

Non Cursive Handwritten Text Recognition

Ajeet Ujjwal (B14CS004)

Annuay J (B14CS006)

Mentor: Dr. Chiranjoy Chattopadhyay

Abstract

Handwriting recognition has been an active area of interest for a long time. In this work, we present an effective way to segment a non-cursive handwritten document into characters. The upcoming stages of the project will focus on recognizing these character images. The given image is assumed to be noise free, without stray marks, smudges and borders. The image is binarized using Otsu's method. Lines and word segmentation is implemented using horizontal and vertical histograms respectively. Connected components are used to segment the characters.

In the next phase of the project, machine learning algorithms will be implemented and trained with character image data so that characters can be successfully recognized.

Problem Statement

To convert non cursive handwritten text into computer readable format with high accuracy. Scanned/photographed image of handwritten text will be taken as input and converted text will be saved in a text file. The handwritten text must be in English and must contain only the following- uppercase and lowercase letters, digits 0-9 and punctuation marks used in English.

This semester, we have implemented till the stage of character segmentation. The final output at this stage is the organized list of character images.

Introduction

Computerized text has been widely preferred over handwritten documents in the recent past. Yet there are several areas where handwriting is still preferred, such as class notes, postal addresses, hand filled forms, etc. It is hard to scan through information contained in these documents using a computer, unless they are converted into machine recognizable text. This is where handwriting recognition is most required.

The purpose of this project is to take handwritten documents as input in an image file and generate the machine recognizable text of its contents. Therefore the project is solely on offline handwriting recognition. The project assumes that the handwritten document is fully in English and consists of only uppercase and lowercase letters, digits 0-9 and punctuations commonly used in English language.

The project is divided into two phases, the first of which will be to segment the handwritten document to character level. The second phase will be aimed at recognizing these characters and the document as a whole. Currently we will be working with non-cursive writings and are planning to extend it to cursive as well in future.

The first step phase 1 is to binarize the image. This will be followed by line segmentation, word segmentation and finally character segmentation.

Phase 2 will involve the implementation of a machine learning algorithm to recognize the segmented characters. The algorithm will be trained and the accuracy will be tested using the datasets. The performance of the algorithm as well as the software will be evaluated.

This report has been written after the completion of phase 1 of the project.

Brief Literature Survey

Extensive research has been done by various people on the topic of handwriting recognition. We are referencing in brief some of the works we encountered while exploring the topic.

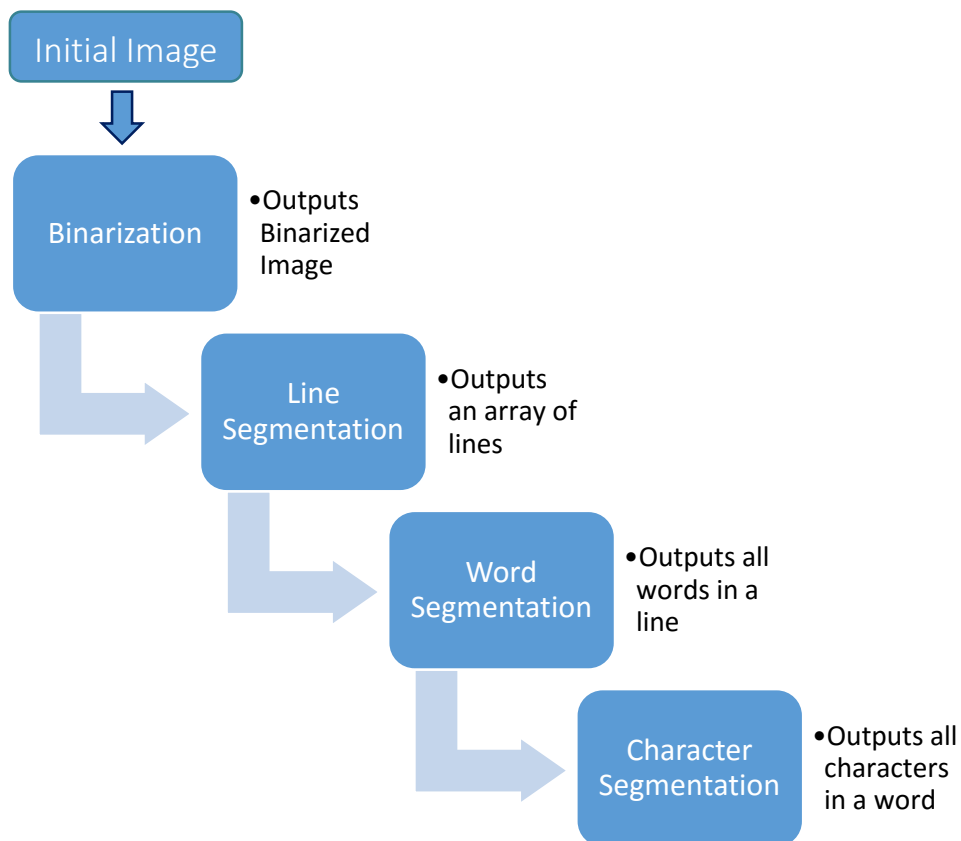
N. Nain and S. Panwar in the paper “Handwritten Text Recognition System Based on Neural Network” explains the use of artificial neural networks to recognize handwritten cursive and non-cursive text. The pre-processing involved is also given in great depth.

G. Louloudis et al. in the paper “Line and Word Segmentation of Handwritten Documents” explains in detail the segmentation of handwritten documents into lines and words using connected components.

Assumptions

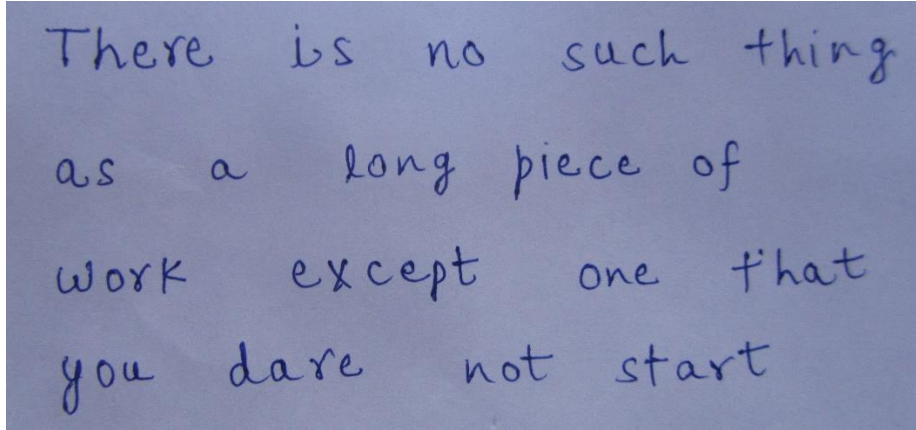
- The level of noise in the image is not very high. Also, the image must be of high pixel density and even lighting.
- The lines are aligned to the baseline of the paper. Lines must not be slanted but they may contain skewed words within them.
- The characters and words must not touch each other. Gap between words must be several times more than the gap between letters.
- The paper used must be white and without any smudges, stray marks, lines and borders.
- The dot symbol on letters 'i' and 'j' are not considered for segmentation.

Flow Chart



Methodology

The following is the original image that is given as input to Binarization stage:



The following are the steps used in character segmentation:

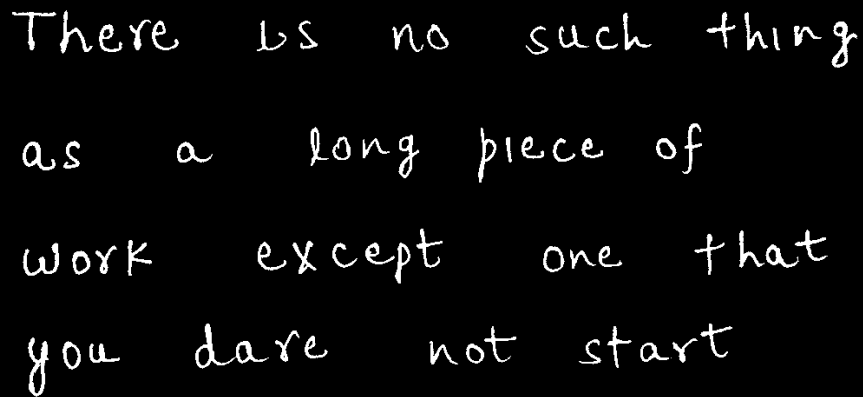
1. Binarization

A custom Binarization technique is applied to the original image to ensure optimal Binarization for the subsequent stages.

Algorithm:

- i. Convert the original image from RGB to grayscale.
- ii. Find the threshold value of intensity using graythresh function of MATLAB.
- iii. Convert the image into binary based on threshold value.
- iv. Remove all objects with pixel length less than 600. This acts as a simple noise filter.

Binarized image:



There is no such thing
as a long piece of
work except one that
you dare not start

2. Line Segmentation

Algorithm:

- i. Calculate the sum of intensities across all columns for each row of pixels in the image (create a horizontal histogram of intensities).
- ii. Find all the row regions in which this value is non zero. These are the rows in which text is present. Flag these rows with value 1 and those without text with value 0.
- iii. Iterate across all rows. Points where flag value goes from 0 to 1 represents the starting of a line. Points where flag value goes from 1 to 0 represents the ending of a line.
- iv. Store the rows in which text is present. These will be used in word segmentation.

The first line segmented from binarized image:



There is no such thing

3. Word Segmentation

Algorithm:

- i. Calculate the sum of intensities across all rows for each column of pixels in the line (create a vertical histogram of intensities).
- ii. Find all the column regions in which this value is zero and the region is considerably large (greater than 85 columns). The minimum length of the region is set to ensure that gaps between characters do not lead to segmentation. These are the columns in which words are not present. Flag these columns with value 0 and those with words with value 1.

- iii. Iterate across all columns. Points where flag value goes from 0 to 1 represents the starting of a word. Points where flag value goes from 1 to 0 represents the ending of a word.
- iv. Store the columns in which text is present (words). These will be the input to character segmentation.

Words segmented from first line:



4. Character Segmentation

Algorithm:

- i. Identify all the all the connected components in the input word.
- ii. Iterate through the list of connected components and save each of them into a separate image.
- iii. These are the characters that will be passed to the character recognition algorithm.

Segmented characters from first word:



Experiments and Results

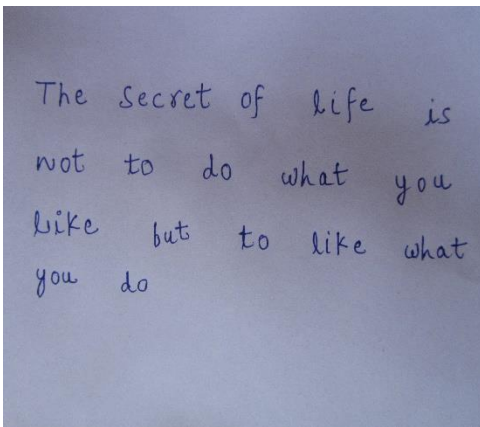
- Images without much noise, even lighting and high pixel density when processed by the program gave accurate results.
- Images that contained high levels of noise were fed into the program. This resulted in poor binarization of the image.
- Documents with ordered lines were processed accurately by the program. However, the program could not identify the line breaks in documents where lines were highly slanted.
- Images with lot of punctuation marks were fed into the character segmentation algorithm. The algorithm segmented them accurately, but could not differentiate between full stops and dot symbol in 'i' and 'j'.
- Custom binarization using global threshold was implemented earlier. This approach was discarded due to high levels of noise.

- Character segmentation using Bounding Box region property of MATLAB was used earlier. This approach was dropped, as it could not segment characters that shared the same column(s).

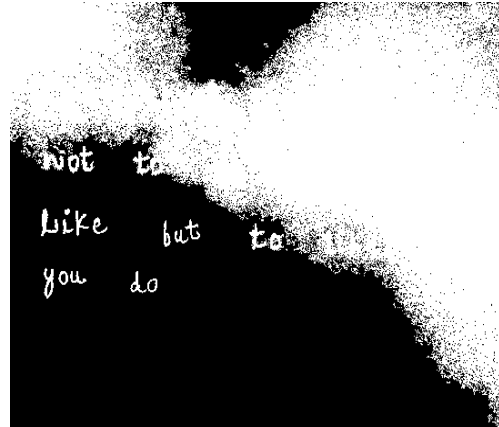
Failure case analysis

- Image with lot of noise:

Input:



Output:



Analysis: Image was not properly pre-processed to remove noise before sending to program.

- Word with letters touching:

Input:



Output:



Analysis: Letters must not touch each other in any word.

- Word containing letter 'i' is sent into character segmentation

Input:



Output:



Analysis: The dot symbol over 'i' is too small and is treated as noise.

Conclusions

The following conclusions were derived regarding the performance of handwritten document segmentation program:

- Images that were scanned using a scanner were processed very well. Images taken using camera also worked fine provided that the pixel density was high and the lighting conditions were even.
- Images that contain very high levels of noise needs to be preprocessed before being fed into the program. The same needs to be done for images with borders and underlined words/ sentences.
- Line segmentation algorithm performed well on lines that were parallel to the baseline of image. It can also process lines containing skewed words, but not lines with slant.
- Word segmentation algorithm can accurately process and segment lines with almost full accuracy. Gap between words must be sufficient.
- Character segmentation works with full accuracy on words in which characters do not touch each other. Gap between characters should not be the order of gap between words.
- Character segmentation cannot differentiate between full stops and dot symbols in 'i' and 'j' characters. All other alphabets, digits and punctuations will be segmented accurately and in correct order.

References

1. N. Nain and S. Panwar, *Handwritten text recognition system based on neural network*, Academy Publish Journal of Computer and Information Technology, Vol.2, No. 2, Pages 88-97, 2012.
2. G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, *Text Line and Word Segmentation of Handwritten Documents*, Pattern Recognition, Vol.41, No. 12, Pages 3758-3772, 2008.

3. S. Mathur, V. Aggarwal, H. Joshi, A. Ahlawat, *Off-Line Handwriting Recognition using Genetic Algorithm*, Proc. Of Sixth International Conference on Information Research and Applications – i.Tech, Varna, Bulgaria, 2008.
4. A. Dutta, J. Llados, and U. Pal, *Bag-of-GraphPaths Descriptors for Symbol Recognition and Spotting in Line Drawings*, Proc. Of 9th international conference on Graphics Recognition: new trends and challenges, 2011.
5. Vorobyov, M., *Shape Classification Using Zernike Moments*, Proc. Of iCamp-University of California Irvine, 2011.