References

There are no sources in the current document.

MEMORIAL
UNIVERSITY

IMAGE PROCESSING AND APPLICATIONS
ENGI 9804

Final Report :
Text Recognition System from Handwritten Text

Submitted By

Nayem Al Tareq :202293442

Abstract

This study explores the concept of handwriting recognition through the use of image processing techniques. Handwriting recognition refers to the capability of devices such as computers and smartphones to identify and interpret handwritten text. This technology can process inputs from various sources, including printed paper documents, photographs, and other digital media. It can also recognize handwriting entered directly onto a touchscreen or other input devices, converting it into digital text. The input for handwriting recognition systems typically involves images, such as photographs of handwritten notes, or real-time scanning through cameras. These systems utilize optical scanning methods to capture the text, which is then processed using pattern-recognition algorithms. The algorithms analyze the shapes, patterns, and strokes of the handwritten characters to translate them into digital text. Handwriting recognition technology has significant applications in various fields. It enhances digital note-taking, making it easier to convert written notes into editable text. It is also used in document digitization, allowing for the efficient storage and retrieval of information from handwritten documents. Additionally, this technology is valuable in accessibility tools, aiding individuals with disabilities in inputting and reading text. The development and refinement of handwriting recognition systems continue to advance, driven by improvements in artificial intelligence and machine learning. These advancements enable more accurate and reliable recognition of diverse handwriting styles and languages, expanding the usability and functionality of the technology across different platforms and applications.

Introduction

Handwriting recognition refers to the capability of computers and mobile devices to interpret handwritten input as text. In the modern mobile landscape, this technology is frequently used for direct input on touchscreens using a stylus or finger. This approach is convenient for quickly jotting down information like names and phone numbers, as it is often faster and more intuitive for many users than typing on a virtual keyboard. While most smartphones and tablets do not include handwriting recognition as a built-in feature, there are numerous apps available that provide this functionality.

The most prevalent technique for handwriting recognition is optical character recognition (OCR). OCR works by scanning a handwritten document and converting it into a digital text file. This process can also be applied by capturing an image of handwritten text. Essentially, OCR is a form of image recognition technology tailored specifically to identify handwritten characters, as opposed to recognizing faces or other features such as landscapes.

Literature Review

[1] Karthikeyan, U., & Vanitha, M. (2019). A Study on Text Recognition using Image Processing.

This paper provides an overview of various text recognition techniques, methods, and algorithms. It includes a discussion of the accuracy of different text recognition algorithms based on a review of the literature. The paper details the steps and processes involved in text recognition, including image acquisition, preprocessing, feature extraction, classification, and post-processing, as described in numerous research articles. Additionally, the paper examines the advantages and disadvantages of different text recognition algorithms.

[2] Kaur, Ravneet. "Text recognition applications for mobile devices." Journal of Global Research in Computer Science 9, no. 4 (2018).
In this study, the authors employed techniques such as binarization, layout analysis, and block segmentation for image preprocessing. For post-processing, they utilized character segmentation with normalization, feature extraction, and classification. These methods enabled them to effectively handle complex content, even in noisy images and with varying font styles.

[3]P. Shivakumara, A. Dutta, U. Pal and C. L. Tan, "A New Method for Handwritten Scene Text Detection in Video," 2010 12th International Conference on Frontiers in Handwriting Recognition, Kolkata, India, 2010, pp. 387-392, doi: 10.1109/ICFHR.2010.67.

In this article, the authors demonstrate that handwritten text often appears in video images, making handwritten scene text detection in videos essential and useful for various applications, such as efficient indexing and retrieval. Many video frames may contain text lines that are multi-oriented. To the best of our knowledge, no prior work has focused on the detection of multi-oriented handwritten text in videos. This paper presents a new method based on maximum color difference and boundary growing techniques for detecting multi-oriented handwritten scene text in videos. The method calculates the maximum color difference for the average of the R, G, and B channels of the original frame to enhance text information. The output of the maximum color difference calculation is processed using the K-means algorithm with K=2 to separate text and non-text clusters. Text candidates are identified by intersecting the text cluster with the Sobel output of the original frame. To address the challenge of varying orientations and skews in handwritten text, a boundary growing method based on a nearest neighbor concept is used. The proposed method is evaluated using our own handwritten text database and publicly available video data (Hua's data). The experimental results obtained from the proposed method are promising.

[4]M. Carbonell, J. Mas, M. Villegas, A. Fornés and J. Lladós, "End-to-End Handwritten Text Detection and Transcription in Full Pages," 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), Sydney, NSW, Australia, 2019, pp. 29-34, doi: 10.1109/ICDARW.2019.40077.

When transcribing handwritten document images, inaccuracies in the text segmentation step often cause errors in the subsequent transcription step. For this reason, some recent methods propose to perform the recognition at paragraph level. But still, errors in the segmentation of paragraphs can affect the transcription performance. In this work, we propose an end-to-end framework to transcribe full pages. The joint text detection and transcription allows to remove the layout analysis requirement at test time. The experimental results show that our approach can achieve comparable results to models that assume segmented paragraphs, and suggest that joining the two tasks brings an improvement over doing the two tasks separately.

Methodology

The primary goal of the Handwriting Recognition project is to develop an intuitive and precise system for converting printed and handwritten text into a digital format. The system is designed to support multiple languages, ensure rapid processing, and provide clear output that can be customized for accessibility. Privacy and security features will be implemented to safeguard user data, and the technology will be continuously improved based on user feedback to enhance its performance. The ultimate objective is to create a versatile OCR solution compatible with various devices and platforms, streamlining the process of text digitization.

In summary, the main objectives of our project include:

- Demonstrating the conversion of images to text using MATLAB.
- Highlighting various types of noise and the pre-processing techniques used to address them.
- Implementing line segmentation and character segmentation.
- Recognizing characters using a set of predefined templates.
- Outputting the words found in images as text in a text file.

Morphological Opening[5]

Image after segmentation

Image after segmentation and morphological processing

Character Identification

- Each character is individually segmented.
- The segmented character is resized to a standard dimension of [24 42] in this context.
- Each character is compared against a collection of predefined character templates.
- The Corr2 function in MATLAB is utilized to calculate the correlation factor between the segmented character and the template set.
- The character that achieves the highest score is selected as the identified character.

The template.m code starts by importing images of capital letters (A to Z), lowercase letters (a to z), and numbers (0 to 9) from a directory named "letters_numbers." These images are assigned variable names like A, a, 1, and so on. The code organizes these images into arrays for capital letters, numbers, and lowercase letters. These individual arrays are then combined into a single array named "character." The combined array is split into a cell array called "templates" using the mat2cell function, with each cell containing a grid-like representation of the characters. This "templates" cell array, which encapsulates various character sets, is saved as "templates.mat."

The "extract_lines.m" code defines a function called "extract_lines," which processes an input image containing text lines. The function begins by preprocessing the image with the "clip" function to remove unnecessary white space. It then calculates the number of columns in the preprocessed image. The code iterates through each column using a loop.

When a column sum is zero, indicating the end of a line, the following steps occur:

- The first letter of the line is extracted and processed using the "clip" function and stored in the variable " lineImage."
- The remaining letters are extracted, processed, and stored in the variable " remainingImage."

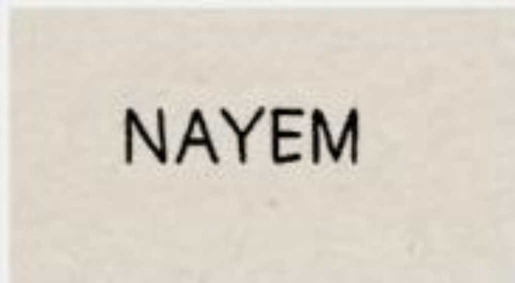
The variable "space" measures the width difference between the original line and the remaining letters stored in "re." The loop terminates upon detecting the end of a line. If the column sum is not zero, indicating that the line continues, the entire line is treated as the first line ("lineImage "), with no remaining letters, meaning "remainingImage " is left empty. The "clip" function trims unnecessary white space around the text by identifying the indices of non-zero pixels. This code is part of a larger text processing pipeline designed to segment an image into individual lines and letters for further analysis or recognition.

The “character_recognition” function performs character recognition by comparing the input character image with a set of predefined templates. It selects the template that shows the highest similarity to the input character. This approach is a fundamental example of template-based character recognition, where each character is associated with a specific template, and the recognition is based on identifying the closest match among the templates.

Results

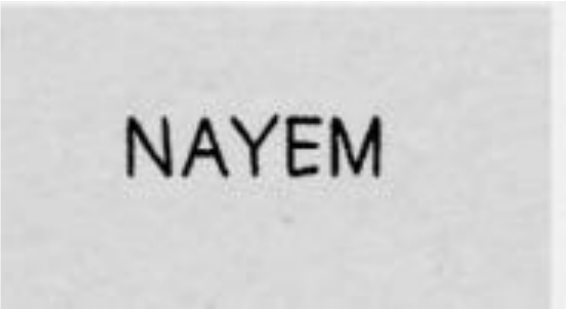
We handled multiple images, processing them through various steps to extract characters. These characters were then saved as text files automatically in the working directory. The path to the image is specified within the code and can be modified to point to different images. The input and outputs are given for demonstration:

Input Image: This is the input image and our goal is to extract characters from it and transform it into a txt. File.

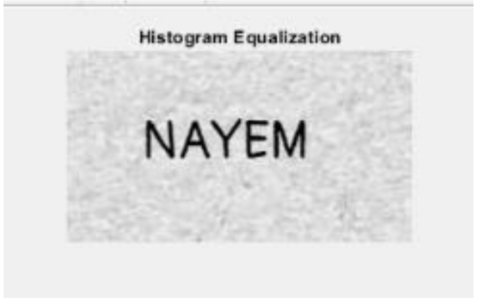


NAYEM

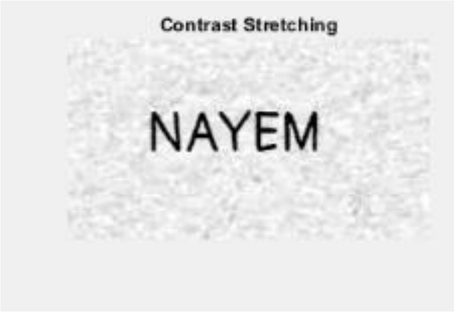
Grayscale Conversion : First, We converted the input file into Grayscale Conversion.



Histogram Equalization : We reduced the noises by histogram equalization and make it a clear image.

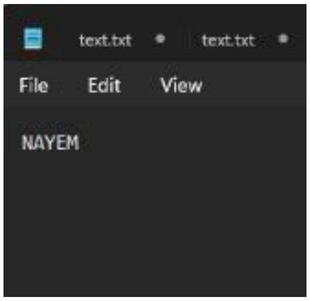


Contrast Stretching: The contrast was stretched and then converted the image to black and white .



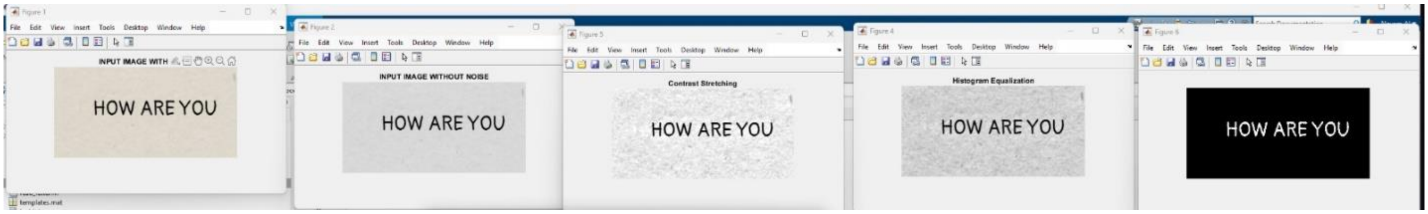
Conversion to Black and White image: We converted the image to black and white to process the pixels with our predefined templates.

Output:

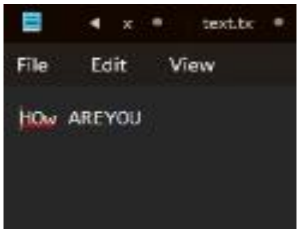


NAYEM

a. Our code failed partially to identify characters for the following image



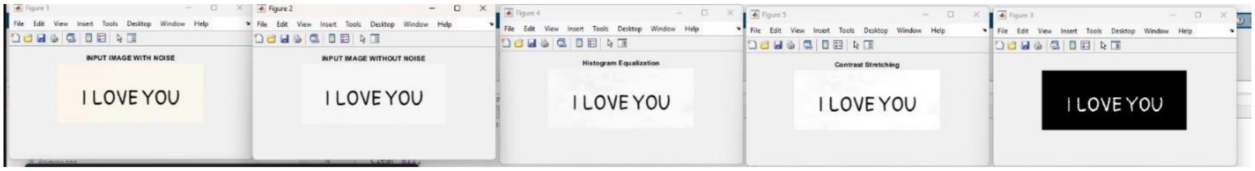
Output Image:



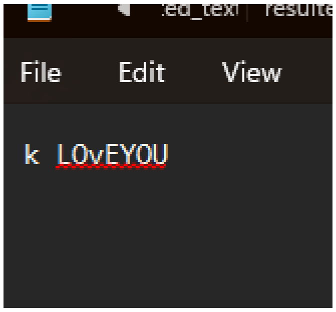
HOw AREYOU

We used a file named “hru.jpg,” with “How are you” written on it. But our code identified the ‘w’ as lower case and didn’t give the space between of Are and You.

b. For another handwritten image our code failed in most of the cases.

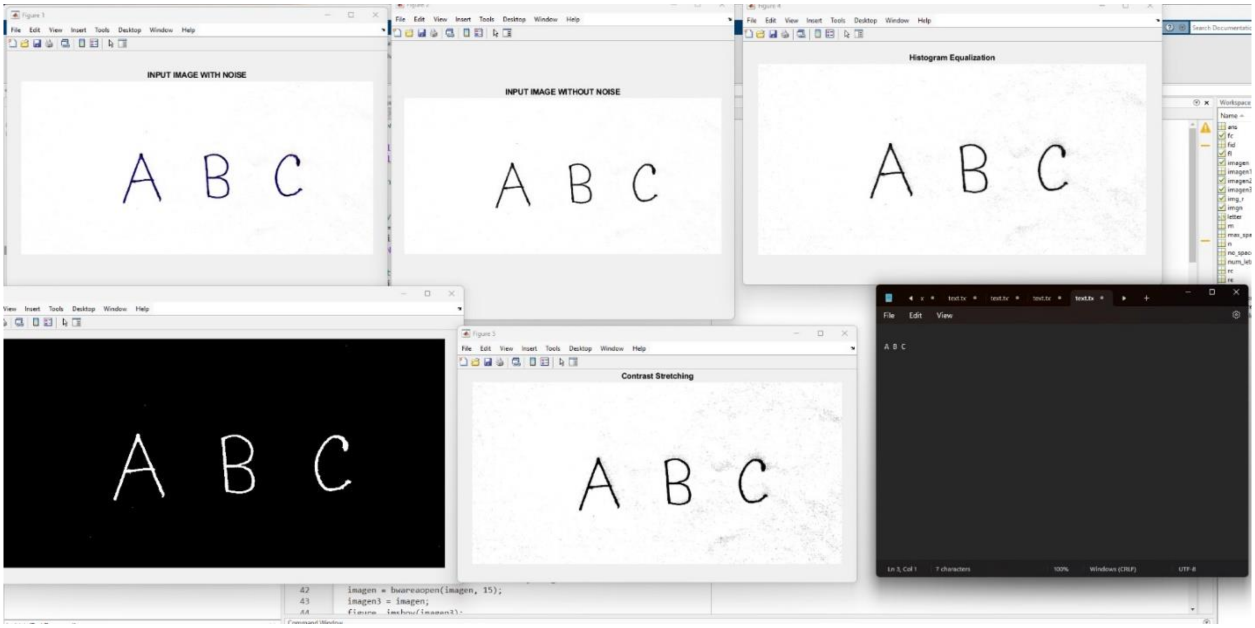


Output Image:



```
k LOvEYOU
```

c. Another successful result



```
A B C
```

```
imagen = bwareaopen(imagen, 15);
imagen3 = imagen;
figure, imshow(imagen3);
```

Discussion

Based on the results, the system performs well when the handwritten text is clear and straight. However, it struggles with images that have curved or distorted text. To improve accuracy, the program can be trained with a more comprehensive and precise dataset. This training should include a variety of images featuring different forms of alphabets and numbers to enhance the template's recognition capabilities. By expanding the dataset and refining the training process, the system's ability to accurately process diverse handwriting styles and orientations will improve, leading to higher overall success rates. This approach will also help the program become more adaptable to different handwriting variations, increasing its reliability and effectiveness in real-world applications.

Conclusion

The primary objective of this project was to develop a handwriting recognition application to assist individuals who struggle with reading handwritten text. The project successfully demonstrated the ability to accurately convert both printed and handwritten text into machine-readable formats. This functionality facilitates efficient data entry, searching, and analysis, significantly streamlining workflows and minimizing human error. Additionally, the project has contributed to greater accessibility and inclusivity by making textual content more accessible to people with visual impairments. Beyond its current capabilities, this technology has the potential to be used for converting large volumes of handwritten text into digital formats, such as DOCX files. As OCR technology advances, there are numerous opportunities for further enhancements. Continuous training with machine learning can improve accuracy, while expanding multilingual support can make the application more versatile and globally applicable. Integration with emerging technologies, such as natural language processing and augmented reality, could also provide new functionalities, such as real-time translation and interactive text overlays. However, several challenges must be addressed to realize these opportunities fully. Handling complex layouts and degraded documents remains a technical hurdle, as these factors can significantly impact the accuracy of text recognition. Moreover, as the technology becomes more sophisticated and widely used, ensuring data privacy and security becomes increasingly important. Protecting sensitive information and maintaining user trust are critical considerations that must be prioritized. The current model is relatively basic but serves as a solid foundation for future development. With further refinement, the system can be enhanced to provide a more user-friendly experience. This could include improvements in the user interface, making it more intuitive and accessible to a broader range of users, including those with limited technical skills. Additionally, incorporating features such as customizable settings and adaptive learning algorithms can help cater to individual user needs and preferences, further enhancing the application's usability and effectiveness.

Overall, while the project has already achieved significant milestones, it is poised for further growth and development. By addressing current limitations and exploring new technological integrations, the handwriting recognition application can become an invaluable tool for a wide range of users. The focus on continuous improvement and user-centered design will be key to its success, ensuring that it remains relevant and effective in an ever-evolving digital landscape. This approach will not only enhance the functionality of the application but also contribute to broader goals of accessibility, inclusivity, and efficiency in information management.

References:

[1] Karthikeyan, U., & Vanitha, M. (2019). A Study on Text Recognition using Image Processing

[2] Kaur, Ravneet. "Text recognition applications for mobile devices." Journal of Global Research in Computer Science 9, no. 4 (2018).

[3]P. Shivakumara, A. Dutta, U. Pal and C. L. Tan, "A New Method for Handwritten Scene Text Detection in Video," 2010 12th International Conference on Frontiers in Handwriting Recognition, Kolkata, India, 2010, pp. 387-392, doi: 10.1109/ICFHR.2010.67.

[4]M. Carbonell, J. Mas, M. Villegas, A. Fornés and J. Lladós, "End-to-End Handwritten Text Detection and Transcription in Full Pages," 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), Sydney, NSW, Australia, 2019, pp. 29-34, doi: 10.1109/ICDARW.2019.40077.

[5] https://towardsdatascience.com/understanding-morphological-image-processing-and-its-operations-7bcf1ed11756

[6] Extensive help was taken from ChatGPT,StackOverflow, MATLAB Site ,Google,Lecture Notes,Gemini.