International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 10 Issue 5 ǁ 2022 ǁ PP. 14-20

Text to Speech Conversion using real-time OCR

Kshitija Chavan[1], Prof. Priya Tambe[2], Prof. Mithun Nair[3], Prof. Jayesh Rane[4]
[*1,2,3,4]*Department of Electronics and Telecommunication Engineering,*
Pillai HOC College of Engineering and Technology, Rasayani..

Abstract
Pattern recognition, a branch in machine learning is/can be helpful in many different ways. OCR is used to recognition of character with high accuracy. Using handheld mobile device camera for capturing an image of a printed or handwritten document to generate text from the same. On global scale there are billions of android devices running. There are about 45 million blind people and 135 million visually impaired people worldwide. Disability of visual text reading has a huge impact on the quality of life for visually disabled people. Although there have been several devices designed for helping visually disabled to see objects using an alternating sense such as sound and touch, the development of text reading device is still at an early stage. Existing systems for text recognition are typically limited either by explicitly relying on specific shapes or colour masks or by requiring user assistance or may be of high cost. Therefore, we need a low-cost system that will be able to automatically locate and read the text aloud to visually impaired persons. The main idea of this project is to recognize the text character and convert it into speech signal.

Keywords: *OCR, machine learning, text-to-speech*

Date of Submission: 18-04-2022 Date of acceptance: 03-05-2022

I. INTRODUCTION

There is a lot research work has done on Pattern Recognition which comes under Machine Learning, Artificial Intelligence. OCR well known as Optical Character Recognition is one of the leading branches of the Pattern Recognition. Optical character recognition is them mechanical or electronic conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document or a photo of document. It is widely use as form as a form of information entry from printed paper data records, whether passport documents, invoices, bank statements, computerized receipts, business cards, mail, printouts of static data or any suitable document. OCR is a field of research in pattern recognition, artificial intelligence, computer vision. The application uses a webcam to take input. Input is a binary image scanned by the webcam. The OCR engine processes the image data and converts it into a text. The Google Vision API detects the text and gives speech in output. The system uses machine learning, it takes a training data and learns from it, hence the accuracy of the output grows down the pages, pass by pass.

1.1.2 Design Methodology

Recognition of scanned document images using OCR is now generally considered to be a solved problem for some scripts. Components of an OCR system consist of optical scanning binarization, segmentation, feature extraction and recognition. With the help of a digital scanner the analog document is digitized and the extracted text will be preprocessed. Each symbol is extracted through a segmentation process. The identity of each symbol comparing the extracted features with descriptions of the symbol classes obtained through a previous learning phase. Contextual information is used to reconstruct the words and numbers of the original text. The binary image taken by the webcam is then sent to the OCR engine for the pre-processing over the image. In the pre-processing the text is recognized by different terms. The text is then further converted into speech. The final output as a speech is provided though the speaker of the desktop or laptop.

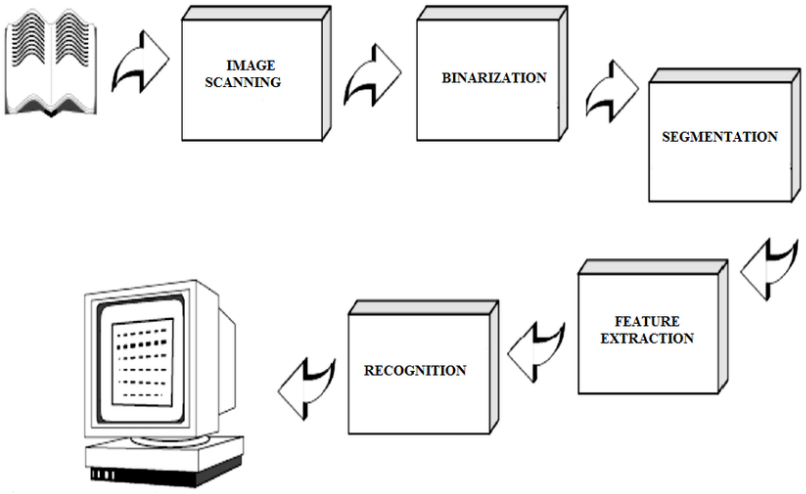


Figure1: Optical Character Recognition

1.2 Software Details

1.2.1 Open CV

OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc. In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.

The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. When OpenCV was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, and threedimension model, and so on. For example, cars can be facilitated with computer vision, which will be able to identify and different objects around the road, such as traffic lights, pedestrians, traffic signs, and so on, and acts accordingly.

(i) Grayscale

Grayscale images are those images which contain only two colors black and white. The contrast measurement of intensity is black treated as the weakest intensity, and white as the strongest intensity. When we use the grayscale image, the computer assigns each pixel value based on its level of darkness.

(ii) RGB

An RGB is a combination of the red, green, blue color which together makes a new color. The computer retrieves that value from each pixel and puts the results in an array to be interpreted.

1.2.2 Convolutional Neural Network (CNN)

T The construction of a convolutional neural network is a multi-layered feed-forward neural network, made by assembling many unseen layers on top of each other in a particular order. It is the sequential design that give permission to CNN to learn hierarchical attributes.

In CNN, some of them followed by grouping layers and hidden layers are typically convolutional layers followed by activation layers. The pre-processing needed in a ConvNet is kindred to that of the related pattern of neurons in the human brain and was motivated by the organization of the Visual Cortex.

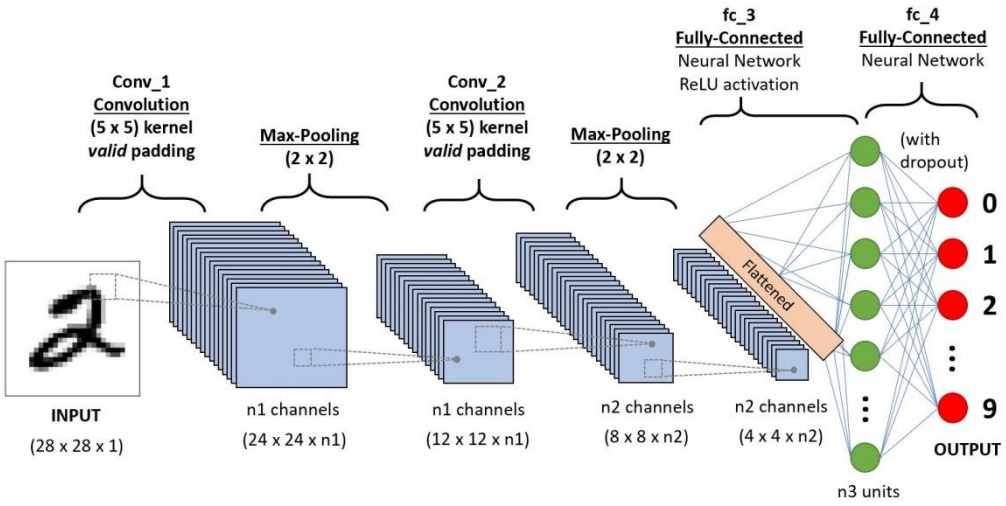


Figure 2: CNN Model

1.2.3 Google Cloud Vision

The Google Cloud Vision API enables developers to understand the content of an image by encapsulating powerful machine learning models in an easy-to-use REST API. It quickly classifies images into thousands of categories (e.g., "sailboat", "lion", "Eiffel Tower"), detects individual objects and faces within images, and finds and reads printed words contained within images. You can build metadata on your image catalog, moderate offensive content, or enable new marketing scenarios through image sentiment analysis. Analyze images uploaded in the request or integrate with your image storage on Google Cloud Storage.

1.3 Hardware Implementation

Zeb-Ultimate Pro is a full HD web camera with a quality 5P lens with a resolution of 1920x 1080.

a) Brand : ZEBRONICS

b) Special Feature : Night Vision

c) Video Capture Resolution : 1080p

d) Connector Type : USB

e) Image Capture Speed : 30 fps

f) Video Capture Format : MP4

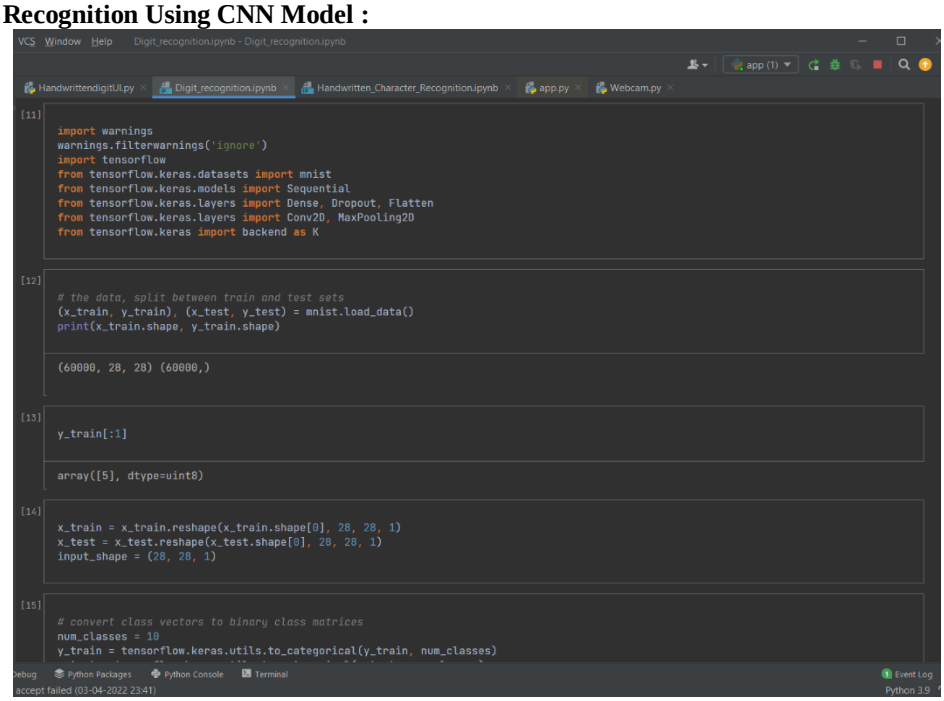
g) Maximum Focal Length : 5

h) Minimum Focal Length : 5

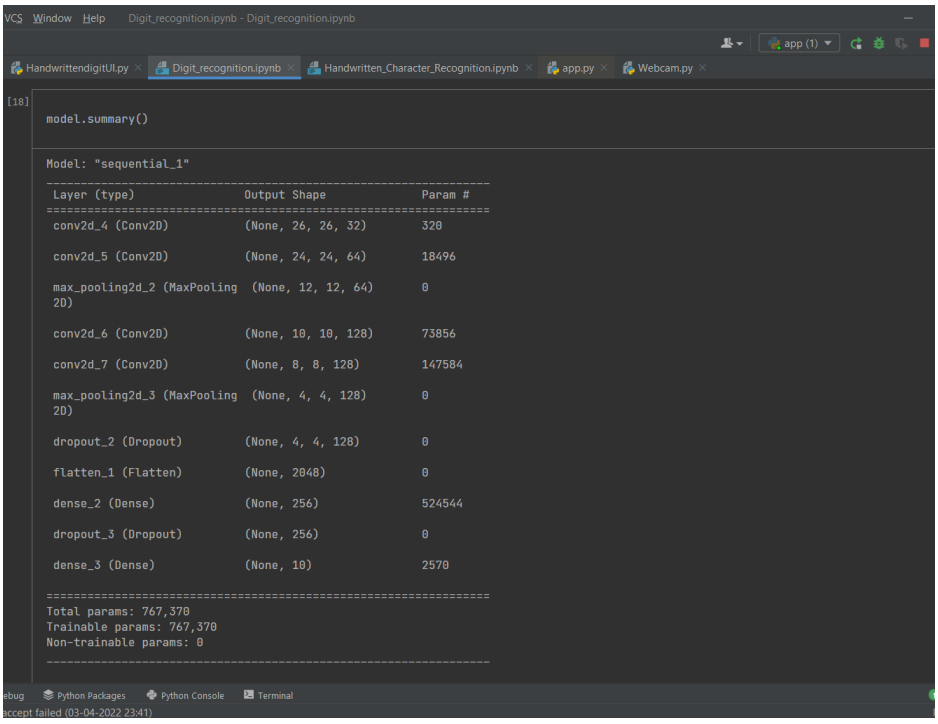
i) Item Weight : 90 Grams

II. SOFTWARE IMPLEMENTATION

2.1 Digit Recognition Using CNN Model :

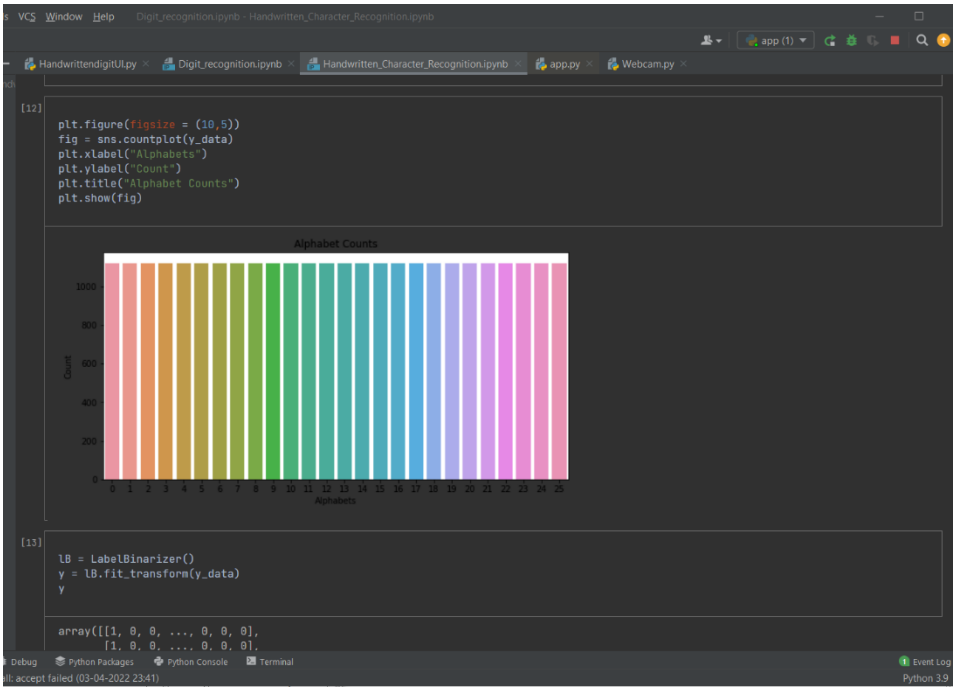


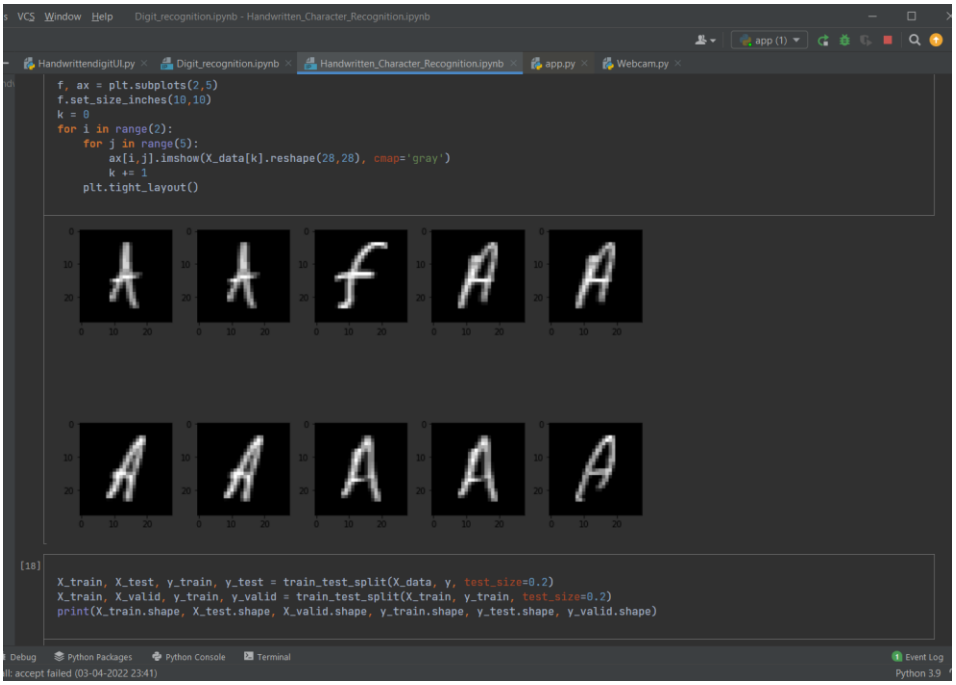
```
[11]
import warnings
warnings.filterwarnings('ignore')
import tensorflow
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras import backend as K

[12]
# the data, split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()
print(x_train.shape, y_train.shape)

(60000, 28, 28) (60000,)

[13]
y_train[:1]

array([5], dtype=uint8)

[14]
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
input_shape = (28, 28, 1)

[15]
# convert class vectors to binary class matrices
num_classes = 10
y_train = tensorflow.keras.utils.to_categorical(y_train, num_classes)
```

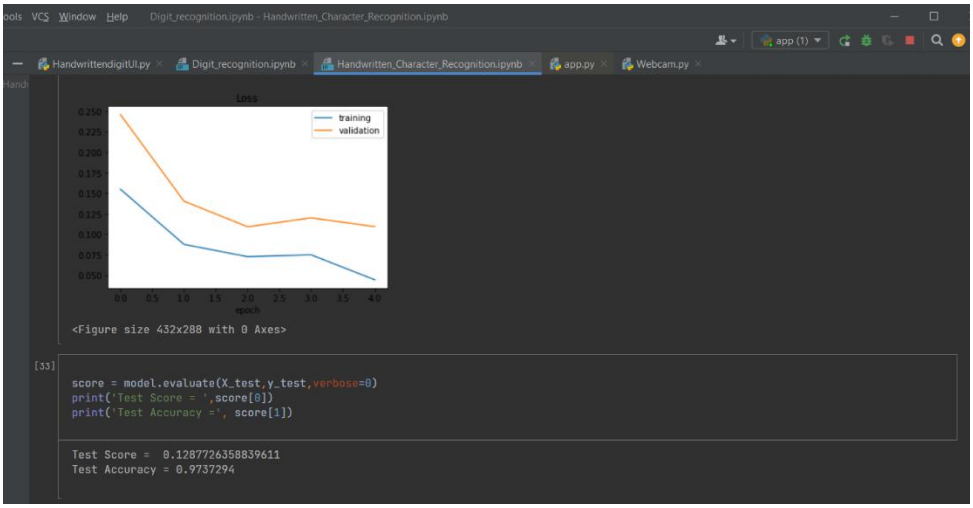


```
[18]
model.summary()

Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 26, 26, 32)        320

 conv2d_5 (Conv2D)           (None, 24, 24, 64)        18496

 max_pooling2d_2 (MaxPooling  (None, 12, 12, 64)       0
 2D)

 conv2d_6 (Conv2D)           (None, 10, 10, 128)       73856

 conv2d_7 (Conv2D)           (None, 8, 8, 128)         147584

 max_pooling2d_3 (MaxPooling  (None, 4, 4, 128)        0
 2D)

 dropout_2 (Dropout)         (None, 4, 4, 128)         0

 flatten_1 (Flatten)         (None, 2048)              0

 dense_2 (Dense)             (None, 256)               524544

 dropout_3 (Dropout)         (None, 256)               0

 dense_3 (Dense)             (None, 10)                2570

=================================================================
Total params: 767,370
Trainable params: 767,370
Non-trainable params: 0
_________________________________________________________________
```

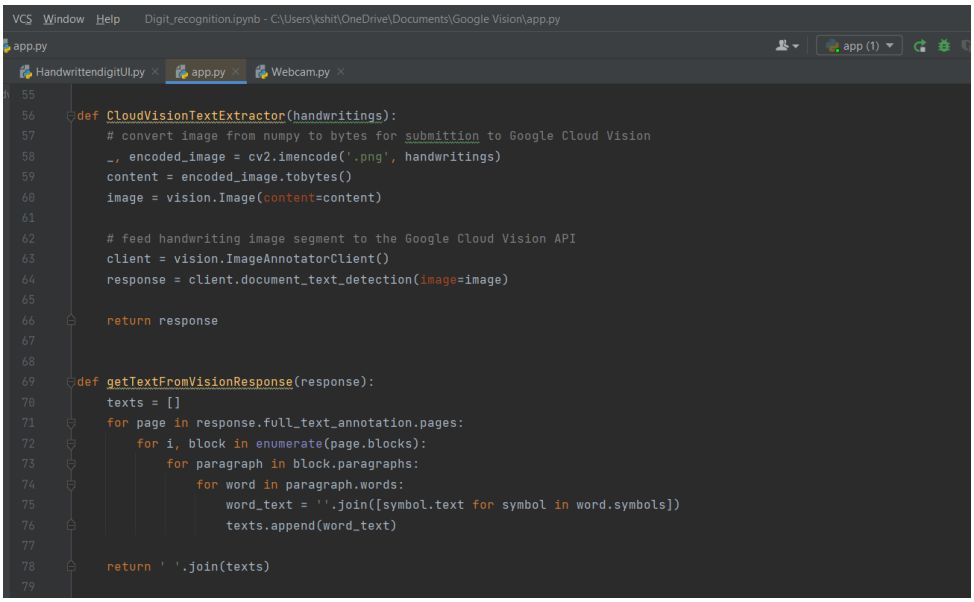
2.2 Character Recognition Using CNN Model:







2.3 Handwritten recognition using Tesseract or Google Vision:

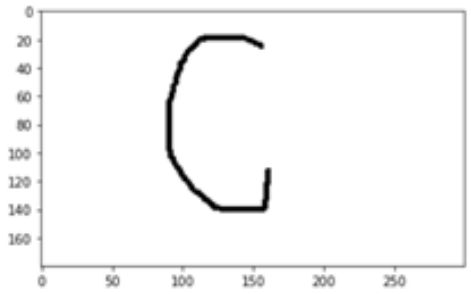


```python
def CloudVisionTextExtractor(handwritings):
    # convert image from numpy to bytes for submittion to Google Cloud Vision
    _, encoded_image = cv2.imencode('.png', handwritings)
    content = encoded_image.tobytes()
    image = vision.Image(content=content)

    # feed handwriting image segment to the Google Cloud Vision API
    client = vision.ImageAnnotatorClient()
    response = client.document_text_detection(image=image)

    return response


def getTextFromVisionResponse(response):
    texts = []
    for page in response.full_text_annotation.pages:
        for i, block in enumerate(page.blocks):
            for paragraph in block.paragraphs:
                for word in paragraph.words:
                    word_text = ''.join([symbol.text for symbol in word.symbols])
                    texts.append(word_text)

    return ' '.join(texts)
```

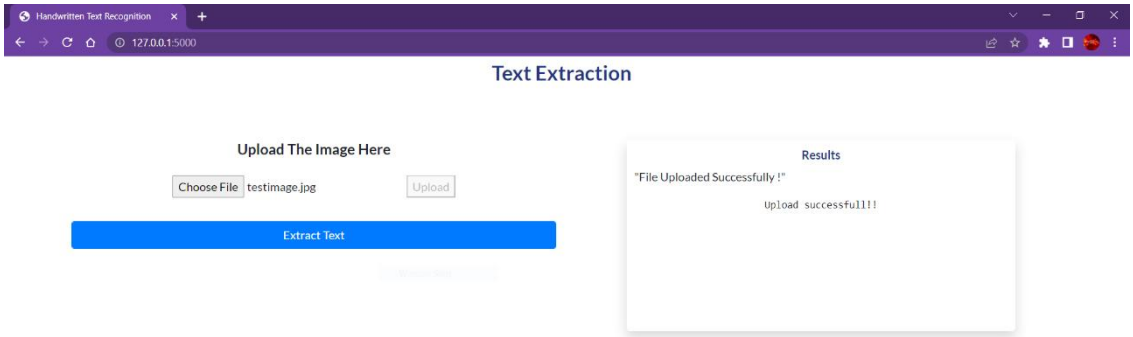
III. RESULTS

3.1 Digit Recognition Using CNN Model:

3.2 Character Recognition Using CNN Model:



3.3 Google Vision API:



IV. CONCLUSION

The application developed is user friendly, cost effective and applicable in the real time.The developed software has set all policies of the singles corresponding to each and every alphabet, its pronunciation methodology, the way it is used in grammar and dictionary. This can save time by allowing the user to listen background materials while performing other tasks. System can also be used to make information browsing for people who do not have the ability to read or write. This approach can be used in part as well. If we want only to text conversion then it is possible and if we want only text to speech conversion then it is also possible easily. People with poor vision or visual dyslexia or totally blindness can use this approach for reading the documents and books. People with speech loss or totally dumb person can utilize this approach to turn typed words into vocalization. Experiments have been performed to test the text reading system and good results have been achieved.

REFERENCES

[1]. Recognition of Offline Handwritten Characters using the Tesseract open-source OCR engine, Qi Li, Weihua An, Anmi Zhou, Lehui Ma, 2016.
[2]. Designing mobile application for retrieving book information using optical character recognition. Nana Ramadijanti, Achmad Basuki, AgrippinaG.J, W. 2016.
[3]. J. Pradeep, E. Srinivasan and S. Himavathi (2011). "Diagonal based feature extraction for handwritten alphabets recognitionsystem using neural network". International Journal ofComputer Science and Information Technology (IJCSIT), Volume 3, No 1, pp.27-38.
[4]. Ray Smith, Daria Antonova, and Dar-Shyang Lee. Adapting the tesser-act open source ocr engine for multilingual ocr. In Proceedings of the International Workshop on Multilingual OCR, page 1. ACM, 2009.
[5]. Mayank Singh1, Rahul, "Handwritten Digit Recognition using Machine Learning," IRJET, 2020.
[6]. Kartik Dutta, PraveenKrishnan, Minesh Mathewand C.V. Jawahar, "Towards Spotting and Recognition of Handwritten Words in Indic Scripts," IEEE, 2018.
[7]. Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, RobFergus, Yann LeCun, Integrated Recognition, Localization and De-tection using Convolutional Networks 2014.
[8]. Raymond W Smith. Hybrid page layout analysis via tab-stop detection.In 2009 10th International Conference on Document Analysis and Recog-nition, pages 241–245. IEEE,2009.