AUTOMATED TEXT RECOGNITION USING OPEN CV

A DESIGN PROJECT REPORT

Submitted by

AJANEESHWAR.S

KAMALESHWAR.A

NAVEEN KUMAR.K.R

SIDDHARTHAN.R

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENGE AND DATA SCIENCE

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM -621112

JUNE 2024

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY
(AUTONOMOUS)
SAMAYAPURAM - 621112

BONAFIDE CERTIFICATE

Certified that this design project report titled **"AUTOMATED TEXT RECOGNITION USING OPEN CV"** is the bonafide work of **AJANEESHWAR.S (REG NO: 811721243005), KAMALESHWAR.A (REG NO:811721243023),NAVEENKUMAR.K.R (REG NO: 811721243038), SIDDHARTHAN.R. (REG NO: 811721243052)** who carried out the project work under my supervision.

SIGNATURE
Dr.T.Avudaiappan,M.E.,Ph.D.
HEAD OF THE DEPARTMENT
Associate Professor
Department of Artificial Intelligence
K.Ramakrishnan College of Technology(Autonomous)
Samayapuram – 621 112

SIGNATURE
Mrs. G. Nalina Keerthana,M.E.
SUPERVISOR
Assistant Professor
Department of Artificial Intelligence
K.Ramakrishnan College of Technology(Autonomous)
Samayapuram – 621 112

Submitted for the viva-voce examination held on ……

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We jointly declare that the project report on "**AUTOMATED TEXT RECOGNITION USING OPEN CV"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This design project report is submitted on the partial fulfilment of the requirement of the award of Degree of **BACHELOR OF TECHNOLOGY**.

SIGNATURE

_____

AJANEESHWAR.S

_____

KAMALESHWAR.A

_____

NAVEEN KUMAR.K.R

_____

SIDDHARTHAN.R

PLACE: SAMAYAPURAM

DATE:

ACKNOWLEDGEMENT

It is with great pride that we express our gratitude and in-debt to our institution **"K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)"** for providing us with the opportunity to do this project.

We are glad to credit honorable Chairman **Dr. K. RAMAKRISHNAN, B.E,** for having provided for the facilities during the course of our study in college.

We would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D,** for forwarding to our project and offering adequate duration in completing our project.

We would like to thank **Dr. N. VASUDEVAN, M.E., Ph.D**, Principal, who gave opportunity to frame the project the full satisfaction.

We whole heartily thanks to **Dr. T. AVUDAIAPPAN, M.E, Ph.D,** Head Of the Department, **ARTIFICAL INTELLIGENCE** for providing his encourage pursuing this project.

I express my deep and sincere gratitude to my project guide **Mrs. G. NALINA KEERTHANA,M.E,** ASSISTANT PROFESSOR, **ARTIFICAL INTELLIGENCE** for her incalculable suggestions, creativity, assistance and patience which motivated me to carry out the project successfully.

I render my sincere thanks to my Project Coordinator **Mrs. G. NALINA KEERTHANA,M.E,** and other staff members for providing valuable information during the course. I wish to express my special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

ABSTRACT

A computer vision system is developed that can accurately extract characters from visual and image data using the OpenCV library. The system is designed to process a wide range of visual inputs, including photographs, videos, scanned documents and extract text characters with high precision and efficiency. The extracted characters can be used for a variety of applications, including text recognition, language translation and document analysis. It presents a detailed analysis of the techniques and algorithms used to implement the character extraction system, including pre-processing methods, feature extraction methods and classification algorithms. A method for extracting characters from visuals and images is an open-source computer vision library. This method involves pre-processing the input image to improve the contrast and remove noise, followed by segmenting the image into individual characters using techniques such as thresholding and contour detection. The characters are then classified using machine learning techniques such as Support Vector Machines or Artificial Neural Networks. This method is evaluated on a dataset of images containing characters of various fonts and sizes, and the results demonstrate the effectiveness of the approach in accurately extracting and classifying the characters.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
CV	Computer Vision
HCR	Handwritten Character Recognition
KNN	K-Nearest Neighbour
MLP	Multi Layered Preceptron
SVM	Support Vector Machine
OCR	Optical Character Recognition

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

Extracting characters from visuals and images is a crucial task in image processing and computer vision. It involves the identification and extraction of individual characters from an image, which can be used for various applications such as optical character recognition (OCR), text-to-speech conversion and natural language processing. OpenCV is a powerful open-source library that provides a wide range of functions and algorithms for image processing and analysis, making it an ideal tool for character extraction from visuals and images. The goal is to develop a system that can accurately extract the characters from a visual or image containing text in ASCII characters such that it can be recognized by a computer. OpenCV provides several techniques for character extraction from visuals and images, including contour detection, edge detection and thresholding. Contour detection is a popular technique used for extracting characters from an image or a video frame. It involves identifying the edges of the characters and creating a closed curve around them. OpenCV provides several functions for contour detection, such as find Contours and draw Contours, which can be used to locate and isolate the characters.

Edge detection is another technique used for character extraction, which involves identifying the edges of the characters using various filters such as Sobel, Canny and Laplacian. OpenCV provides functions for edge detection, such as Canny and Laplacian, which can be used to locate and isolate the characters.

Thresholding is a technique used for separating the characters from the background based on a threshold value. OpenCV provides functions for thresholding, such as threshold and adaptive Threshold, which can be used to isolate the characters.

Optical Character Recognition is a technique used for recognizing the characters and extracting textual information from the image or video frame. OpenCV provides several OCR libraries, such as Tesseract and OCRopus, which can be used to recognize and extract textual information accurately.

Character extraction from visuals and images has numerous applications, including automated text recognition in images, license plate recognition, document analysis and more. Automated text recognition in images is used in various applications such as image captioning, image indexing and image search. License plate recognition is used for automated toll collection, parking management and law enforcement. Document analysis is used for automated text recognition, indexing and archiving. License plate recognition is used for automated toll collection, parking management and law enforcement. It involves extracting the license plate number from an image or a video frame and using it for various purposes such as toll collection, parking management and law enforcement.

Character extraction from visuals and images using OpenCV is a complex and challenging task that plays a crucial role in various applications such as automated text recognition, license plate recognition and document analysis. OpenCV provides a range of functions and algorithms for character extraction, including contour detection, edge detection and thresholding. While there are several challenges associated with character extraction, there is still a lot of scope for future research in this field, including the development of more accurate and robust algorithms, integration of machine learning techniques, exploration of new applications and development of user-friendly tools and software. Overall, character extraction from visuals and images using OpenCV has enormous potential for various applications and further research in this field can lead to significant advancements in computer vision and image processing.

1.2 OBJECTIVE

The project aims to achieve several objectives in the extraction of characters from visuals and images using OpenCV. The first objective is to accurately identify and extract individual characters from various types of visual inputs. This involves implementing techniques that can precisely locate and extract the characters from the images. To improve the accuracy of character extraction, the project focuses on optimizing the pre-processing stage. OpenCV's image processing techniques will be applied to enhance the quality of the input images. By adjusting contrast, reducing noise, and improving overall image quality, the system aims to enhance the accuracy of character extraction.

Another important objective is to refine the feature extraction process. OpenCV provides advanced feature extraction methods that can identify relevant characteristics of characters, such as edges, lines, and corners. By leveraging these methods, the project aims to ensure accurate recognition of characters.

Efficient segmentation is another key objective of the project. The system will develop algorithms and approaches that effectively divide the image into individual character regions. This segmentation step is crucial for isolating each character and enabling accurate extraction. It also explores integration with Optical Character Recognition (OCR) techniques. By leveraging OpenCV's capabilities, the system aims to interpret the extracted characters as text. This integration with OCR will enhance the functionality and usefulness of the extracted characters.

To assess the effectiveness of the system, the project includes evaluation and performance analysis. Various metrics, such as accuracy, efficiency, and robustness, will be used to measure the system's performance. This evaluation process will ensure that the system meets the desired objectives and performs accurately in character extraction from visuals and images.

CHAPTER 2

LITERATURE SURVEY

2.1 A FUZZY LOGIC BASED HANDWRITTEN NUMERAL RECOGNITION SYSTEM

Authors: Mahmood K Jasim, Anwar M Al-Saleh, Alaa Aljanaby

Year: 2013

Abstract

A novel method for recognizing handwritten numerals using fuzzy logic, which involves a unique approach to treating the input data. The system was trained on a dataset of 100 patterns of handwritten numerals ranging from 0 to 9, obtained from 10 different subjects. Each pattern was scanned and converted into a 30x20 binary image, which was then fed into the system. The recognition process was carried out offline, resulting in an impressive. The method offers a promising approach for accurately recognizing handwritten numerals, which could have important implications for a wide range of applications, including document processing, character recognition and computer vision.

Introduction

Pattern recognition system is regarded as a system, whose input is the information of the pattern to be recognized and output is a class to which the entered pattern belongs. One of the important fields in pattern recognition is character recognition.The main objective of character recognition is the conversion of a graphical document into a textual one. Many systems have been proposed to recognize pattern. Some of these systems have been used the fuzzy logic. Most of the character recognition systems require pre-processing operations on the pattern. Pre-processing is an important step in pattern recognition systems in which fundamental features of pattern are extracted and/or enhanced, to classify and recognize unknown pattern .

Many tedious tasks can be made more efficient by automating the process of reading handwritten numerals. In such system an optical scanner converts each handwritten numeral to a digital image and computer software classifies the image as one of the digits zero through nine. By reducing the need for human interaction, numeral-recognition systems can speed up jobs such as reading income tax returns, sorting inventory and routing mail. Several steps are necessary to achieve this. A recognition system must first capture digital image of handwritten numerals. Before attempting to classify the numerals, some pre- processing image might be necessary. An algorithm must then classify each handwritten numeral as one of the ten decimal digits.

Although a qualitative description of this process is straightforward, it cannot be easily reduced to a few simple mathematical rules. The difficulty results from the natural variations in human handwritten. A useful recognition system must be robust to alterations in size, shape, orientation, thickness, etc. Closed- form mathematical models tend to be inadequate for such a task because of the many possible representations of the same image. It can present an off-line system for the recognition of handwritten numerals with pre-processing steps uses fuzzy logic and called Handwritten Numeral Recognition Using Fuzzy Logic.

Methodology

Image acquisition is a crucial step in the process of recognizing objects or characters in an image. This process involves capturing digital images of the target objects or characters using imaging devices such as cameras or scanners. The quality of the acquired images plays a vital role in the accuracy and reliability of the recognition system. Therefore, the image acquisition process should be optimized to ensure that the images captured are clear, consistent and free of distortions or noise.

There are several factors that affect the quality of acquired images, including the resolution, color depth, lighting conditions and focus. The

image determines the level of detail that can be captured and it is typically measured in pixels. Higher resolutions lead to more detailed images, but this also require more storage space and computational resources to process. The color depth refers to the number of colors that can be represented in an image and it affects the visual quality of the image. Lighting conditions also play a significant role in image acquisition, that can affect the contrast and brightness of the image. Therefore, it is essential to ensure that the lighting is consistent and optimal for the object being captured. Finally, the focus of the imaging device should be adjusted to ensure that the image is sharp and clear.

Once the images are acquired, they are typically pre-processed to enhance their quality and remove any noise or distortions. Pre-processing may involve operations such as filtering, normalization or segmentation. Filtering is used to remove noise from the image, while normalization is used to adjust the brightness and contrast of the image to a consistent level. Segmentation involves dividing the image into smaller regions or segments, which can then be analysed individually.

The acquired and pre-processed images are fed into the recognition system, which utilizes algorithms and machine learning techniques to analyse and classify the images. Recognition systems can be trained using various techniques, such as supervised learning, unsupervised learning or reinforcement learning. Supervised learning involves training the system on a dataset of labelled images, while unsupervised learning involves discovering patterns and relationships in unlabelled data. Reinforcement learning involves training the system to optimize a specific reward function.

Effective image acquisition is crucial for achieving high recognition accuracy and for ensuring the overall success of the recognition system. Poor image quality can result in misclassification or inaccurate recognition, which can have significant consequences in applications such as document processing or security systems. Therefore, image acquisition should be optimized to

ensure that the images captured are of high quality and the recognition system should be designed to handle variations in the image quality and other factors that may affect the recognition accuracy. With advances in imaging technologies and machine learning techniques, recognition systems are becoming increasingly reliable and accurate, opening up new possibilities for applications in various fields.

The handwritten character recognition involves the development of a method that can generate descriptions of the handwritten objects in a short period of time. Due to its low computational requirement, fuzzy logic is probably efficient method available for character recognition. Fuzzy systems are robust; even if some rules are removed from the rule map, the system could still work properly. The fuzzy systems are also robust toward changing conditions in the environment and provide simple and effective methods for handwritten character recognition because fuzzy features can represent the characters.

Merits

- **Robustness** Fuzzy logic-based handwritten numeral recognition systems are generally more robust to variations in handwriting styles and noise. This is because fuzzy logic allows for a more flexible and tolerant approach to recognizing patterns, which can be particularly useful in cases where the input data is noisy or poorly defined.
- **Scalability** Fuzzy logic-based recognition systems can be easily scaled to handle large volumes of input data. This is because the system can be designed to recognize a wide range of patterns, which makes it suitable for applications such as document analysis or automated data entry.
- **Interpretability** Fuzzy logic-based recognition systems are often more interpretable than other recognition systems. This is because fuzzy logic allows for a more natural and intuitive approach to modelling patterns, which can be particularly useful in applications where it is important to understand how the system arrived at its decisions.

- **Ease of Development** Fuzzy logic-based recognition systems are generally easier to develop than other recognition systems. This is because fuzzy logic provides a more straightforward and intuitive approach to modelling patterns, which can be particularly useful for researchers and developers who are new to the field.

Demerits

- **Lack of Precision** Fuzzy logic-based recognition systems can be less precise than other recognition systems. This is because fuzzy logic allows for a more flexible and tolerant approach to recognizing patterns, which can result in false positives or false negatives.
- **Complexity** Fuzzy logic-based recognition systems can be more complex to design and implement than other recognition systems. This is because fuzzy logic requires a more sophisticated understanding of pattern recognition and machine learning, which can be challenging for researchers and developers who are new to the field.
- **Computational Cost** Fuzzy logic-based recognition systems can be more computationally expensive than other recognition systems. This is because fuzzy logic requires a more sophisticated and computationally intensive approach to modelling patterns, which can result in slower performance and higher computational costs.
- **Lack of Standardization** Fuzzy logic-based recognition systems are often less standardized than other recognition systems. This is because fuzzy logic allows for a more flexible and adaptable approach to modelling patterns, which can result in a lack of standardization across different systems and applications.

2.2 HANDWRITTEN CHARACTER RECOGNITION USING NEURAL NETWORK AND TENSOR FLOW

Authors: Megha Agarwal, Shalika, Vinam Tomar, Priyanka Gupta

Year: 2019

Abstract

Handwritten Character Recognition is an important problem that has been extensively studied over the years. Despite the efforts, obtaining accuracy in HCR is a challenging task. One of the major reasons for this is the complexity of handwritten characters, which makes it difficult for algorithms to accurately recognize them. A new approach to offline HCR that utilizes Convolutional Neural Networks and TensorFlow. CNNs have proven to be a powerful tool for image recognition tasks and have shown promising results in HCR as well. To assign probabilities to handwritten characters, employ Soft Max Regression, a widely used method in image recognition tasks. HCR software system with high accuracy rates while minimizing time and space complexity and optimizing overall performance. Achieving this goal would greatly benefit various industries that rely on HCR for data processing, such as banking, insurance and logistics. Overall aims is to push the boundaries of HCR and contribute to the advancement of this field. By leveraging the power of CNNs and Soft Max Regression, to make significant progress towards achieving accuracy rates in HCR and ultimately make HCR more accurate and efficient for real-world applications.

Introduction

The human ability to understand handwritten characters or typed documents can also be imparted to machines through the use of machine learning and artificial intelligence, which is the field of study known as Optical Character Recognition. spans across various fields such as pattern recognition, image vision and AI and involves the conversion of

electronic and image text into digital characters that can be easily read by machines. By reducing the time taken to enter data and the storage space required for documents, OCR can enhance the efficiency of document processing in various domains, such as banking, legal and administrative sectors, without the need for human intervention.

OCR can be categorized into two types based on the type of text and document acquisition: Handwritten Character Recognition and Printed Character Recognition. HCR involves intelligent recognition of handwritten text, which is challenging due to the varying handwriting styles of different individuals and even the same individual at different times. To achieve high recognition accuracy, OCR systems must employ advanced techniques such as neural networks and deep learning algorithms.

OCR is further characterized into two forms: Offline and Online recognition systems based on the method of document acquisition. Offline systems recognize pre-written documents acquired through various input methods, whereas online systems recognize writing as soon as it is produced. Online systems typically use an electric pen to capture pen movement and record the input on a digitizer.

While OCR has several advantages, such as the ability to process sensitive documents faster and without human intervention, it also has certain limitations. OCR systems can encounter issues with low-quality input images, such as blurred or noisy images, which can result in inaccurate recognition. OCR systems also require significant computing resources, including high-speed processors and large amounts of storage, which can increase the system's overall cost. Therefore, careful consideration of OCR's advantages and limitations is necessary to determine its suitability for different applications.

Methodology

Pre-processing is an essential step in HCR that involves cleaning and enhancing the input image before it is fed into the neural network. This step is critical as it directly affects the accuracy of the recognition system. One common pre-processing technique is normalization, which involves scaling the input image to a fixed size. This technique helps to reduce the effects of variations in handwriting styles and makes it easier for the neural network to recognize the characters.

Another pre-processing technique is binarization, which converts the input image into a binary image by setting a threshold value. This technique helps to separate the handwriting from the background and makes it easier for the neural network to recognize the characters. A popular binarization technique is Otsu's method, which automatically calculates the threshold value based on the image histogram.

In addition to normalization and binarization, other pre-processing techniques include noise removal, skew correction and thinning. Noise removal involves removing any unwanted artifacts in the input image, such as speckles or smudges. Skew correction involves straightening the text lines in the input image, which helps to improve the accuracy of character recognition. Thinning involves reducing the width of the characters in the input image, which helps to improve their separability.

Tensor Flow provides several built-in functions and tools for pre-processing techniques, such as image resizing, rotation and flipping. These functions can be used to normalize the input image and correct any distortions or deformations that may affect the accuracy of character recognition. Tensor Flow also provides a data augmentation tool, which can generate new training data by applying random transformations to the input images, such as rotation, scaling and flipping. This tool helps to improve the generalization ability of the neural network and reduce overfitting.

It is a crucial step in HCR that can significantly affect the accuracy of character recognition. Neural networks combined with Tensor Flow provide an effective approach for pre-processing techniques in HCR. By leveraging the power of neural networks and Tensor Flow, helps us to develop more accurate and efficient HCR systems that can be applied to various real-world applications.

It deals with the methods necessary to construct the raw input images. The segmentation phase deals with segmenting the raw image into single characters and then arranging it into m*n pixels to deal with it mathematically. The feature extraction method is perhaps of main significant issue in achieving high recognition percentage for which various techniques like Deformable templates, Gradient feature, Contour profiles, Fourier descriptors, Gabor features, Graph description, Geometric moment invariants, Template matching, Unitary Image transforms, Projection Histograms, Zoning, Zernike Moments and Spline curve approximation are used.

The input image is provided to the recognition system. The input can be either in an image format such as JPEG, BMT, etc. or scanned image, digital camera or any other suitable digital input device or one can draw on the canvas provided on the user interface.

Merits

- **High accuracy** The primary advantage of HCR using Neural Network and Tensor Flow is its ability to achieve high accuracy. The use of advanced algorithms and deep learning techniques has significantly improved the accuracy of HCR.
- **Flexibility** Another advantage of HCR using Neural Network and Tensor Flow is its flexibility. It can recognize different styles of handwriting, including cursive writing, which makes it useful for various applications.

- **Faster processing** HCR using Neural Network and Tensor Flow has also shown remarkable speed in processing handwritten characters. This is particularly useful in applications where real-time recognition is required.
- **Reduced human intervention** HCR using Neural Network and Tensor Flow eliminates the need for human intervention in the recognition process, making it more efficient and reliable.
- **Easy to implement** The use of open-source platforms like Tensor Flow makes it easier to implement HCR systems. This, in turn, reduces the development time and cost.

Demerits

- **Training time** One of the biggest drawbacks of HCR using Neural Network and Tensor Flow is the time required to train the system. The system needs to be trained on a large dataset of handwritten characters, which can be time-consuming.
- **Large dataset requirement** The accuracy of the HCR system is directly proportional to the size of the dataset used for training. A larger dataset leads to higher accuracy, but collecting and organizing such a large dataset can be challenging.
- **Sensitivity to noise** HCR using Neural Network and Tensor Flow can be sensitive to noise in the input data. Even minor variations in handwriting can lead to incorrect recognition.
- **Cost** The cost of implementing HCR using Neural Network and Tensor Flow can be high, particularly for small-scale applications.
- **Maintenance** The HCR system requires regular maintenance to ensure high accuracy. Any changes in the input data or the system's configuration can impact the accuracy of the system, requiring constant monitoring and adjustments.

2.3 A COMPARATIVE STUDY ON HANDWRITING DIGIT RECOGNITION BY USING CONVOLUTIONAL NEURAL NETWORK AND PYTHON

Authors: Tarun Patel1, Shivansh Tiwari, Vaibhav Dubey, Vaibhav Singh, Dr. Harvendra Kumar

Year: 2021

Abstract

Handwritten Digit Recognition has become increasingly popular in the field of pattern recognition and machine learning. Both Optical Character Recognition and Handwritten Digit Recognition (HDR) have specific domains in which can applied for Digit Recognition in an HDR System. Numerous strategies have been proposed and many research papers outline the ways to transfer textual content from a paper document to a machine-readable format. Digit Recognition systems could potentially play a crucial role in creating a paperless world in the future by converting and processing the remaining paper documents.

Deep learning has revolutionized the field of machine learning by making it more artificially intelligent. Deep learning is widely used in various industries, such as surveillance, health, medicine, sports, robots and drones, due to its extensive range of applications. Convolutional Neural Network is the core of remarkable advancements in deep learning, combining ANN and cutting-edge deep learning algorithms. Some of the applications of CNN include pattern recognition, phrase classification, audio recognition, face recognition, text classification, document analysis, scene recognition and HDR. The purpose of this research is to compare the accuracies of CNN in classifying handwritten digits using varying numbers of hidden layers and epochs. The experiment evaluates the CNN's performance.

Introduction

Handwritten Digit Recognition is fundamental but most challenging in the field of Digit Recognition with a good number of useful applications. It has been an intense field of study from the ancient era of computer science. It is the natural way of the interactions between computers and humans. More precisely Digit recognition is a process of detecting and recognizing the digit from the given image and converting it into the ASCII or another equivalent machine editable form. The technique from which the computer system can recognize digits and other symbols which is written by a human hand in its natural handwriting is called as HDR. Handwritten recognition is further classified into offline handwritten recognition and online handwritten recognition. If the writing is scanned and then understood by the computer, that is called offline handwriting recognition. In case, if the handwriting is recognized while it was written through a touchpad by using the stylus pen and then it is called online handwritten recognition. From a classifier perspective, digit recognition systems are further divided into two types i.e., segmentation free (global) and segmentation based (analytic).

The divided part is also known as a holistic approach for recognizing the digit without dividing it into subgroups or digits. Each word is appeared as a set of global features. e.g., ascender, loops, etc. Whereas the segmentation based on approach of each of the word is divided into other subgroups are uniform or non-uniform or then subunits are considered independent. The Handwritten Digit processing system is the domain and the application-specific like it's not possible for designing a generic system to process all of the types of handwritten scripts and languages. There is a lot work has been done on the European languages and Arabic (Urdu) language. The domestic languages which are Hindi, Punjabi, Bangla, Tamil, Gujarati, etc. are very less or looked into them due to the limited usage.

Methodology

Handwriting Digit Recognition is a challenging task that involves identifying handwritten digits from scanned images. This task has a wide range of applications, from digit recognition in postal services to recognizing handwritten digits in online transactions. In recent years, various techniques have been developed to achieve accurate and efficient Handwriting Digit Recognition. Some of the techniques are Fuzzy Membership Function, Hidden Markov Models, Supervised Learning Method using Convolutional Neural Network and Python.

Fuzzy Membership Function is a mathematical function that assigns a degree of membership to each pixel of the image, indicating the likelihood that the pixel belongs to a particular numeral. The degree of membership can be calculated using a set of rules that are based on the characteristics of the numeral. A pixel that has a high degree of membership for numeral "0" is likely to be circular and located near the center of the image. Fuzzy Membership Function is an effective method for handling noisy and ambiguous images.

Hidden Markov Models is another method used for Handwriting Digit Recognition. HMM models the sequence of strokes used to draw a digit. HMMs can capture the temporal information in the sequence of strokes, making them suitable for recognition tasks. HMMs can also handle variations in the size and shape of the digit, making them robust to different writing styles.

Supervised learning methods, such as Convolutional Neural Networks, have also been employed for Handwriting Digit Recognition. CNN is a type of deep learning network that uses convolutional layers to extract features from the input image. The extracted features are then fed into a fully connected layer to perform the classification task.

Python is a popular programming language used in the implementation of Handwriting Digit Recognition algorithms. Python provides various libraries and frameworks, such as TensorFlow, Keras and PyTorch, that simplify the implementation of machine learning models. These libraries provide tools for data preparation, model training and evaluation.

These techniques have advantages and disadvantages and the choice of the technique depends on the specific requirements of the task. However, CNN has emerged as a promising method for Handwriting Digit Recognition due to its ability to extract relevant features from the input image and achieve high accuracy. Python, with its rich set of libraries and frameworks, can be used to implement these methods effectively.

The altered quadratic classifier-based technique for offline handwritten numeral recognition in six of the popular Indian scripts. Handwritten English characters have been recognized using a multilayer perceptron. The Boundary tracing and associated Fourier Descriptors were used to retrieve the features. The character is set on by examining the shape and analysing its distinguishing characteristics. To obtain high performance in the backpropagation network, an analysis was also conducted to identify the no. of the hidden layer's nodes. Handwritten English alphabets or characters have been claimed to have a recognizing accuracy with minimal training time.

Merits

- **High Accuracy** CNN has achieved state-of-the-art performance in many image recognition tasks, including HDR. It can extract relevant features from the input image and use them to perform the classification task accurately.

- **Robustness** CNN is robust to variations in the size and shape of the digit, making it suitable for different writing styles. It also handle noisy and ambiguous images, which can occur in real-world scenarios.
- **Scalability** CNN can be trained on large datasets, making it scalable to handle a wide range of HDR tasks. It can be used for other image recognition tasks, such as object detection and scene classification.
- **Ease of Implementation** Python provides various libraries and frameworks, such as TensorFlow, Keras and PyTorch, that simplify the implementation of CNN for HDR. These libraries provide tools for data preparation, model training and evaluation.

Demerits

- **Computational Complexity** CNN requires significant computational resources, especially during the training phase. This needs to process a large amount of data and perform multiple convolutions and pooling operations, which can be time-consuming.
- **Data Requirement** CNN requires a large amount of labelled data to achieve high accuracy. This can be a challenge in cases where the data is limited or expensive to obtain.
- **Overfitting** CNN is prone to overfitting, which occurs when the model performs well on the training data but poorly on the test data. Overfitting can occur when the model is too complex and the training data is insufficient or noisy.
- **Black-Box Model** CNN is a black-box model, meaning that it is difficult to interpret the results of the model. It can be challenging to understand how the model makes its predictions, which can be a concern
  in some application.

2.4 HANDWRITTEN CHARACTER RECOGNITION TO OBTAIN EDITABLE TEXT

Authors: Vaibhav. V. Mainkar, Mr. Ajinkya B. Upade, Jyoti A. Katkar, Poonam R. Pednekar

Year: 2020

Abstract

Developing an android application for character recognition and text extraction from an image is a crucial area of research in the present era. With the rising trend of storing information from handwritten documents for future use, image capturing of the handwritten document and saving it in an image format is an easy way to store the information. To transform handwritten data into electronic format the method of Optical Character Recognition is used which involves various steps like pre-processing, segmentation, feature extraction and post-processing. OCR has been used by many researchers for recognizing characters from different styles of handwriting. This system utilizes an android phone to capture the image of the document and perform further steps by OCR to recognize the characters and extract text. However, the main challenge in developing an OCR system is to recognize the characters from different styles of handwriting. Hence, a system is designed that recognizes handwritten data to obtain editable text with good accuracy rate. The output of this system depends on the data written by the writer. Our system offers an easy way to edit and share the recognized data and it can prove to be a helpful tool for people dealing with handwritten documents.

Introduction

In today's world, there is a growing demand to create a paperless environment. However, recognizing handwritten text is not an easy task for computer systems, despite being a simple task for humans. Although several researchers have worked on this field, achieving 100% accuracy is still a challenge. Humans can differentiate the handwritten characters of different individuals, but computers cannot do this as

easily. To tackle this issue, 'Optical Character Recognition' has been introduced as a solution.

The main objective of this project is to leverage OCR's capabilities through an android application. As the demand for mobile applications is growing in the software industry, the android app provides users with the ability to recognize text from saved images from the gallery or captured through the camera. Users can easily edit and store the written data in a text file format. The app uses the camera of an Android mobile device to take input, which is a binary image scanned by the camera. The OCR engine processes the image data and converts it into text.

The technology behind OCR involves scanning the document or image using a scanner. Once the image is scanned, OCR software converts it into a black and white version. Then, the image is analysed by identifying the dark and light areas. Dark areas are identified as characters, while light areas are identified as background and the dark area is considered for further processing. Despite the potential for accuracy, achieving it is a challenge due to the variety of handwriting styles, which makes it difficult for the OCR system to recognize the characters accurately.

Methodology

Optical Character Recognition is a technique used to convert scanned or printed image documents into editable text documents. OCR can be used for recognizing both printed and handwritten text. The methodology used in OCR for handwritten character recognition involves several steps.

The first step is image acquisition, where a digital image of the handwritten text is obtained. This image is then pre-processed to remove noise, which can include artifacts such as scratches, smudges and uneven illumination. The pre-processing step also involves normalizing the image, which can involve scaling the image to a standard size and aligning it properly.Next, the image is segmented to separate the

characters from each other.

Once the characters have been segmented, features are extracted from each character. This can involve computing features such as stroke width, slant angle and curvature. Feature extraction is an essential step in the OCR process, as it provides the necessary information to distinguish one character from another.

Finally, the OCR system applies a recognition algorithm to the extracted features to recognize the characters. There are various recognition algorithms used in OCR systems, such as template matching, neural networks and hidden Markov models.

After the characters have been recognized, post-processing is performed to correct any errors that may have occurred during the recognition process. This can involve using techniques such as spell-checking and grammar checking to improve the accuracy of the recognized text.

Overall, the OCR methodology for handwritten character recognition is a complex process that involves several steps, including image acquisition, preprocessing, segmentation, feature extraction, character recognition and postprocessing.

Merits

• **Increased Efficiency** OCR helps to automate the process of manual data entry by recognizing handwritten or printed text and converting it into digital format. This leads to increased efficiency and productivity in various fields such as banking, legal and administrative work.

• **Time-Saving** OCR helps to save time by automating the process of manual data entry. It can recognize text much faster than humans and can process a large amount of data in a short period of time.

• **Accuracy** OCR has a high accuracy rate and can recognize text with great precision. This leads to improved data quality and reduced errors in data entry.

• **Searchability** Once text is recognized by OCR, it can be easily searched and retrieved using digital search tools. This makes it easier to manage and access large amounts of data.

Demerits

• **Limited Recognition Ability** OCR has limited recognition ability when it comes to recognizing handwritten text. It may struggle to recognize text that is poorly written or has unusual writing styles.

• **Language Limitations** OCR may not recognize text in languages that are not supported by the software. This can lead to difficulties in recognizing text from documents that are written in multiple languages.

• **Complexity** OCR can be complex to implement and require specialized hardware and software. This can be expensive and time-consuming to set up and maintain.

• **Training and customization requirements** This process involves collecting a large dataset of handwritten samples, manually labelling them, and training the recognition model.

2.5 HANDWRITTEN CHARACTER RECOGNITION USING NEURAL NETWORK

Authors: Ayush Purohit, Shardul Singh Chauhan

Abstract

Handwritten Character Recognition system using Optical Character Recognition technology for recognizing handwritten characters and converting them into editable text. HCR has gained significant attention in the field of pattern recognition and machine learning due to its wide application in various fields. The objective of this project is to conduct a detailed review of existing techniques and propose an optimized solution for HCR. The system will allow for the digitization and processing of existing paper documents, serving as a key factor in creating a paperless environment. The project will utilize an android app to provide users with easy access to recognize the text from saved images or captured images through the camera. The proposed system will contribute towards achieving accurate and efficient handwritten character recognition, thus eliminating the need for manual transcription and enhancing productivity.

Introduction

Character recognition is a complex yet essential aspect of pattern recognition that finds its applications in various domains. Since the advent of computer science, it has been a subject of intense research, driven by the need to establish a natural way of communication between humans and machines. In simple terms, character recognition is the process of identifying and recognizing characters from an input image and converting them into machine-readable ASCII or other equivalent formats. Handwriting recognition, on the other hand, refers to the technique by which a computer system can recognize handwritten characters and other symbols.

Handwriting recognition is broadly classified into offline and online handwriting recognition. Offline handwriting recognition involves scanning the

handwriting and then recognizing it while online handwriting recognition is done in real-time while writing on a touchpad using a stylus pen. Character recognition systems are further classified into two main categories based on the classifier perspective, segmentation-free (global) and segmentation-based (analytic). In the segmentation-free approach, the character is recognized without segmenting it into subunits, while in the segmentation-based approach each word is segmented into subunits that are then recognized independently.

The important thing to note that handwritten character processing systems are domain and application-specific and therefore, designing a generic system that can process all kinds of handwritten scripts and languages is a challenge. While there has been significant progress in character recognition of European languages and Arabic (Urdu) language, domestic languages such as Hindi, Punjabi, Bangla, Tamil, Gujarati, etc. are still underexplored due to limited usage. Overall, character recognition remains an active area of research with many exciting opportunities for future development.

Methodology

Handwritten Character Recognition has been an active area of research in the field of pattern recognition and machine learning for many years. Various techniques have been proposed to develop robust and efficient HCR systems. Among these techniques, Hidden Markov Model, Artificial Neural Network and Multilayer Perceptron have shown promising results in recognizing handwritten characters. The methodology for HCR using HMM, ANN and MLP involves several stages. The first stage is the pre-processing stage where the input image is processed to reduce noise, remove any unwanted features and enhance the quality of the image. The pre-processing stage includes tasks such as normalization, binarization and thinning. The second stage involves feature extraction, where a set of relevant features is extracted from the pre-processed image.

The extracted features are used to represent the characteristics of the input

image, which are then fed into the recognition model. The features extracted for HCR include geometrical features like stroke width, size and aspect ratio and statistical features like the number of loops, ascenders and descenders. The third stage is the training stage, where the recognition model is trained using a dataset of handwritten characters. In HMM, the model is trained to learn the probability distribution of the input data using an unsupervised learning approach. In ANN and MLP, the model is trained using a supervised learning approach with backpropagation algorithm. During training, the model adjusts its weights and biases to minimize the error between the predicted output and the actual output.

The fourth stage is the testing stage, where the trained model is tested on a separate set of handwritten characters. The model predicts the output label for each input image and the accuracy of the model is evaluated using performance metrics like precision, recall and F1 score.

In HMM, the model is trained to learn the probability distribution of the input data and the Viterbi algorithm is used to decode the input sequence. The Viterbi algorithm finds the most probable state sequence that produced the input sequence.

In ANN, the trained model consists of multiple layers of neurons that process the input data to produce the output label. The neurons in each layer are connected to the neurons in the previous and next layer and the weights and biases of each neuron are adjusted during training.In MLP, the trained model is a feedforward neural network with multiple layers of neurons that use a non-linear activation function. The model processes the input data through the layers and produces the output label. The methodology for HCR using HMM, ANN and MLP involves preprocessing the input image, extracting relevant features.

Merits

• **Reduced data entry time** One of the main advantages of HCR is its ability to reduce data entry time, making it more efficient and cost effective than manual data entry.

• **Increased accuracy** HCR can achieve a higher level of accuracy in character recognition than human beings, which can result in fewer errors and improved data quality.

• **Accessibility** HCR can enable people with disabilities, such as visual impairment, to access information that might not be able to access otherwise.

• **Time-saving** HCR can save a significant amount of time in tasks such as sorting, classifying and searching through handwritten documents, making it easier to find relevant information quickly.

• **Scalability** HCR technology can be scaled up or down, depending on the needs of the user, making it suitable for a wide range of applications.

Demerits

• **Quality of input** HCR performance can be affected by the quality of the input image, which can be affected by factors such as ink quality, paper quality and handwriting style. This can result in reduced accuracy and increased error rates.

• **Language limitations** HCR systems are usually designed for a specific language may not be able to recognize characters from other languages. Additionally, HCR may not be able to recognize handwriting styles that differ from the ones used during training.

• **Cost** HCR technology can be expensive to implement and maintain, making it less accessible to small businesses and individuals.

• **Performance limitations** HCR can be computationally intensive, requiring powerful hardware and software. This can limit its performance on low-end or older devices.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

OpenCV is a popular computer vision library that provides a wide range of tools for image processing and computer vision tasks. The system consists of several modules, including image pre-processing, character segmentation, feature extraction and character recognition.

The image pre-processing module is responsible for cleaning up the input image, removing any noise or artifacts and enhancing the contrast and brightness of the image. The character segmentation module then divides the pre-processed image into individual characters, which are then processed by the feature extraction module to extract relevant features such as stroke width, aspect ratio and curvature.

The final step in the system is character recognition, where the extracted features are fed into a machine learning algorithm such as a support vector machine or artificial neural network to classify the characters. The system can recognize characters in a variety of fonts and sizes and has applications in fields such as document analysis and handwriting recognition. One of the advantages of this system is its ability to recognize characters in noisy and low-quality images.

Additionally, OpenCV is an open-source library, which makes it easily accessible to developers and researchers. However, the system may not perform well on complex images with multiple fonts and sizes and the accuracy of the system is highly dependent on the quality of the input image.

The existing system has shown promising results in recognizing characters from images and visuals with high accuracy. However, it has limitations in recognizing handwritten characters or characters in noisy environments, where additional pre-processing techniques and feature extraction algorithms may be required.

Furthermore, the system is limited to recognizing characters in a single language or script and may require language-specific training data and models for recognition in other languages or scripts.

3.1.1 Demerits

• **Variability in handwriting** Handwriting varies significantly between individuals, making it challenging to create a universal recognition system that can accurately recognize all handwriting styles.

• **Limited language support** Most handwritten character recognition systems are developed for a limited set of languages, which means cannot be used for languages with different scripts.

• **Poor accuracy** Despite significant progress, the accuracy of handwritten character recognition systems is still not perfect and errors can occur frequently. This is especially true for cursive or irregular handwriting.

• **Complex characters** Some characters in certain languages can be very complex, making them difficult to recognize accurately by a computer system.

• **Resource-intensive** Handwritten character recognition systems require significant computational resources and large amounts of training data to achieve high accuracy.

• **Time-consuming** It can take a significant amount of time to train a handwriting recognition system and even after training, the system may require significant time to process large amounts of data.

• **Limited applications** Handwritten character recognition systems are typically limited to specific applications, such as OCR or signature recognition and cannot be easily adapted for other applications.

3.2 PROPOSED SYSTEM

The proposed system focuses on creating a system that can extract the characters from visuals and images with good accuracy rate using Open Computer Vision technique. OpenCV provides a wide range of algorithms for image processing and computer vision tasks, such as feature detection and extraction, image filtering, segmentation, object recognition and tracking. It also includes a set of tools for image and video manipulation, such as resizing, cropping and color space conversion. The existing system has the good merits as well as the demerits. This project includes the merits of an existing system and it also includes the solution for the Demerits. The system includes Data collections, Pre-processing, Character segmentation, Future extraction, Classification and Validation for text extraction. The project includes the several benefits of each methodology in image processing to extract the character from the Visual images. The Machine Learning Techniques, such as support vector machines and K -nearest neighbor are used to achieve high accuracy and efficiency in text extraction.

3.2.1 Merits

- **Accuracy** OpenCV has high accuracy in character recognition, which means that the system can correctly identify the characters in the image.
- **Speed** OpenCV is designed to be highly efficient and can perform character recognition in real-time.
- **Accessibility** OpenCV is an open-source library that is freely available, making it easy to access and use for character recognition tasks.
- **Integration** OpenCV can be easily integrated with other technologies such as machine learning and artificial intelligence to further enhance its capabilities.

CHAPTER 4

SYSTEM SPECIFICATION

4.1 HARDWARE SPECIFICATION

Processor: Dual core processor

RAM: 4GB

Hard disk: 256 GB

Monitor: 15-inch colour monitor

Webcam: 5 MP

4.2 SOFTWARE SPECIFICATION

Operating system: Any

IDLE: Python

Web Application: Jupyter Notebook

Browser: Any

CHAPTER 5

ARCHITECTURAL DESIGN

5.1 SYSTEM ARCHITECHTURE

A system architecture is the conceptual model that defines the structure, behavior and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

The system architecture for "Extraction of character from Visuals and Images using OpenCV" involves several stages and components working together to achieve the desired outcome.

The first stage is image acquisition, where the input image is obtained from a source such as a camera or file. The acquired image is then pre-processed to enhance its quality and reduce noise. This stage involves image resizing, conversion to grayscale and smoothing using filters such as Gaussian or median filters.

The next stage involves segmentation, where the characters in the preprocessed image are isolated from the background. This stage can be performed using techniques such as thresholding, contour detection or edge detection. Once the characters are isolated, then subjected to feature extraction.

Feature extraction involves extracting relevant features from the segmented characters that can be used for recognition.These features can include aspects such as shape, size, curvature or texture. Principal Component Analysis or Linear Discriminant Analysis can be used to extract these features

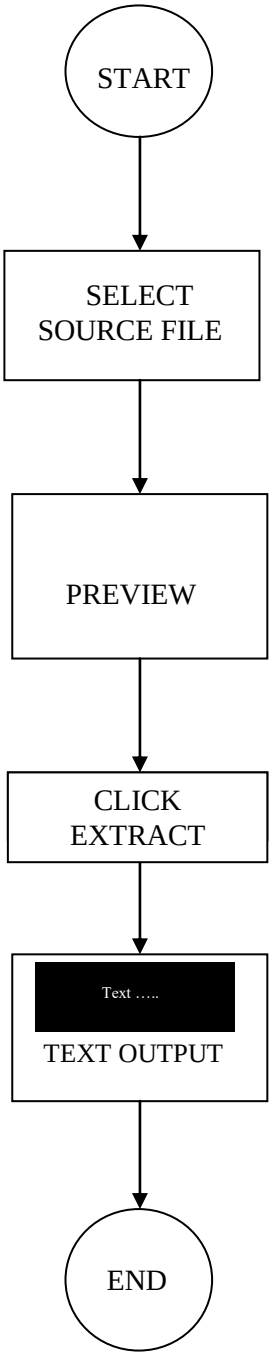

Figure.No. 5.1 System Architecture

Finally, the recognized characters are post-processed to refine the output and correct any errors that may have occurred during recognition. The postprocessing stage involves techniques such as spell-checking, error-correction and character grouping. the segmentation module, the feature extraction module, the character recognition module and the post-processing module. These components work together in a pipeline to extract characters from visuals and images accurately.

CHAPTER 6

MODULE DESCRIPTION

6.1 IMAGE CHARACTER EXTRACTION

The project abstract outlines the development of a sophisticated computer vision system designed to accurately extract characters from both document pages and image data. Leveraging the capabilities of OpenCV, Pytesseract, Matplotlib, and Tkinter, this system aims to deliver precise and efficient character extraction functionality across a diverse range of visual inputs.

At its core, the system comprises several interconnected modules. Firstly, the image pre-processing module meticulously cleans input images, eliminating noise and enhancing contrast and brightness to optimize subsequent processing steps. Following this, the character segmentation module meticulously dissects preprocessed images.

Subsequently, the feature extraction module meticulously scrutinizes these segmented characters, extracting pertinent features such as stroke width, aspect ratio, and curvature. These extracted features are then meticulously analyzed and interpreted by the character recognition module, which diligently identifies the characters based on their unique attributes.In evaluating the system's performance, a comprehensive dataset containing diverse visual inputs serves as the benchmark, scrutinizing the accuracy and efficiency of character extraction. The project's significance lies in its potential applications, spanning document digitization, text recognition, and information retrieval, where accurate character extraction plays a pivotal role.

Implemented with a meticulous attention to detail, the system encapsulates the essence of precision and efficiency in character extraction, aiming to provide a reliable solution for processing visual data. By harnessing the combined power of OpenCV, Pytesseract, and Tkinter, it endeavors to empower users with a versatile

toolset for character extraction tasks, catering to a multitude of real-world applications.

6.2 VIDEO CHARACTER EXTRACTION

This module represents a sophisticated fusion of computer vision and optical character recognition (OCR) technologies, tailored to process and extract textual information from video feeds in real-time. At its core, the integration of OpenCV provides a comprehensive suite of tools for capturing, manipulating, and analyzing video frames, ensuring efficient handling of the continuous stream of visual data. This includes functionalities such as frame retrieval, preprocessing for noise reduction, and feature extraction for subsequent analysis.

Complementing OpenCV, Tesseract OCR, accessed through the pytesseract wrapper, serves as the engine for character recognition within the video frames. Tesseract OCR is renowned for its accuracy and versatility in detecting and interpreting textual content from images, making it an ideal choice for extracting text from video frames. By harnessing the power of Tesseract OCR, the module can reliably identify characters, words, and even complex textual structures within the dynamic context of video content.

To strike a balance between computational efficiency and thorough analysis, the module selectively processes every 8th frame from the video feed. This strategy optimizes resource utilization while ensuring that a substantial portion of the video stream is analyzed for textual content. Bounding boxes provides users with immediate feedback on the recognized text's spatial distribution within the video frames.

User interaction is seamlessly incorporated into the module's design, enabling users to interact with the system through intuitive keyboard commands. This allows

users to trigger specific actions, such as printing extracted characters for further inspection or terminating the processing loop, thereby facilitating a smooth and interactive workflow.

Furthermore, the versatility of this module extends beyond mere character extraction, finding applications across diverse domains. From video surveillance and automatic transcription to content analysis and live captioning, the ability to extract textual information from video streams opens up a myriad of possibilities for extracting insights, monitoring activities, and enhancing understanding in real-time scenarios.

In essence, this module encapsulates the synergy between computer vision and OCR technologies, offering a powerful and adaptable solution for unlocking textual information embedded within video content. Its real-time capabilities, coupled with user-friendly interaction and versatile applications, position it as a valuable tool for researchers, developers, and practitioners seeking to harness the rich trove of textual data present in video streams.

6.3 VISUALS CHARACTER EXTRACTION

This module is designed to perform real-time character extraction from a live video stream using OpenCV and Tesseract OCR. It begins by initializing the video capture using the default camera (index 0). If the camera fails to open, an IOError is raised to alert the user.

Within a continuous loop, each frame from the video stream is read using cap.read(). If a frame is successfully captured (ret == True), Tesseract OCR is utilized to extract text from the frame using pytesseract.image_to_string(frame).

The extracted text is then printed for debugging purposes. Additionally, bounding

boxes are drawn around the detected characters using pytesseract.image_to_boxes(frame). These boxes visually represent the positions of the recognized characters within the frame.

To provide visual feedback to the user, the extracted text is overlaid onto the frame using cv2.putText(). This allows the user to see the recognized text displayed alongside the video feed.

The processed frame, with the overlaid text and bounding boxes, is displayed in a window titled 'Character Extraction from Visuals' using cv2.imshow().

User interaction is supported through keyboard commands. Pressing 'q' terminates the program and closes the video capture. Pressing 'e' prints the recognized text to the console along with an additional developer message.

In essence, this module encapsulates the synergy between computer vision and OCR technologies, offering a powerful and adaptable solution for unlocking textual information embedded within visual content.

Overall, this module demonstrates the seamless integration of OpenCV and Tesseract OCR for real-time character extraction from live video streams, providing users with valuable insights into the textual content present within visual data.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The proposed system for Extraction of character from Visuals and Images using OpenCV is a robust and efficient solution for extracting characters from images and visual media. The system is designed to preprocess the images, segment the characters, extract the features, classify them using machine learning algorithms and post-process the results to generate accurate output.

The user interface is designed to be user-friendly and intuitive, allowing users to easily upload images and extract the characters. The system's pre-processing module is responsible for enhancing the image quality and reducing noise, thereby improving the overall character recognition accuracy. The segmentation module uses various techniques to separate the characters from the image's background and isolate them for feature extraction.

The feature extraction module extracts relevant features from the segmented characters. The classification module uses machine learning algorithms like SVM and KNN to classify the characters based on their extracted features. The postprocessing module eliminates false positives and generates the final output. Overall, the system is expected to deliver accurate and efficient results in character extraction, making it suitable for applications like document digitization, handwriting recognition and optical character recognition. With further research and development, the system can be improved to handle complex image processing tasks and deliver better performance.

7.2 FUTURE ENHANCEMENT

The future scope is a wide open in research aspect for all applications. Various other feature extraction methods can be applied to test the accuracy of the system. There are several potential future enhancements that could be made to the "Extraction of character from Visuals and Images using OpenCV" system.

• **Multi-language support** Currently, the system is designed to recognize characters in a specific language. Expanding the system to recognize multiple languages would greatly increase its utility.

• **Deep learning-based approaches** While the current system uses traditional machine learning techniques, incorporating deep learning-based approaches such as Convolutional Neural Networks and Recurrent Neural Networks could potentially improve the accuracy and speed of character recognition.

• **Improved segmentation techniques** The segmentation module is a critical component of the system and further research could be done to develop more advanced techniques for identifying and separating individual characters from an image.

• **Real-time translation** as technology advances, it may become possible to implement the system in real-time applications such as video surveillance or document scanning.

• **Mobile application** Developing a mobile application that utilizes the character recognition system would allow users to easily extract text from images taken on their mobile devices.

Overall, there is great potential for the "Extraction of character from Visuals and Images using OpenCV" system to be improved and expanded upon in the future, making it an even more powerful tool for character recognition.

APPENDIX 1 SAMPLE CODE

Image Character Extraction

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

pytesseract.pytesseract.tesseract_cmd = 'C:\\Program Files\\Tesseract-
OCR\\tesseract.exe'

root = Tk()
root.title('Text Extraction from Images')
scrollbar = Scrollbar(root)
scrollbar.pack( side = RIGHT, fill = Y )

newline= Label(root)
uploaded_img=Label(root)


def imgextract(path):
    img = cv2.imread(path)
    Sample_img = cv2.resize(img,(400,350))

    img2char = pytesseract.image_to_string(img)
    imgH, imgW, _ = img.shape
    imgbox = pytesseract.image_to_boxes(img)
    for boxes in imgbox.splitlines():
        boxes = boxes.split(' ')
        x,y,w,h = int(boxes[1]),int(boxes[2]),int(boxes[3]),int(boxes[4])
        cv2.rectangle(img,(x,imgH-y),(w,imgH-h),(255,50,0),1)

    print("RESULT\n\n\n", img2char)
    #plt.imshow(img)
    #print("\n\n\n\n\nBounding Box\n\n",imgbox, "\n\n\n\nBounding Box Output")
    return img2char, img, imgbox
```

```
def show_extract_button(path):
    extractBtn= Button(root,text="Extract text",command=lambda:
imgextract(path),bg="#2f2f77",fg="white",pady=15,padx=15,font=('Times',15,'bold'
))
    extractBtn.pack()

def upload():
    try:
        path=filedialog.askopenfilename()
        image=Image.open(path)
        img=ImageTk.PhotoImage(image)
        uploaded_img.configure(image=img)
        uploaded_img.image=img
        show_extract_button(path)
    except:
        pass

uploadbtn = Button(root,text="Upload an
image",command=upload,bg="#2f2f77",fg="white",height=2,width=20,font=('Times
',15,'bold')).pack()
newline.configure(text='\n')
newline.pack()
uploaded_img.pack()


root.mainloop()
```

Video Character Extraction

```
import pytesseract
import cv2
import matplotlib.pyplot as plt
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-
OCR\tesseract.exe'

font_scale = 1.5
font = cv2.FONT_HERSHEY_PLAIN

cap = cv2.VideoCapture("test.mp4")
```

```
if not cap.isOpened():
    cap = cv2.VideoCapture(0)
if not cap.isOpened():
    raise IOError("Cannot open Video")


cntr = 0;
while True:
    ret,frame = cap.read()
    cntr =cntr+1;
    if((cntr%8)==0):

        imgH,imgW,_ = frame.shape

        x1,y1,w1,h1 = 0,0,imgH,imgW

        imgchar = pytesseract.image_to_string(frame)

        imgboxes = pytesseract.image_to_boxes(frame)
        for boxes in imgboxes.splitlines():
            boxes = boxes.split(' ')
            x,y,w,h = int(boxes[1]),int(boxes[2]),int(boxes[3]),int(boxes[4])
            cv2.rectangle(frame,(x,imgH-y),(w,imgH-h),(0,0,255),3)
            cv2.putText(frame, imgchar, (x1 + int(w1/50),y1
            int(h1/50)),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0,0,255), 1)


        font = cv2.FONT_HERSHEY_SIMPLEX

        cv2.imshow('Character Extraction form video - Batch 11', frame)

        if cv2.waitKey(2) == ord('e'):
            print(imgchar)

        if cv2.waitKey(1) == ord('q'):
            break

cap.release()
cv2.destroyAllWindows()
```

Visuals Character Extraction

```
import pytesseract
import cv2
import matplotlib.pyplot as plt
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-
OCR\tesseract.exe'


font_scale = 1.5
font = cv2.FONT_HERSHEY_PLAIN

cap = cv2.VideoCapture(1)

if not cap.isOpened():
    cap = cv2.VideoCapture(0)
if not cap.isOpened():
    raise IOError("Cannot open Camera")


cntr = 0;
while True:
    ret,frame = cap.read()
    if((cntr%2)==0):

        imgH,imgW,_ = frame.shape

        x1,y1,w1,h1 = 0,0,imgH,imgW

        imgchar = pytesseract.image_to_string(frame)

        imgboxes = pytesseract.image_to_boxes(frame)
        for boxes in imgboxes.splitlines():
            boxes = boxes.split(' ')
            x,y,w,h = int(boxes[1]),int(boxes[2]),int(boxes[3]),int(boxes[4])
            cv2.rectangle(frame,(x,imgH-y),(w,imgH-h),(0,0,255),1)

        cv2.putText(frame, imgchar, (x1 + int(w1/50),y1 + int(h1/50)),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255,0,0), 2)

        font = cv2.FONT_HERSHEY_SIMPLEX
```

```
        cv2.imshow('Character Extraction form Visuals - Batch 7', frame)

        if cv2.waitKey(1) == ord('q'):
            cv2.destroyAllWindows()
            break


cap.release()
cv2.destroyAllWindows()
```

APPENDIX 2 SCREENSHOTS

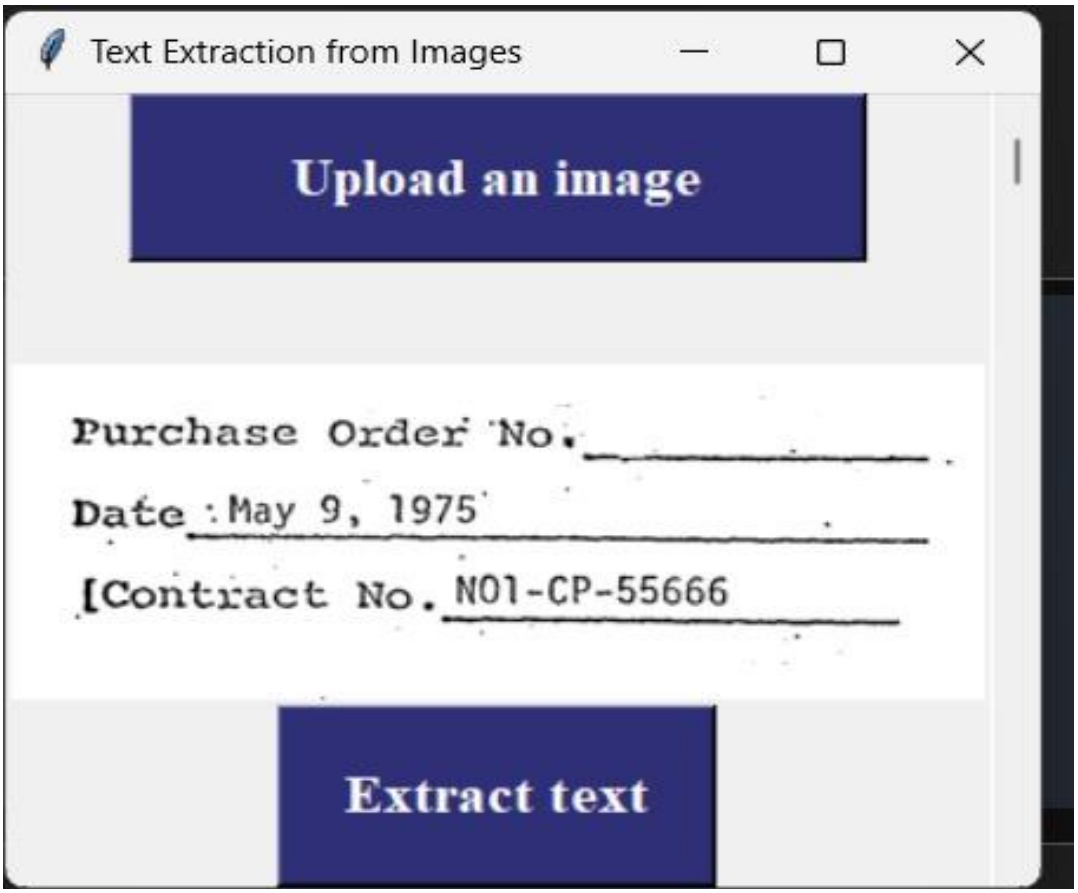

Figure. No. A.2.1 Image Character Extraction

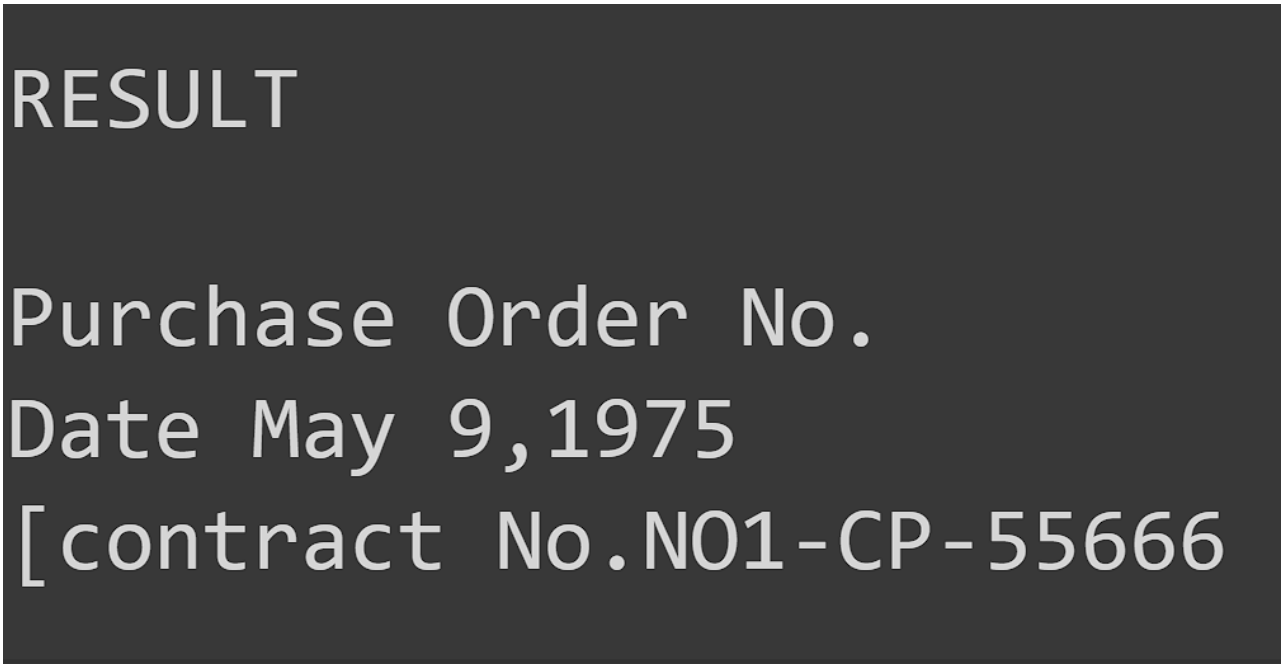

Figure. No. A.2.2 Image Character Extraction

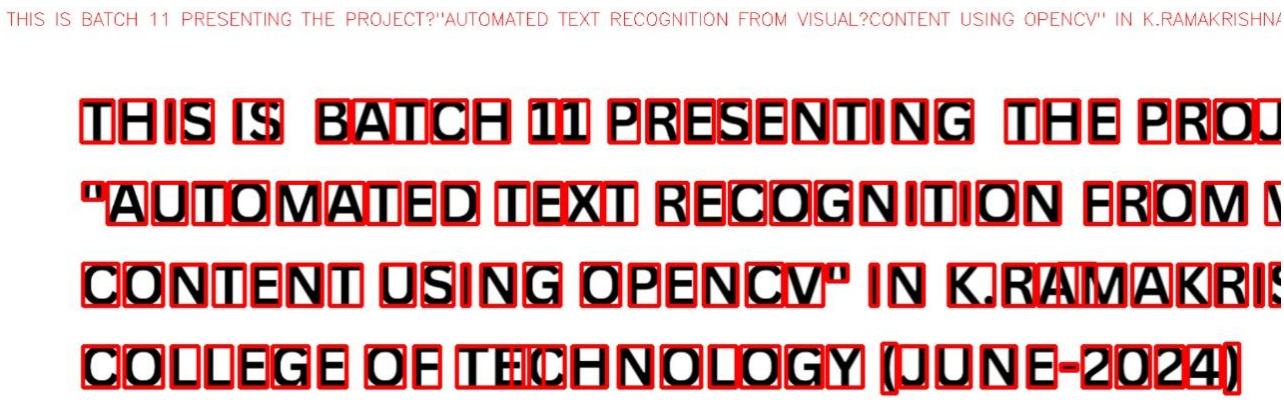

Figure. No. A.2.3 Video Character Extraction

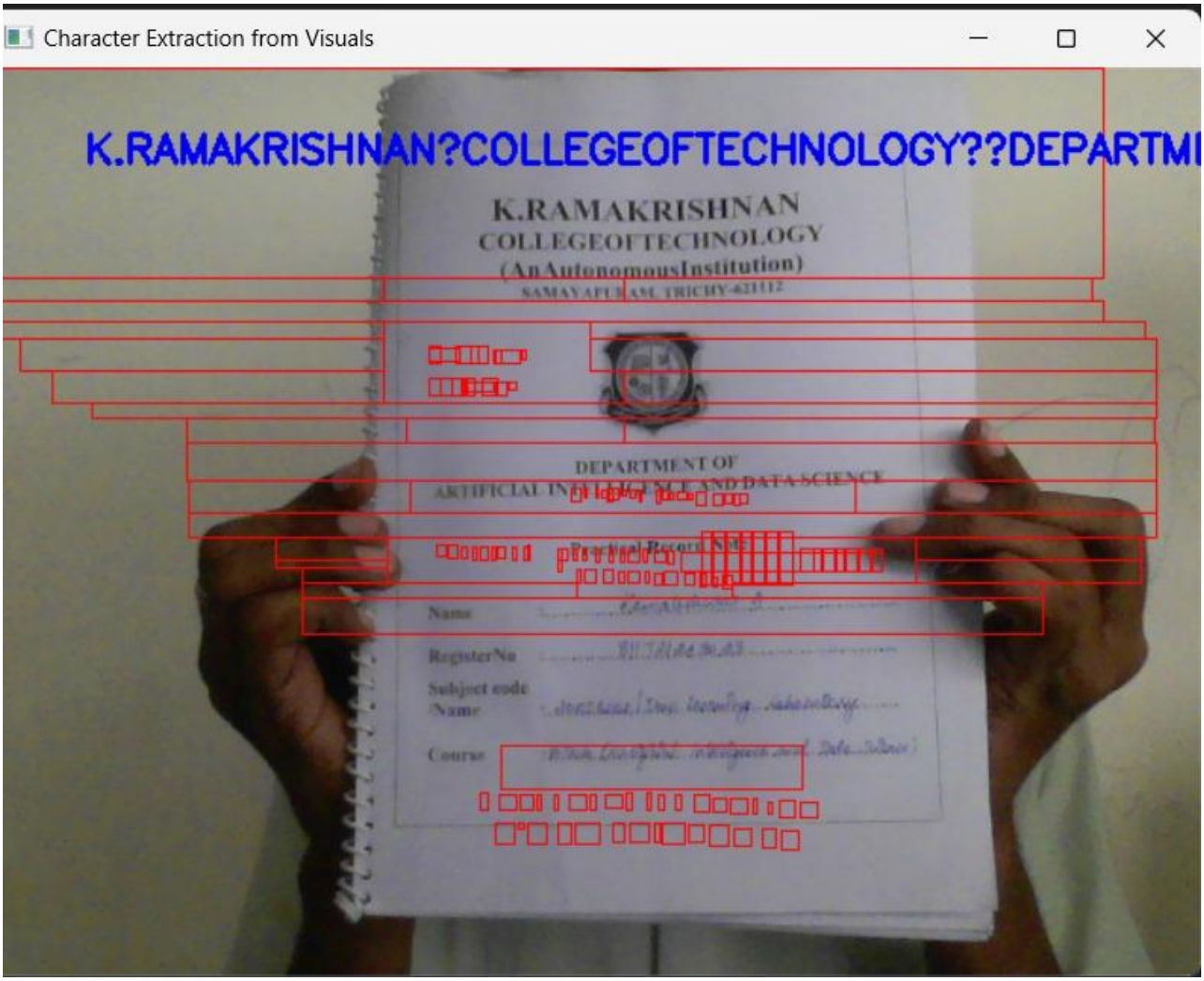

Figure. No. A.2.4 Visual Character Extraction

REFERENCES

1. Ayush Purohit, Shardul Singh Chauhan, (2016) "Handwritten Character Recognition using Neural Network" International Journal of Computer Science and Information Technologies, Vol. 7.

2. Mahmood K Jasim, Anwar M Al-Saleh, Alaa Aljanaby, (2013) "A Fuzzy Logic based Handwritten Numeral Recognition System" International Journal of Computer Applications (0975 – 8887) Volume 83 – No 10.

3. Megha Agarwal, Shalika, Vinam Tomar, Priyanka Gupta (2019) "Handwritten Character Recognition using Neural Network and Tensor Flow" International Journal of Innovative Technology and Exploring Engineering ISSN: 2278-3075, Volume-8, Issue- 6S4, April 2019.

4. Mohammed Z. Khedher, Gheith A. Abandah, and Ahmed M. AlKhawaldeh, (2005) "Optimizing Feature Selection for Recognizing Handwritten Arabic Characters", proceedings of World Academy of Science Engineering and Technology, Vol. 4, ISSN 1307-6884.

5. Nafiz Arica, and Fatos T. Yarman-Vural, (2002) "Optical Character Recognition for Cursive Handwriting", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.24, no.6, pp. 801-113.

6. Reena Bajaj, Lipika Dey, and S. Chaudhary, (2002) "Devanagari numeral recognition by combining decision of multiple connectionist classifiers", Sadhana, Vol.27, part. 1, pp.-59-72. 55