty, expanding wome
n's rights, building religious and ethnic amity, ending untouchability, and, above all, achieving swaraj or self-rule. Bringing
anti-colonial nationalism to the common Indians, Gandhi led them in challenging the British-imposed salt tax with the 400 km (2
50 mi) Dandi Salt March in 1930 and in calling for the British to quit India in 1942.
```

```
In [52]: len(text)
Out[52]: 2272

In [53]: len(summary)
Out[53]: 467
```

Figure (c) Text Summarization from the Detected Text

```
        out = translator.translate(sent, dest='EN')
        val = out.text
        print(val)
        engine = pyttsx3.init()
        engine.setProperty('rate', 130)
        engine.say(val)
        engine.runAndWait()
```

```
Enter the text : इस पृष्ठ पर इन्टरनेट पर उपलब्ध विभिन्न हिन्दी एवं देवनागरी सम्बंधित साधनों की कड़ियों की सूची है। इसमें ऑनलाइन एवं ऑफ़लाइन उपकरण
(टूल्स) शामिल हैं।
This page on the internet is a list of links to various resources related to Hindi and the Devanagari. This includes online and
offline tools.
Enter the text : STOP
```

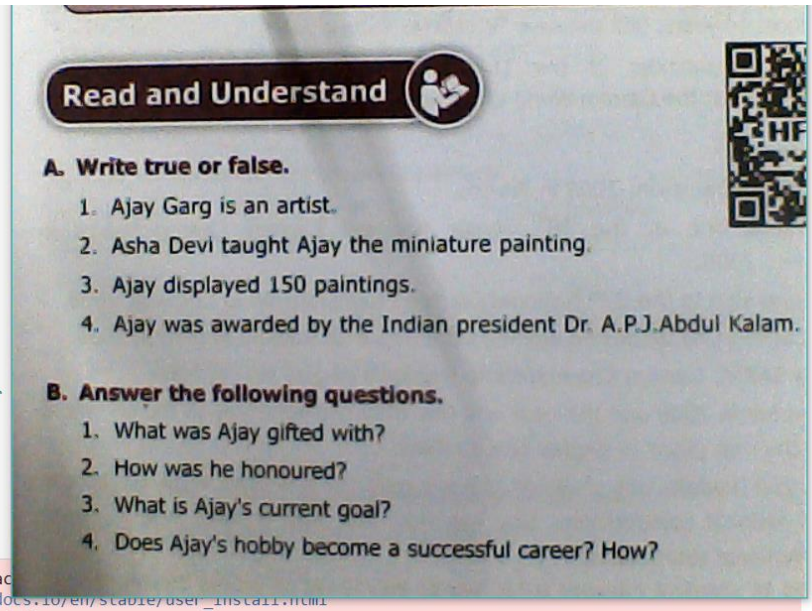
English to Tamil Translate

```
In [*]: from googletrans import Translator

translator = Translator()

while True:
    sent = str(input("Enter the text : "))
    sent = sent.lower()
    if sent == 'stop':
        break
    else:
        out = translator.translate(sent, dest='ta')
        val = out.text
        print(val)
```

```
Enter the text : These are short, famous texts in English from classic sources like the Bible or Shakespeare. Some texts have w
ord definitions and explanations to help you. Some of these texts are written in an old style of English.
இவை பைபிள் அல்லது ஷேக்ஸ்பியர் போன்ற உன்னதமான மூலங்களிலிருந்து ஆங்கிலத்தில் உள்ள குறுகிய, பிரபலமான நூல்கள். சில உரைகளில் உங்களுக்கு உதவ வார்த்தை வரையறைகள் மற்றும் விளக்கங்கள் உள்ளன. இந்த நூல்களில் சில பழைய ஆங்கில பாணியில் எழுதப்பட்டுள்ளன.
```

Figure (d) Text Translation to the Specified Language

```
            import pyttsx3
            print(text)
            engine = pyttsx3.init()
            engine.setProperty('rate', 130)
            engine.say(text)
            engine.runAndWait()

        except:
            pass

    if text==False:
        pass


    cv2.imshow('video', img)

    # Listen to the keyboard for presses.
    keyboard_input = cv2.waitKey(1)
    # 27 is the ASCII for the esc key on your
    if keyboard_input == 27:
        break
    if keyboard_input == ord('t'):
        text=True

camera.release()
cv2.destroyAllWindows()
```




```
C:\Users\KAAD\.conda\envs\project\lib\site-pac
d ipywidgets. See https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
Using CPU. Note: This module is much faster with a GPU.
```

```
 Read and Understand Write true or false: Ajay is an artist- Asha Devi taught Ajay the minlature painting  Ajay displayed 150 p
aintings. Ajay was awarded by the Indian president Dr: A.PJ Abdul Kalam Answer the following questons: What was Ajay gifted wit
h? How was he honoured? What is Ajays current goal? Does Ajay $ hobby become a successful career? How? Garg
```

Figure (e) Text Detection from the Video frames or Webcam

```
summary = nlargest(select_length, sentence_scores, key = sentence_scores.get)
final_summary = [word.text for word in summary]
summary = ' '.join(final_summary)

summary = nlargest(select_length, sentence_scores, key = sentence_scores.get)

final_summary = [word.text for word in summary]
summary = ' '.join(final_summary)

print('\n\nSUMMARY',summary)

engine.say(summary)
engine.runAndWait()
```

```
C:\Users\KAAD\.conda\envs\project\lib\site-packages\tqdm\auto.py:21: TqdmWarning: IProgress not found. Please update jupyter an
d ipywidgets. See https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
Using CPU. Note: This module is much faster with a GPU.
```

```
ORIGINAL  Dies ist ein Blindtext. An ihm lasst sich vieles uber die Schrift ablesen;, in der er ge- setzt ist Auf den ersten Bl
ick wird der Grauwert der Schriftflache sichtbar: Dann kann man prufen; wie die Schrift zu lesen ist und wie sie auf den Leser
wirkt. Dies ist ein Blindtext. An ihm lasst sich vieles iber die Schrift ablesen; in der er ge- setzt ist. Auf den ersten Blick
wird der Grauwert der Schriftflache sichtbar Dann kann man prufen; wie die Schrift zu lesen ist und wie sie auf den Leser wirk
t. gut gut

 இது போலி உரை. அதில் எந்த எழுத்துரு அமைக்கப்பட்டுள்ளது என்பதைப் பற்றி நிறைய படிக்கலாம்.முதல் பார்வையில், த
ட்டச்சு முகத்தின் சாம்பல் மதிப்பு தெரியும்: பிறகு நீங்கள் சரிபார்க்கலாம். எழுத்தை எப்படி படிக்க வேண்டும், அது வாசக
னை எப்படி பாதிக்கிறது. இது போலி உரை. அவரிடமிருந்து வேதத்தைப் பற்றி அதிகம் படிக்கலாம்; அதில் அது அமைக்கப்ப
ட்டுள்ளது. உரை பகுதியின் சாம்பல் மதிப்பு முதல் பார்வையில் தெரியும். பிறகு நீங்கள் சரிபார்க்கலாம்; எழுத்தை எப்படி ப
டிக்க வேண்டும், அது வாசகனை எப்படி பாதிக்கிறது. நல்லது நல்லது


SUMMARY Auf den ersten Blick wird der Grauwert der Schriftflache sichtbar: Dann kann man prufen; wie die Schrift zu lesen ist u
nd wie sie auf den Leser wirkt. Auf den ersten Blick wird der Grauwert der Schriftflache sichtbar Dann kann man prufen; wie die
Schrift zu lesen ist und wie sie auf den Leser wirkt. An ihm lasst sich vieles uber die Schrift ablesen;, in der er ge-
```

Figure (f) All Summarization, Translation, Speech Representation in one