Document Image Binarization for Degraded Document Images along with OCR

Prof. D. D. Pukale¹, Aparna Thorat², Pooja Bora³, Shruti Barmecha⁴, Dipika Suryawanshi⁵

¹,²,³,⁴,⁵ Department of computer engineering, Bharati Vidyapeeth’s college of engineering for women, Pune-43

Abstract: In the field of Image Processing there are various newly developed techniques and algorithms due to which the binarization technique is becoming more and more efficient. In this paper, we have implemented the binarization technique on the degraded document images to remove the noise and then the OCR technique to retrieve the characters. In our system we have made binarization and OCR more efficient by combination of most efficient algorithm for binarization and OCR. We are using RGB to Greyscale, Otsu’s Thresholding, Local thresholding, Manual thresholding for binarization of degraded document image. If still any noise remains after binarization the post processing technique is applied to remove the noise. Along with the binarization, the OCR technique is applied to preserve the content of the document image in text format. OCR technique implemented by the algorithms like Thinning, Segmentation, Scaling, Template Generation and Template Matching and rich text document is generated. This text format of the content helps in preserving the huge data.

Keywords: OCR, Binarization, Thresholding, Otsu’s Thresholding, Local Thresholding, Degraded document images.

1. Introduction

Document Image Binarization is performed in the very first stage for document analysis and it goal is to separate the foreground text from the document background. It is important to ensure that the document image binarization technique should be fast and accurate so that document image processing tasks such as optical character recognition (OCR) should be performed efficiently. The document image binarization has been improved in many years due to development of many efficient algorithms the thresholding of degraded document images is still a problem due to the high inter/intravariation between the text stroke and the document background across different document images. The variation of noise in different types of document degradations tend to induce the document thresholding error and thus the document image binarization is a big challenge. The new document binarization technique extends the older techniques. It makes use of more efficient thresholding technique for binarization like otsu’s thresholding. Local Thresholding and different OCR technique to convert the content into text format. In particular, the binarization with OCR technique can reduce the human error and time consumption thus, making the project more efficient.

2. Related Work

Bolan Su[1], For a given degraded document image, firstly an adaptive contrast map is constructed and through the combination of the binarized adaptive contrast map and the canny edge map the text stroke edges are then detected. Based on the local threshold the text is then segmented that is estimated from the detected text stroke edge pixels.

Faisal Mohammad[4], the main functional modules in this OCR systems are: Image acquisition module, pre-processing module, and feature extraction module and pattern generation. Feature extraction is the process of getting information about an object or a group of object in order to facilitate classification. The track-sector matrix is generated then matched with existing template. The existing template consist of each track-sector intersection value, each track value and each sector value. If all these parameters are found to match with the template values then the resultant is the character identified.

Poonam R. Deokar[3], The central objective of this system is demonstrating the capabilities of the artificial neural network in recognition of characters. Feed forward network are commonly used for performing recognizing due to high noise tolerance.
3. Proposed System

Our system is divided into following modules:

3.1 Pre-processing.
  3.1.1 RGB Separation
  3.1.2 Grayscale

3.2 Binarization.
  3.2.1 Manual Thresholding.
  3.2.2 Local Thresholding.
  3.2.3 Otsu’s thresholding.

3.3 Post Processing.

3.4 OCR Pre-processing
  3.4.1 Thinning.
  3.4.2 Segmentation and Scaling.

3.5 Template Generation and Matching

Description:

3.1 Pre-Processing.

3.1.1 RGB Separation.

3.1.2 Grayscale.

The coloured image is converted into a image having grey shades using the greyscale technique other than black and white. It is an important technique in the field of image processing.

RGB to Greyscale conversion algorithm contains the following steps [3].

Algorithm

1. Row by row traverse through the input image.
2. The color values for each pixels are read (24-bit).
3. R, G and B 8bit values are found out for each individual pixel.
4. For given R, G and B values calculate the greyscale value using the standard formula.
5. 24 bit pixel values are composed using 8-bit greyscale values.
6. At the same location the new value in the output image is stored.
7. The whole image is traversed.

Now, the greyscale component is computed using the following formula:

\[ gs = \frac{r + g + b}{3}; \]

Here the average of R, G, and B is calculated and then saved in the output image.

The alternative formula for calculating the greyscale :  

1. \[ gs = r \times 0.33 + g \times 0.33 + b \times 0.33; \]
2. i.e. 33% of all colors is used to compose final 100% greyscale component.
3. NTSC Formula

1. \[ gs = r \times 0.33 + g \times 0.56 + b \times 0.11; \]
3. The NTSC formula takes 56% of green where as only 11% of blue since green color has more luminescence or brightness than blue.

3.2 Binarization.

3.2.1 Thresholding: one of the simplest method of image segmentation is thresholding technique. The binary image i.e. black and white from a greyscale image is created using thresholding technique. We can set the threshold value manually for manual thresholding whereas the algorithm decides the threshold value in case of otsu’s and local thresholding. In the manual thresholding in the case of multiple image as an input we can change the threshold value for the particular image at the time of processing[3].

Algorithm:

1. Pixel by pixel the entire input image is traverse.
2. Decide the threshold value for the image.
3. The binary output pixel value (24 bit) is calculated based on current Threshold.
4. At the same location the new calculated value in the output image is stored.

3.2.1 Local Thresholding

The local thresholding is based on the windowing technique. Here the windows are considered for deciding the threshold value of the 3*3, 5*5 etc window. The threshold value of the window is calculated using otsu’s thresholding. The entire image is traversed window by window and for each window the entire procedure is repeated.

3.2.3 Otsu’s Thresholding

The threshold value is calculated using otsu’s thresholding algorithm. Here the value is calculated by considering the whole image. The entire image is traversed and the threshold value calculated is used for thresholding. The threshold value is calculated using the standard formula.

\[
\begin{align*}
\text{sum} & += i \times \text{hist}[i] \\
\text{wB} & += \text{hist}[i] \\
\text{wF} & = \text{total} - \text{wB} \\
\text{sumB} & += (i \times \text{hist}[i]) \\
\text{mB} & = \text{sumB} / \text{wB} \\
\text{mF} & = (\text{sum} - \text{sumB}) / \text{wF} \\
\text{varianceBetween} & = \text{wB} \times \text{wF} \times (\text{mB} - \text{mF}) \times (\text{mB} - \text{mF}) \\
\end{align*}
\]

if (varianceBetween > maxVariation)

\{
  maxVariation = varianceBetween;
  maxThreshold = i;
\}

if (gs < maxThreshold)

\{
  gs = 0;
\}

else

\{
  gs = 0xffffff;
\}
Here, \( i \) is the index value which is from 0-256. \( w_B \) and \( w_F \) are the weight background and weight foreground respectively. \( m_B \) and \( m_F \) are mean background and mean foreground. \( \text{maxThreshold} \) is the threshold value for the whole image.

### 3.3 Post-Processing.

#### Blurring:

Each pixel from the source image gets mixed into the neighbourhood pixel i.e. surrounding pixel in the blurring technique. The blurring technique is used to remove the extra noise remaining after the thresholding technique is applied.

**Algorithm.**

1) Pixel by pixel traverse through entire input image.
2) The color values for each pixels are read (24-bit).
3) R, G and B 8bit values are found out for each individual pixel.
4) Calculate the RGB average of surrounding pixels and assign this average value to it.
5) Store the new value at same at same location in output image.

\[
R_{\text{sum}} = R_{\text{sum}} + R \\
G_{\text{sum}} = G_{\text{sum}} + G \\
B_{\text{sum}} = B_{\text{sum}} + B \\
R = R_{\text{sum}}/(\text{depend on type of window used}) \\
G = G_{\text{sum}}/(\text{depend on type of window used}) \\
B = B_{\text{sum}}/(\text{depend on type of window used})
\]

### 3.4 OCR Pre-processing.

#### 3.4.1 Thinning:

The skeletonization of character in the OCR techniques done using thinning. The skeleton of an object is extracted as a result using this technique. The edge pixels having at least one adjacent background point are deleted in every iteration. If its removal doesn’t effect the object all those pixels can be removed. In small number of pixels it represents the shape of the object.

It uses a set of four 3 *3 templates to scan the image[10].

![Figure 1: shows these four templates.](image)

**Figure 1**. Templates to identify pixels to be eroded in the Stentiford Method. The empty white boxes belong to places where the color of the pixel does not need to be checked.

The Stentiford Algorithm can be stated as following[10]:

1) Find such a pixel location \( (i, j) \) that matches with the above template T1. Pixels are removed moving from left to right and from bottom to top of the image with this template.
2) Mark the pixel for deletion, if the central pixel is not an endpoint, and has connectivity number=1.

#### Endpoint pixel:

If the given pixel is connected to just one other pixel then given pixel is endpoint. That is, out of eight possible neighbour the black pixel have only one black neighbour.

#### Connectivity number:

It is a count about how many objects are connected with a particular pixel.

\[
C_n = \sum_{k \in S} N_k - (N_{k} \cdot N_{k+1} \cdot N_{k+2})
\]

where:

- \( N_k \) is the color value of the eight neighbours of the pixel \( i \) analysed. \( N_0 \) is the central pixel. \( N_1 \) is the color value of the pixel to the right of the central pixel and the rest are numbered in counterclockwise order around the center.

\( S = \{1,3,5,7\} \)

**Figure** illustrates the connectivity number.

3) Repeat steps 1 and 2 for all pixel locations matching T1.
4) Repeat steps 1-3 for the rest of the templates: T2, T3, and T4. T2 will match pixels on the left side of the object, moving from bottom to top and from left to right. T3 will select pixels along the bottom of the image and move from right to left and from bottom to top. T4 locates pixels on the right side of the object, moving from top to bottom and right to left.

5) Set to white the pixels marked for deletion.

#### 3.4.2 Segmentation and Scaling:

The segment of characters i.e. a single character from the word is a output we get after segmentation. The Raster scale is the algorithm used for segmentation. Here the whole text image is divided into horizontal and vertical strips known as “scan line”.

There can be several lines of text in the given input document that needs to be categorized into single character for recognition. For this purpose the following steps are to be applied[4]:

1) The scanned line containing continuous black pixels are considered as top of row.
2) Now for bottom the next black line is detected. The area between this is row of characters in image.

![Serialization is the process of converting the data Objects into streams of](image)

**Figure 2**: Row Detection
3) From the row each and every character is segmented. This is done by scanning the row from top to bottom and left to right. When the first black pixel is found it is stored in an array as in the location of the character.

4) Then the array of character is updated as soon as we get the continues black pixels. This is how we will get array of locations of pixel and hence using the location the character is segmented.

5) The character segmented is then scaled to the from any pixel size to 15 X 15 pixel.

3.5 Template Generation:
When a document is put to visual recognition, it is expected to be consisting of printed (or handwritten) characters pertaining to one or more scripts or fonts. This document however, may contain information besides optical characters alone. Now the cropped image of 15 X 15 can be digitalized into array of 15 X 15, where black representing 1 and white representing 0 as shown in figure[4].

Template Matching:
The generated template are stored using the serialization database technique. The template are matched pixel by pixel. Each character is compared with every template stored in the master list. The template with highest matched pixel are considered to be recognized character. The template matching is highly dependent upon the score of matched pixel. If the templates are wrongly matched then we can dynamically train the database by adding the actual template. The database is trained with different fonts. If the user loads the document image of a particular font and if the user loads the database of that particular font then the output of that process will be more efficient or we can say that we will get maximum accuracy.

4. Architecture:
This system is implemented using client server architecture. Client can load multiple images for the binarization. The further processing is done on server side. The basic image processing algorithms are applied for binarization like greyscale, thresholding and blurring. The server will also perform the OCR with algorithms like thinning, segmentation, scaling,
template generation and template matching. The client can train the character database dynamically using different fonts. The deletion and updation of the character master list can be done. The output of OCR is stored in the text format.

5. Result

<table>
<thead>
<tr>
<th>SL. NO</th>
<th>Fonts</th>
<th>Actual Input</th>
<th>Total Characters Recognized</th>
<th>Correct Characters Recognized</th>
<th>Characters Not Recognized</th>
<th>Precision</th>
<th>Recall</th>
<th>Accuracy percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cambria(True)</td>
<td>100</td>
<td>97</td>
<td>03</td>
<td>0.97</td>
<td>0.97</td>
<td>0.90</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Palatino(False)</td>
<td>100</td>
<td>83</td>
<td>17</td>
<td>0.83</td>
<td>0.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Rockwell(True)</td>
<td>100</td>
<td>94</td>
<td>06</td>
<td>0.94</td>
<td>0.94</td>
<td>0.94</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rockwell(False)</td>
<td>100</td>
<td>75</td>
<td>25</td>
<td>0.75</td>
<td>0.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Cambria(True)</td>
<td>100</td>
<td>94</td>
<td>06</td>
<td>0.94</td>
<td>0.94</td>
<td>0.94</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Calibri(False)</td>
<td>100</td>
<td>75</td>
<td>25</td>
<td>0.75</td>
<td>0.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Calibri(True)</td>
<td>100</td>
<td>83</td>
<td>07</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Calibrielight(False)</td>
<td>100</td>
<td>72</td>
<td>27</td>
<td>0.73</td>
<td>0.73</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Palatino(True)</td>
<td>100</td>
<td>87</td>
<td>13</td>
<td>0.87</td>
<td>0.87</td>
<td>0.85</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rockwell(False)</td>
<td>100</td>
<td>84</td>
<td>16</td>
<td>0.84</td>
<td>0.84</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The system is trained with different datasets. For example refer the table result. Here, The contents of the input image given to the system are of cambria font which contains 100 characters. Then, the cambria dataset which is already trained is loaded. Hence, it gives more efficient output like 97 characters of the 100 characters are correctly recognized. And with the same dataset loaded the input image which contains 100 characters of palatino font. Thus, it gives output like 83 characters out of 100 characters are correctly recognized. If we want the more accurate output for palatino then we can train the dataset with the palatino font. Here, in the table true represent the dataset loaded and the contents of the input image are of same font and vice versa.

6. Result Graph

The following graph shows the result of the system.

![Result Graph](image)

7. Conclusion

In this system we have implemented 3 types of thresholding which are manual thresholding, otsu’s thresholding and local thresholding. We can use any type of thresholding depending upon the noise intensity in the document. Finally, once the initial binarization result is derived, the result can be further improved by post processing. The OCR technique is applied to extract the contents of the document images in text format. The system is trained with different fonts. Here we are calculating precision, recall and accuracy for different font. The accuracy of the OCR technique depends upon the type of the dataset loaded during processing. If the dataset loaded is of the same font as that of the image content then we will get more efficient result and vice versa. The proposed method has been tested on the various datasets. Experiments show that the proposed method outperforms most reported OCR methods in term of the Precision, Recall and Accuracy.

8. Acknowledgment

We take this opportunity to express our profound gratitude and deep regards to our guide Prof. Mr. D. D. Pukale for his valuable guidance, monitoring and constant encouragement throughout the course of this work. The blessing, help and guidance given by him time to time made this work successful.

9. Reference


[3] Poonam R. Deokar, Dipali A. Badade, Deepali B. Chavan, Sharadchandra Pawar College of Engineering, University of Pune, India “REAL TIME OPTICAL CHARACTER RECOGNITION BASED ON FEED FORWARD NETWORKS”


[6] Khurram Khurshid1, Imran Siddiqi1, Claudie Faure2, Nicole Vincent1 “COMPARISON OF NIBLACK INSPIRED BINARIZATION METHODS FOR ANCIENT” Laboratoire CRIP5 – SIP, Université Paris Descartes 45, rue des Saints-Pères, 75006 Paris France


[9] Sameeksha Barve, computer science department, Jawaharlal institute of technology borawan, khargone (mp) “OPTICAL CHARACTER RECOGNITION USING ARTIFICIAL NEURAL NETWORK” International journal of advanced technology & engineering research (ijater) issn no: 2250-3153 volume 2, issue 2, may 2012 139

[10] Alberto Martin and Sabri Tosunoglu Florida International University Department of Mechanical Engineering 10555 West Flagler Street Miami, Florida 33174 “IMAGE PROCESSING TECHNIQUES FOR MACHINE VISION”


Authors

First Author – Prof D.D. Pukale, Department of Computer engineering ,Bharati Vidyapeeth’s college of Engineering for Women, Pune, Maharashtra, INDIA, Email: pukaledd@yahoo.com

Second Author – Miss. Aparna Thorat, Department of Computer engineering ,Bharati Vidyapeeth’s college of Engineering for Women, Pune, Maharashtra,INDIA, Email:aparnathorat5@gmail.com

Third Author – Miss. Pooja Bora, Department of Computer engineering ,Bharati Vidyapeeth’s college of Engineering for Women, Pune, Maharashtra, INDIA, Email: bpooj07@gmail.com

Fourth Author – Miss. Shruti Barmecha, Department of Computer engineering ,Bharati Vidyapeeth’s college of Engineering for Women, Pune, Maharashtra,INDIA Email:shrutibarmecha.94@gmail.com

Fifth Author – Miss. Dipika Suryawanshi, Department of Computer engineering ,Bharati Vidyapeeth’s college of Engineering for Women, Pune, Maharashtra, INDIA, Email: dipika3695@gmail.com