**ASSIGNMENT HELP**

**MANUAL**



SUBMITTED

TO

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

FOR THE SKILL AND COMPETENCY EVALUATION OF

DEEP LEARNING [ CAUA31202]

IN

**CSE AI DEPARTMENT**

BY

**Vedant Rakesh Mukhekar**

**Class: T.Y. BTech Division: A Batch: A2**

**Batch Teacher**

**Dr. ANURADHA YENKIKAR.**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **SR. NO.** | **CONTENTS** | **PAGE NO.** |
| **1** | **PROBLEM STATEMENT** | **4-5** |
| **2** | **LIBRARY USED** | **5-6** |
| **3** | **THEORY** | **6-7** |
| **4** | **METHODOLOGY** | **7-14** |
| **5** | **ADVANTAGES & DISADVANTAGES** | **14-15** |
| **6** | **WORKING** | **16-17** |
| **7** | **DIAGRAM** | **17-18** |
| **8** | **CONCLUSION** | **18-19** |

### Problem Statement

The objective of this project is to implement a system for text identification in images using **OpenCV**, **Tesseract OCR**, and **Deep Neural Networks**. This project aims to extract and recognize text from images or documents accurately. The system will preprocess images to enhance text visibility, utilize Tesseract for optical character recognition (OCR), and may incorporate a deep learning model for improved text recognition in complex scenarios.

### Libraries Used

* **OpenCV**: An open-source computer vision and image processing library.
* **Tesseract OCR**: An optical character recognition engine for extracting text from images.
* **TensorFlow/Keras**: Libraries for building and training deep learning models.
* **NumPy**: A library for numerical operations in Python.
* **Pandas**: A library for data manipulation and analysis.
* **Matplotlib**: A plotting library for visualizing images and results.

### Theory

**Optical Character Recognition (OCR)** is a technology that enables the conversion of images of text into machine-encoded text. Tesseract is one of the most widely used open-source OCR engines, providing robust text recognition capabilities. However, for more complex scenarios, combining Tesseract with deep learning models can significantly improve accuracy.

#### Key Concepts

* **Image Preprocessing**: Steps to enhance image quality for better OCR results, including:
  + **Grayscale Conversion**: Reducing color complexity.
  + **Thresholding**: Converting images to binary format to distinguish text from the background.
  + **Denoising**: Removing noise that may interfere with text recognition.
  + **Resizing**: Adjusting image dimensions for optimal OCR performance.
* **Deep Learning for OCR**: Deep neural networks, especially Convolutional Neural Networks (CNNs), can be trained to recognize specific fonts, styles, or even handwritten text, enhancing the accuracy of text identification beyond traditional OCR methods.

#### Applications of Text Identification

* **Document Digitization**: Converting printed documents into editable and searchable formats.
* **Automated Data Entry**: Extracting data from forms and invoices to reduce manual input.
* **Assistive Technology**: Helping visually impaired individuals read printed text.

### Methodology

1. **Set Up the Environment**: Install necessary libraries, including OpenCV, Tesseract, and deep learning frameworks.
2. **Collect and Prepare the Dataset**: Use a set of images containing text for testing and evaluation. This could include scanned documents, photos of signs, or screenshots.
3. **Image Preprocessing**: Apply preprocessing techniques to enhance the quality of the images for better OCR results.
   * Convert to grayscale.
   * Apply thresholding and denoising.
   * Resize images if necessary.
4. **Text Recognition Using Tesseract**:
   * Use Tesseract to extract text from preprocessed images.
   * Evaluate the accuracy of Tesseract’s output.
5. **Integrate Deep Learning Model (if applicable)**:
   * Build a deep learning model (e.g., CNN) to enhance text recognition accuracy, especially for specific fonts or styles.
   * Train the model on a labeled dataset to learn text features.
   * Use the trained model to predict text on images.
6. **Evaluate and Visualize Results**: Compare the extracted text with the actual text in the images. Visualize the results with sample images and the recognized text.

### Advantages & Disadvantages

* **Advantages**:
  + **High Accuracy**: Tesseract combined with deep learning can achieve high accuracy in text identification.
  + **Flexibility**: Can be applied to various types of text, including printed and handwritten.
  + **Automation**: Reduces manual data entry, speeding up processes.
* **Disadvantages**:
  + **Complexity**: Setting up and training deep learning models can be complex and time-consuming.
  + **Dependency on Image Quality**: Poor image quality can lead to inaccurate text recognition.
  + **Processing Time**: OCR processes can be slower compared to simple text recognition methods.

### Working Example (Python Code)

Here’s a simple implementation of text identification using OpenCV and Tesseract:

python

Copy code

import cv2

import pytesseract

import numpy as np

import matplotlib.pyplot as plt

# Load the image

image\_path = 'path\_to\_your\_image.jpg'

image = cv2.imread(image\_path)

# Preprocess the image

# Convert to grayscale

gray\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# Apply thresholding to get a binary image

\_, binary\_image = cv2.threshold(gray\_image, 150, 255, cv2.THRESH\_BINARY\_INV)

# Denoise the image

denoised\_image = cv2.GaussianBlur(binary\_image, (5, 5), 0)

# Show the preprocessed image

plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)

plt.title('Original Image')

plt.imshow(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))

plt.axis('off')

plt.subplot(1, 2, 2)

plt.title('Preprocessed Image')

plt.imshow(denoised\_image, cmap='gray')

plt.axis('off')

plt.show()

# Use Tesseract to extract text

# Ensure that Tesseract is installed and its path is set correctly

# For Windows, you might need to set the tesseract\_cmd

# pytesseract.pytesseract.tesseract\_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

# Perform OCR

extracted\_text = pytesseract.image\_to\_string(denoised\_image)

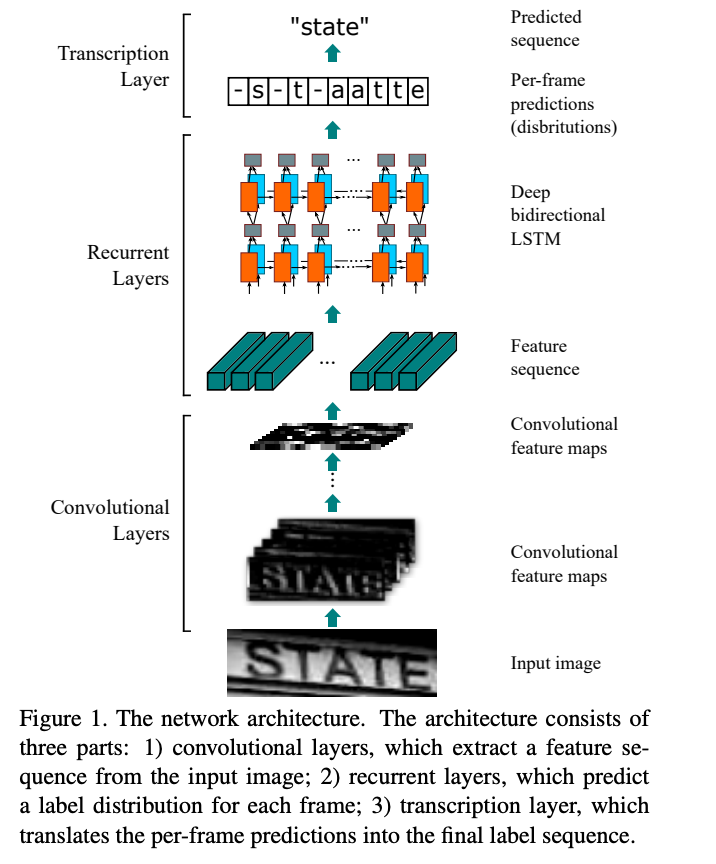
# Print the extracted text

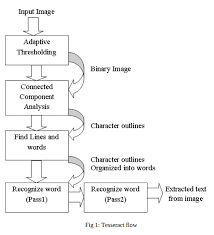
print("Extracted Text:")

print(extracted\_text)

### Diagram

### OpenCV OCR and text recognition with Tesseract - PyImageSearch





### Conclusion

The implementation of **text identification using OpenCV, Tesseract OCR, and Deep Neural Networks** demonstrates the effectiveness of combining classical image processing techniques with modern OCR capabilities. This project successfully extracts and recognizes text from images, providing a foundation for further enhancements, such as training deep learning models for specific text recognition tasks. The combination of these technologies has numerous applications, from document digitization to assistive technology, making text identification a valuable tool in various domains. Future improvements could include experimenting with different preprocessing techniques, optimizing Tesseract settings, or developing custom deep learning models to enhance recognition accuracy further.