Report On

Offline Handwriting Character Recognition

Submitted in partial fulfillment of the requirements of the Course project in Semester IV of Second Year Computer Engineering

By
Siddhi Wade (Roll No. 73)
Kumud Zimal (Roll No. 65)
Supriya Virkar (Roll No. 58)

Supervisor
Ms. Sneha Mhatre

Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

(2023-24)

Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

CERTIFICATE

This is to certify that the project entitled "**Offline Handwriting Character Recognition**" is a bonafide work of " Siddhi Wade (Roll No. 73), Kumud Zimal (Roll No. 65), Supriya Virkar (Roll No. 58)" submitted to the University of Mumbai in partial fulfillment of the requirement for the Course project in semester IV of Second Year Computer Engineering.

Ms.Sneha Mhatre
Mentor

Dr Megha Trivedi
Head of Department

Dr. H.V. Vankudre
Principal

ABSTRACT

In this project we present an innovative method for offline handwritten character detection using deep neural networks. In today world it has become easier to train deep neural networks because of availability of huge amount of data and various Algorithmic innovations which are taking place. Now-a-days the amount of computational power needed to train a neural network has increased due to the availability of GPU's and other loud based services like Google Cloud platform and Amazon Web Services which provide resources to train a Neural network on the cloud. We have designed an image segmentation based Handwritten character recognition system. In our system we have made use of OpenCV for performing Image processing and have used Jupyter, Tensorflowlite and Keras for training the neural Network with the. We have developed this system using python programming language.

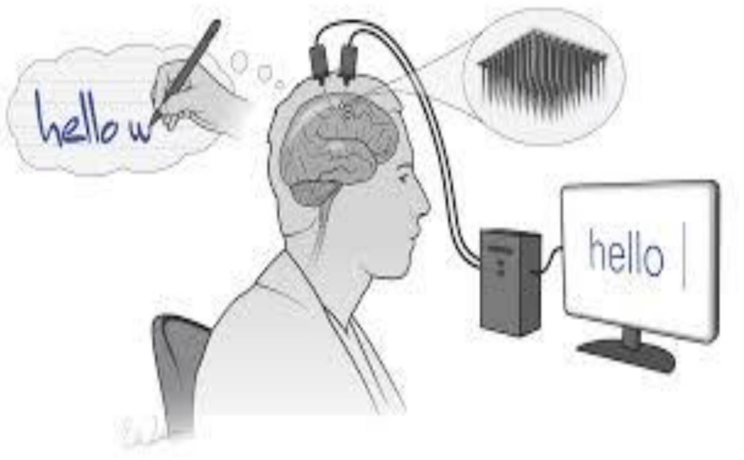
Keywords: Handwritten Character Recognition, Neural Network, OpenCV, Android, Jupyter, Tensorflowlite, Keras, Python, OCR, Image Processing, Feature Extraction

CONTENTS:

MODULE DESCRIPTION AND FLOWCHART:

In today's world AI(Artificial Intelligence) is the new Electricity. Advancements are taking place in the field of artificial intelligence and deep-learning every day. There are many fields in which deep-learning is being used. Handwriting Recognition is one of the active areas of research where deep neural networks are being t utilized. Recognizing handwriting is an easy task for humans but a daunting task for computers. Handwriting recognition systems are of two types: Online and Offline. In an online handwriting recognition system the handwriting of the user is recognized as the user is writing. The information like the order in which the user has made the strokes is also available. But in offline handwriting recognition system, the handwriting of user is available as an image. Handwriting recognition is a challenging task because different people have different styles of writing. and there are lot of characters like Capital letters Small letters, Digits and Special symbols. Thus a large dataset is required to train a near-accurate neural network model. To develop a good A system an accuracy of atleast 98% is required. However even the most modern and commercially available systems have not been able to achieve such a high accuracy. That's why we have chosen our Capstone Project topic as offline handwritten character system which will provide high accuracy rate.

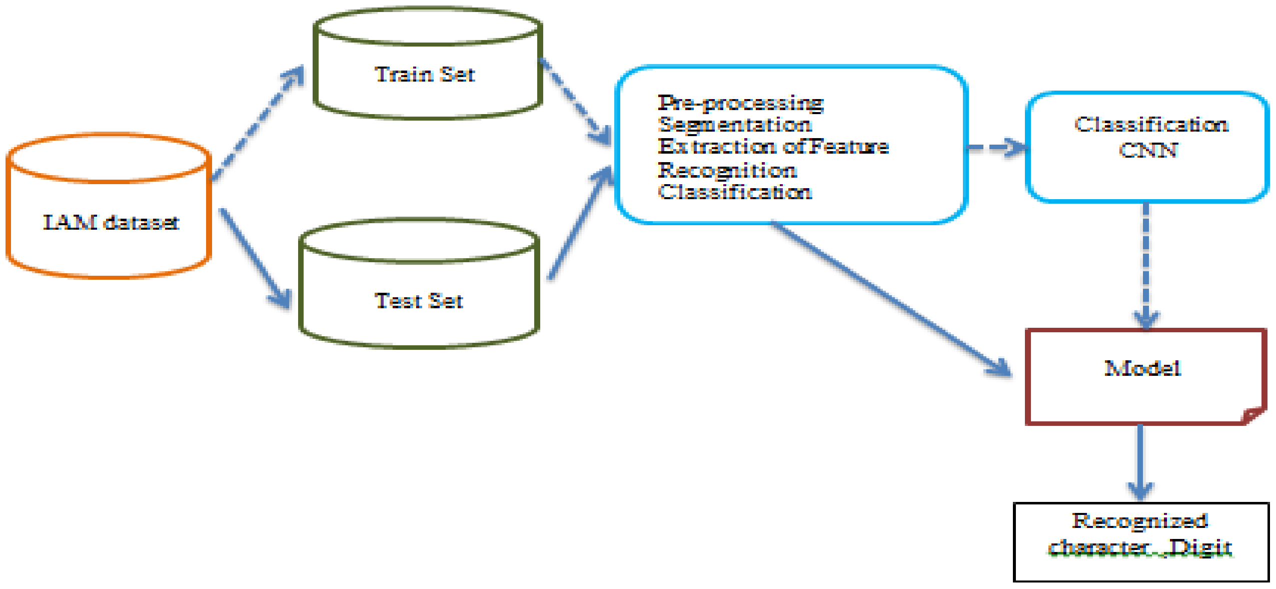
Handwritten character recognition is the process of converting handwritten text into digitalformat

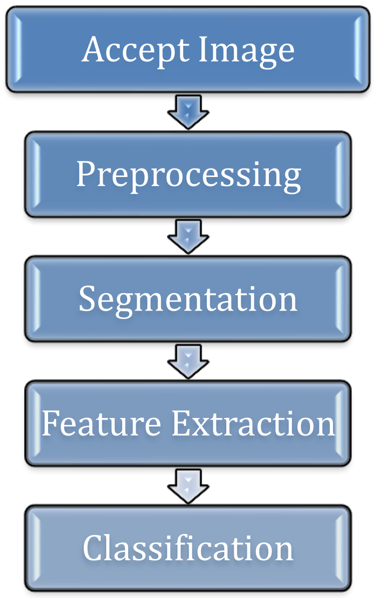


It has become increasingly important in today's world as more and more information is being recorded digitally. Offline handwritten character recognition system is an innovative technology that can recognize and interpret handwritten characters without the need for an internet connection. This technology has a wide range of applications, including document analysis, digitization, and data entry.
An offline handwritten character recognition system is a software application that uses machine learning algorithms to recognize handwritten characters from scanned images or photos. The system typically consists of several modules, including image preprocessing, feature extraction, and classification.
The image preprocessing module is used to enhance the quality of the input image, which may include steps such as noise removal, binarization, and segmentation. The feature extraction module is responsible for extracting relevant features from the preprocessed image, such as stroke width,

curvature, and orientation. These features are then used by the classification module, which identifies the characters based on the extracted features and a pre-trained model.
The pre-trained model is created using a dataset of handwritten characters, which is used to train the machine learning algorithms. The dataset may be created by collecting data from various sources, such as online handwriting samples, digitized documents, or specialized datasets for specific languages or scripts.
Once the system identifies the handwritten characters, it may perform additional post-processing steps, such as spell-checking, character segmentation, or handwriting recognition feedback to improve the accuracy of the recognition process.
Overall, an offline handwritten character recognition system can bring significant benefits to various industries by automating the process of reading and analyzing handwritten text, improving efficiency and accuracy, and saving time and resources.

Implementation:





SOFTWARE REQUIRED:

Python

Python is an interpreter, high-level, general-purpose programming language. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Rather than having all of its functionality built into its core, Python was designed to be highly extensible.

Jupyter notebook

Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. It is used for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

To use Jupyter Notebook, you need to install it on your computer or use a cloud-based service such as Google Colaboratory, Microsoft Azure Notebooks, or IBM Watson Studio. Once installed, you can create a new notebook and start working by creating code cells and adding text and visualizations using markdown language. You can also save and share your notebooks with others, allowing for collaboration and reproducibility.

Functions()

Following are the operations performed on the input in the system: Recognize ():

- pd.read_csv() = use to read path of csv file
- data.head() = use to display rows
- drop() = use to drop coloums
- split(X, y, test_size)= use to split x,y coordinate
- reshape() = use to custom image
- shuffle() = use to shuffle image.
- Subplots() = use to creates a new figure and returns a tuple containing a figure object and an array of subplots.
- flatten() = use to returns a copy of the array with all of its elements flattened into a single dimension.
- Threshold() = It is used to create a binary image from a grayscale or color image by setting all pixels above or below a certain threshold value to either 0 (black) or 1 (white).
- Imshow() = use to display image.
- Zeros() = used to create an array filled with zeros.

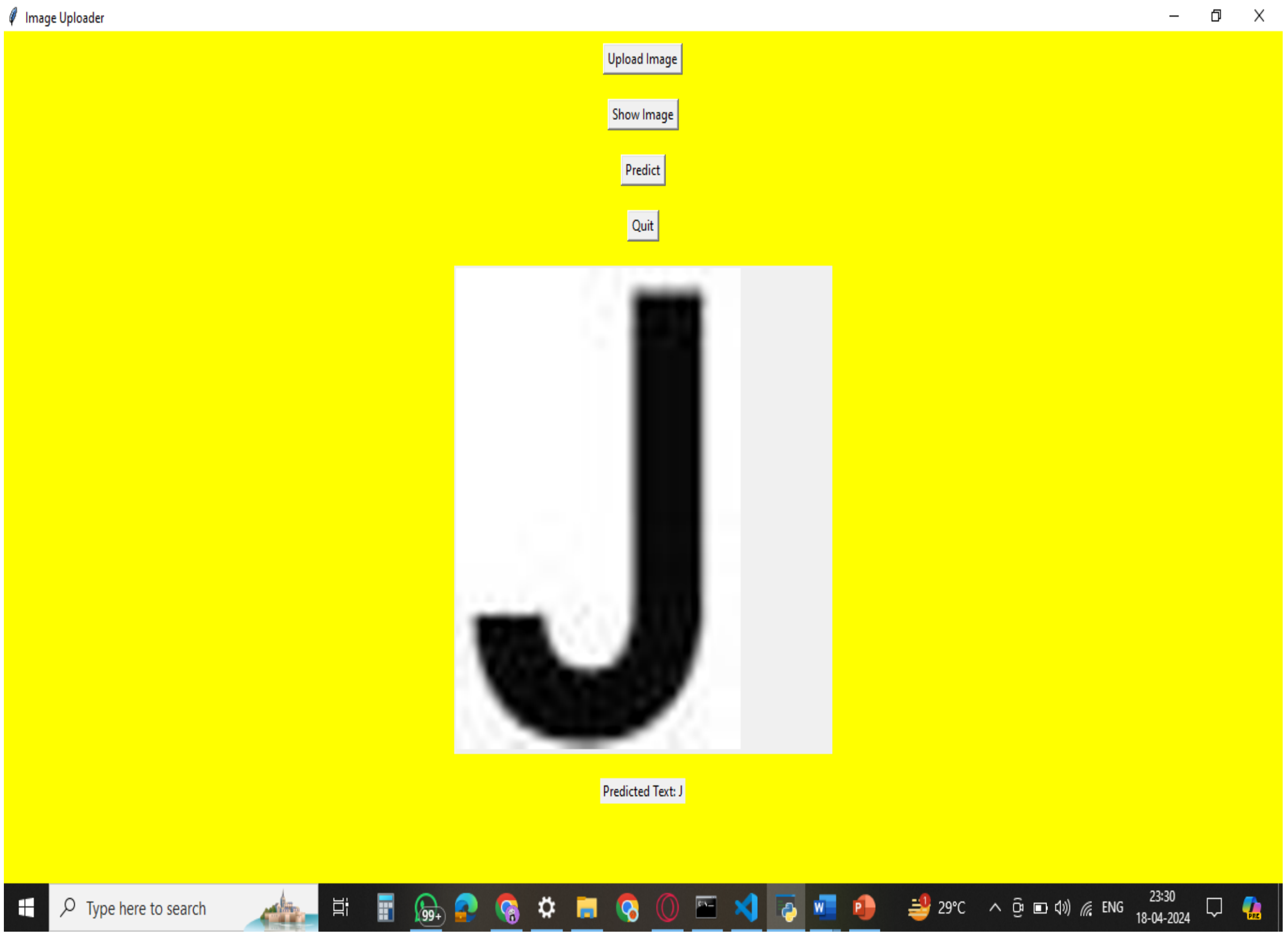
- values() = used to return a NumPy representation of a pandas object.
- Barh() = used to create a horizontal bar chart.
- Grid() = used to display grid lines on a plot.
- show() = used to display a plot on the screen.
- to_categorical() = used to convert a class vector (integers) to binary class matrix.
- Sequential() = used to create a linear stack of layers in a neural network.
- Conv2D() = used to add a 2D convolutional layer to a neural network.
- MaxPool2D() = used to add a 2D max pooling layer to a neural network.
- Flatten() = used to flatten the output of a convolutional layer or a max pooling layer into a one-dimensional vector.
- Fit() = used to train a neural network model.
- summary() = used to print a summary of a neural network model.
- Save() = used to save a trained neural network model to a file.

PROGRAM:

```
import tkinter as tk
from tkinter import filedialog
import cv2
from PIL import Image, ImageTk
import matplotlib.pyplot as plt

import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Flatten, Conv2D, MaxPool2D, Dropout
from keras.optimizers import SGD, Adam
from keras.callbacks import ReduceLROnPlateau, EarlyStopping

from keras.utils import to_categorical
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.utils import shuffle
from keras.models import load_model

model = load_model('model_hand.h5')

class ImageUploader:

    def __init__(self, master):
        self.master = master
        master.title("Image Uploader")

        self.image = None
        self.filename = None
        self.upload_button = tk.Button(master, text="Upload Image",
    command=self.upload_image)
        self.upload_button.pack(pady=10)

        self.show_button = tk.Button(master, text="Show Image",
    command=self.show_image, state=tk.DISABLED)
        self.show_button.pack(pady=10)

        self.predict_button = tk.Button(master, text="Predict", command=self.predict)
        self.predict_button.pack(pady=10)

        self.quit_button = tk.Button(master, text="Quit", command=master.quit)
        self.quit_button.pack(pady=10)


        self.canvas = tk.Canvas(master, width=400, height=400)
        self.canvas.pack(pady=10)



        self.canvas_image = None


        self.prediction_label = tk.Label(master, text="")
        self.prediction_label.pack(pady=10)

    def upload_image(self):
        self.filename = filedialog.askopenfilename(initialdir=".", title="Select Image File",
    filetypes=(("Image Files", ".jpg;.jpeg;*.png"),))
        if self.filename:
            self.image = cv2.imread(self.filename)
            self.show_button.config(state=tk.NORMAL)


    def show_image(self):
        pil_image = Image.fromarray(cv2.cvtColor(self.image, cv2.COLOR_BGR2RGB))
        canvas_width = 400
        canvas_height = 400
        img_width, img_height = pil_image.size
        ratio = min(canvas_width/img_width, canvas_height/img_height)
        new_width = int(img_width * ratio)
        new_height = int(img_height * ratio)
        pil_image = pil_image.resize((new_width, new_height))
        self.photo_image = ImageTk.PhotoImage(pil_image)
        self.canvas.create_image(0, 0, anchor=tk.NW, image=self.photo_image)

    def predict(self):
```

```
    model = load_model('model_hand.h5')
    word_dict                                                                                    =
{0:'A',1:'B',2:'C',3:'D',4:'E',5:'F',6:'G',7:'H',8:'I',9:'J',10:'K',11:'L',12:'M',13:'N',14:'O',15:'P'
,16:'Q',17:'R',18:'S',19:'T',20:'U',21:'V',22:'W',23:'X', 24:'Y',25:'Z'}

    if self.image is not None:
      img_copy = self.image.copy()
      img = cv2.cvtColor(self.image, cv2.COLOR_BGR2RGB)
      img = cv2.resize(img, (400,440))
      img_copy = cv2.GaussianBlur(img_copy, (7,7), 0)
      img_gray = cv2.cvtColor(img_copy, cv2.COLOR_BGR2GRAY)
      _, img_thresh = cv2.threshold(img_gray, 100, 255, cv2.THRESH_BINARY_INV)
      img_final = cv2.resize(img_thresh, (28,28))
      img_final = np.reshape(img_final, (1,28,28,1))
      img_pred = word_dict[np.argmax(model.predict(img_final))]
      self.prediction_label.config(text=f"Predicted Text: {img_pred}")

root = tk.Tk()
root.configure(bg='yellow')
uploader = ImageUploader(root)
root.mainloop()
```

RESULTS AND CONCLUSION:



Predicted Text: J

CONCLUSION:

- In conclusion, an offline handwritten character recognition system has the potential to bring significant benefits to various industries, including finance, healthcare, education, and more. By automating the process of reading and analyzing handwritten text, these systems can reduce errors, increase efficiency, and improve accuracy, thereby saving time and resources.
- The system architecture of an offline handwritten character recognition system typically involves various components, including data preprocessing, feature extraction, classification, and postprocessing. The accuracy and performance of the system depend on the quality of the dataset used for training and testing, the complexity of the model, and the optimization techniques employed.
- In the financial industry, an offline handwritten character recognition system can be particularly useful for tasks such as cheque processing, signature verification, customer identification, and document processing, among others.