Jurnal Elektronika dan Telekomunikasi (JET), Vol. 17, No. 2, December 2017, pp. 56-62
Accredited by RISTEKDIKTI, Decree No: **32a/E/KPT/2017**
doi: 10.14203/jet.v17.56-62

Latin Letters Recognition Using Optical Character Recognition to Convert Printed Media Into Digital Format

Rio Anugrah, and Ketut Bayu Yogha Bintoro

Jurusan Teknik informatika
Universitas Trilogi
Jl. TMP Kalibata No. 1 - 12760
Jakarta, Indonesia

Abstract

Printed media is still popular in nowadays society. Unfortunately, such media encountered several drawbacks. For example, this type of media consumes large storage that impact in high maintenance cost. To keep printed information more efficient and long-lasting, people usually convert it into digital format. In this paper, we built Optical Character Recognition (OCR) system to enable automatic conversion the image containing the sentence in Latin characters into digital text-shaped information. This system consists of several interrelated stages including preprocessing, segmentation, feature extraction, classifier, model and recognition. In preprocessing, the median filter is used to clarify the image from noise and the Otsu's function is used to binarize the image. It followed by character segmentation using connected component labeling. Artificial neural network (ANN) is used for feature extraction to recognize the character. The result shows that this system enable to recognize the characters in the image whose success rate is influenced by the training of the system.

Keywords: Optical Character Recognition (OCR), segmentation, feature extraction, artificial neural network (ANN).

I. INTRODUCTION

In today's work environment where shared work space is a common thing, most people prefer to store information digitally. Printed documents take up spaces. Moreover, with digital format, information will be easier to be accessed and retrieved. Print-based media such as books, newspapers and written sheets can be converted into digitally-based information manually by retyping with word processing applications such as notepad and microsoft word. However, when the typed books have thousands of words and hundreds of pages then typing will take a long time and much effort. To minimize errors and optimize the conversion time of printed media to a digital format, it is essential to have tools to facilitate the conversion of printed media information.

Technology makes it easy to turn paper documents into digital files. Optical character recognition (OCR) is one of the tools that enables us to convert printed characters into digital text. By using OCR, retyping process is no more needed. Some researches have developed OCR system. In [1], OCR system was designed but had no noise filter on the image. As for [2], the proposed OCR system still lack of thinning process which supposed to amplifies the thickness of the character up to one pixel thick. This leads to performance degradation. In [3], OCR system had been built without segmentation process. Segmentation process plays important part in OCR to separate each character in the picture and to distinguish one character with another.

This research, we have built OCR system to digitize printed Latin letters automatically. To enhance the performance of all works mentioned before, we apply the thinning and segmentation process. In addition, we also perform median filter to refine the noise and create better performance.

II. METHOD

The proposed OCR system consists of several steps as shown in Figure 1 [4]. These steps can be categorized into three main stages including preprocessing, segmentation, and recognition.

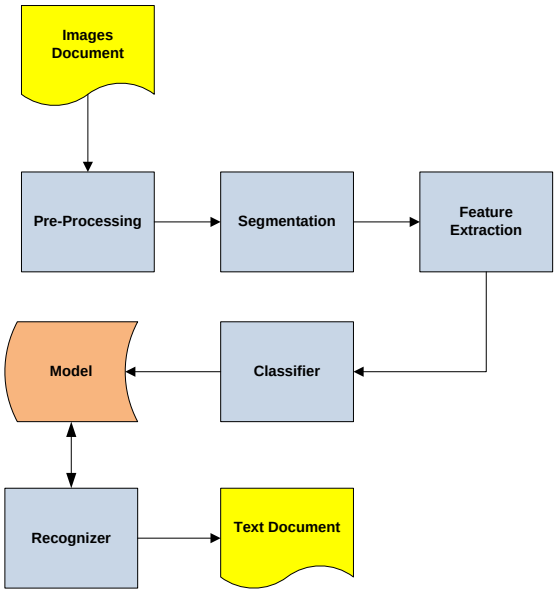


Figure 1. OCR architecture

* Corresponding Author.
Email: rioanugrah@trilogi.ac.id
Received: August 8, 2017 ; Revised: December 17, 2017
Accepted: December 29, 2017 ; Published: December 30, 2017

A. Preprocessing

Preprocessing aims to improve image quality by suppressing the noise and enhancing specific image features. There are two techniques in this stages i.e. noise reduction and binarization. Converting images from analog to digital will add salt and pepper noise effects on them, causing noise to mask objects in images [5]. This noise greatly affects the results of recognition process in OCR system [6]. Median filter is a common and effective method to reduce salt and pepper type of noise in image processing.

Binarization makes it easy to recognize the image since binarization can separate the pixel of writing with the background pixels. The text pixels will be given a white color with a value of "1" and background pixels are black with a value of "0" [7]. Otsu's method is used to binarize a grayscale input image.

- Otsu's Formula

The gray image pixel value is graded by its gray level *L[1,2,3....L]*. The number of pixel level i expressed with n_i and the total pixel is expressed by $N = n_1 + n_2 \ldots.+ n_L$ [8]. Simplify pixels by dividing them into two classes Cb (background) and Cf (foreground) with threshold at level *t*. The Cb pixel level is expressed as [1, 2,...,*t*], Cf is expressed by [*t*+1, *t*+2,.....,L]. The calculation to find the value of variance for each *t* is indicated by the formula below. For background pixel class Cb:

$$Weight\ W_b = \sum_{i=1}^{t} \frac{n_i}{N} \quad (1)$$

$$Mean\ \mu_b = \frac{\sum_{i=1}^{t} i * n_i}{\sum_{i=1}^{t} n_i} \quad (2)$$

$$Variance\ \sigma_b^2 = \frac{\sum_{t=1}^{t} (i - \mu_b)^2 * n_i}{\sum_{i=1}^{t} n_i} \quad (3)$$

For foreground pixel class Cf:

$$Weight\ W_f = \sum_{i=1}^{t} \frac{n_i}{N} \quad (4)$$

$$Mean\ \mu_f = \frac{\sum_{i=1}^{L} i * n_i}{\sum_{i=1}^{t} n_i} \quad (5)$$

$$Variance\ \sigma_b^2 = \frac{\sum_{i-t+1}^{L} (i - \mu_b)^2 * n_i}{\sum_{i=1}^{t} n_i} \quad (6)$$

$$Within\ class\ variance\ \sigma_w^2 = W_b \sigma_b^2 + W_f \sigma_f^2 \quad (7)$$

B. Segmentation

Segmentation process separates each character contained in the image. The separation process works by examining each pixel in a binary image, then a set of pixels of value 1 will separate from a binary set of values 1 in the same image to facilitate the next process [9]. The segmented character is then selected for narrowing of the character thickening to facilitate analysis in the feature extraction process without leaving characteristics of the characters, otherwise, it can reduce the size of the image data which can speed up further processing [10].

C. Recognition

Artificial Neural Network (ANN) approach is used here for character recognition. Feature extraction and classification are two main processes in character recognition stage. Feature extraction recognizes an object of character based on special characteristics and uniqueness based on the value of statistical information of characters on each pixel, so as to distinguish character traits on other character traits [11].

The classifier specifies the characteristic value and classifies its character traits, then fed to the model for storing attributes in training and refinement of the OCR system [12]. Recognition compares the attribute values of each character stored in the model with the newly acquired attribute values, then adjusted to the weight value of the two attributes, if the result of the large weight calculation can then be used as a predictor of character output.

III. Result and Discussion

A. Preprocessing

In this stage, firstly, the input image goes through grayscale conversion to change input image into gray color with the intensity of 8 bits. This conversion is necessary because if the image continues to use its original color, the computing process becomes heavy [13]. Grayscale generated using the following equation:

$$Gray = 0.2989 * R + 0.5870 * G + 0.1140 * B \quad (8)$$

Figure 2 shows input image and its grayscale convertion. Matlab implementation of grayscale convertion is as follow: I = imread ('TRILOGI2017.png'); Igray = rgb2gray(I);

For noise reduction, median filter chooses the initial pixel value that is not the edge of the image. Then it looks at the pixel value around and sorts the initial and surrounding pixel values ascending or descending. Finally, it grabs the median value from the pixel value sequence and replaces it with initial pixel value. Figure 3 shows set of pixels before and after median filter process. Matlab function for median filter: "imedian = medfilt(Igray)".

(a)

(b)

Figure 2. Input image (a) original image (b) grayscale image

17	23	27
20	**93**	20
20	48	50

17	23	27
20	**23**	20
20	48	50

Figure 3. The left figure is a set of pixels before the median filter is applied, the right figure is the result

The Otsu's method is used in binarization because it distinguishes automatically the object class (foreground) and background without affecting the character trait of the image. The Otsu's method follows these steps [14]:

a. Read a gray scale image.
b. Calculate image histogram.
   The histogram is obtained by counting the number of occurrences of each pixel, then mapped to the intensity value of the grayscale image [15], [16]. Figure 3 shows histogram of grayscale version of input image (Figure 4(b)).
   To calculate the histogram value of the image, use matlab function "imhist(I)".
c. Select a threshold *t*.
d. Calculate variance of the foreground and background classes.
e. Repeat steps c and d until all threshold values are completed.
f. Initialize T as the lowest within-class variance value.
g. Any pixel value whose level is greater or equal to T is the object, otherwise the background.

Matlab implementation of Otsu's method is as follow: "Ibw = im2bw (Igray,graythresh(imedian))".

B. Segmentation

Character segmentation employs connected component labeling algorithm [17] as follow:

a. Scan each image pixel of binarization output and select object pixels, starting with label = 1.
b. See the availability of nearby object pixels, if exist see label values on nearby object pixels.
c. If the nearest object's pixel does not yet have a label value, assign the initial label value to the pixel of the selected object and the value of the label increases by one.

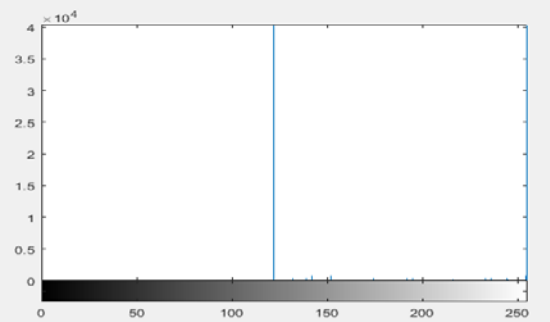
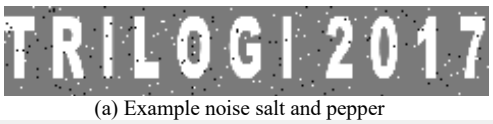


Figure 3. Histogram of grayscale input image (Figure 4(b))



(a) Example noise salt and pepper

(b) The result of noise reduction implementation

(c) Binarization result of image in (b)

Figure 4. Preprocessing images

d. If the pixel of the closest object has a label value, then copy that label's value into the pixel label value of the selected object.
e. Scan for pixels to find different but connected label values if exist compare their label values and then the largest label value are replaced by the smallest label value and do the looping until the values of the each connected labels are same.

Figure 5(k) shows each character has been separated with other character which bounded by the green box for each character. Segmentation function using matlab: "[Ilabel num] = bwlabel(Ibw); disp(num); Iprops = regionprops (Ilabel); Ibox = [Iprops.BoundingBox]; Ibox = reshape (Ibox, [4 11]); imshow (Ibw); hold on; for cnt = 1:11 rectangle('position',Ibox(:,cnt),'edgecolor','g'); end".

- Thinning

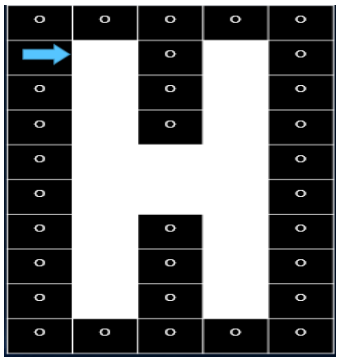
In image processing, thinning is a fundamental step to reduce a thick digital object into a thin skeleton. Zhang-Suen (ZS) is an iterative thinning algorithm which is fast and simple to implemented [18]. The algorithm follows these conditions:

1. $2 \leq N(P1) \leq 6$
2. $S(P1) = 1$
3. $P2 * P4 * P6 = 0$
4. $P4 * P6 * P8 = 0$

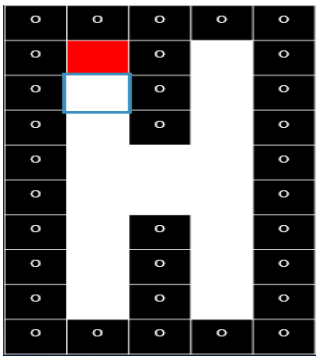
Figure 6 shows $P1$ is the pixel of selected object, $N(P1)$ is the nearest object pixel of $P1$ so the value $N(P1) = 3(P3, P6$ and $P7)$ suits to the first rule. $S(P1)$ is a transition of pixel value 0 to 1 from a comparison of two pixels in a clockwise direction without involving $P1$. It means $S(P1) = 2(P2(0)$ with $P3(1)$ and $P5(0)$ with $P6(1))$ are not in accordance with the second rule. This means that thinning for case in Figure 6 can not be done because it does not meet the condition of the algorithm.

Figure 7(a) follows the first and second rule, because $N(P1) = 3(P4, P5$ and $P6)$ and $S(P1) = 1(P3(0)$ with $P4(1))$. The third rule is the product of three object pixels ($P2$, $P4$ and $P6$) must be 0. Figure 5(k) follows the third rule because $P2 = 0$. The fourth rule is same because $P8 = 0$. If all algorithm rules are met mark the pixel $P1$ then search again to select the pixel of the object.

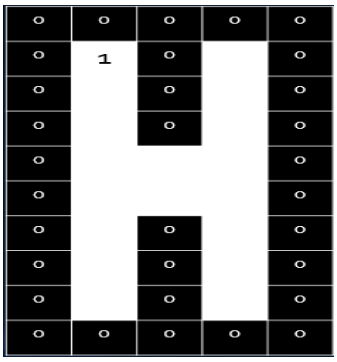
From the described condition, the value of $P1$ of Figure 7(c) is not marked, because the fourth rule is the product of three object pixels must be zero ($P4 * P6 * P8$). Figure 7(c) shows the value of three pixels is 1. After that, do all the rule of the algorithm for each pixels object. If the rules are followed, marks the pixels, and if not, check other pixels object. Lastly, change the pixel value of marked pixel object to 0. Repeat the ZS algorithm condition for each pixel of the object to check the availability of pixels that can be marked until no pixel of the object can be marked and replace the marked value to 0. Figure 7(h) shows the thinning output of image in 7(a). Figure 7(i) is the thinning output image. Each character through thinning still has the features of each character and still restricted and separated from the result of segmentation.



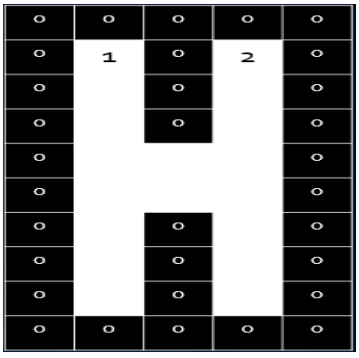
(a) The pixel of the selected object is marked with a blue arrow.



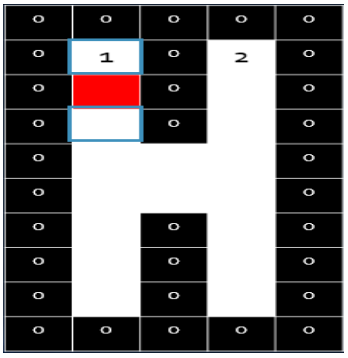
(b) The pixel of the selected object is marked by the blue arrow and the pixel of the object closest to it that does not have the label value marked in blue



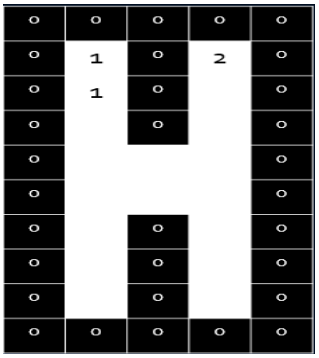
(c) Assign labeling values on selected object pixels.



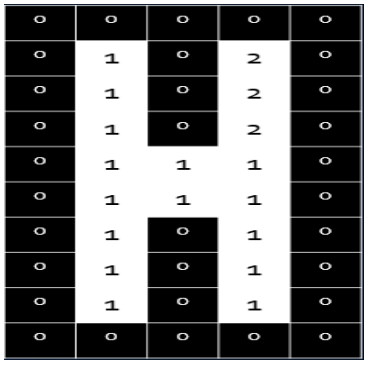
(d) Assign labeling values on the next selected pixels object



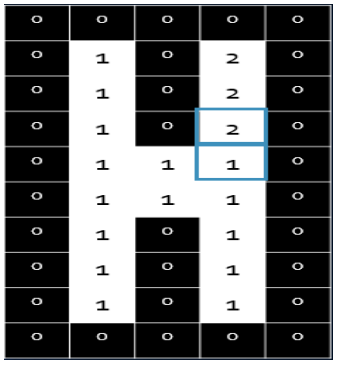
(e) The selected object's pixel is marked in red and the pixel of its nearby object is marked in blue



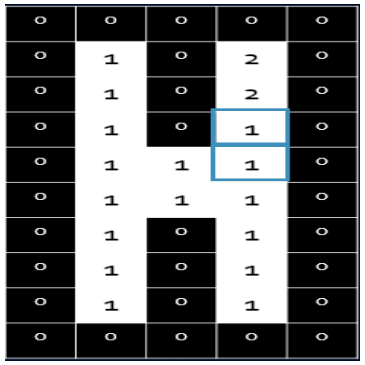
(f) Result of copied label value



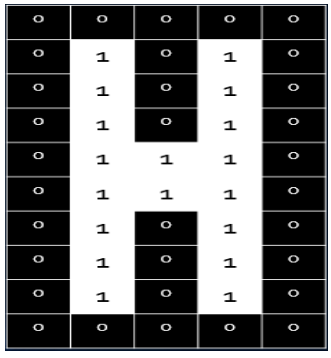
(g) Result of copied label values for each object pixel



(h) Two connected different label values marked in blue



(i) Change label value



(j) Result of segmentation using connected component labeling

(k) Implementation of segmentation with matlab using connected component labeling

Figure 5. Character segmentation using connected component labeling algorithm

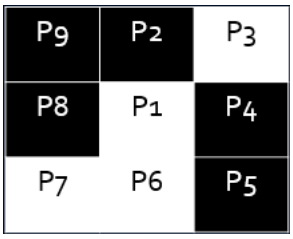


Figure 6. An example of an experimental zhang-suen algorithm for the first and second conditions.

C. Feature Extraction

In this step, the features of character is simplified into a form of 10x10 matrix which matrix value is derived from the value of each character pixel shown in the image [1]. To enhance this stage mfp is used because it can filter out noise resulting in changes in pixel value from image. Example of the "T" characters that have passed the segmentation process shown in Figure 30 has the corresponding pixel value in Figure 8. Change the matrix to be a vector, then the vector of Figure 31 i s VI (vector *input*) = [1110000111001100110000011110000000010000000001000000000010000000001000000000100000000010000000001000000000010000]

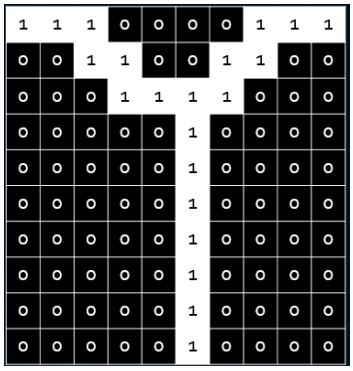
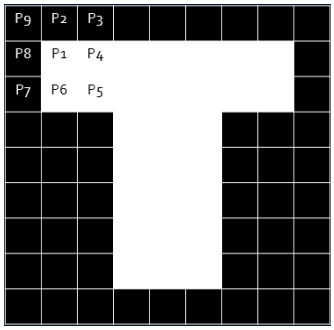
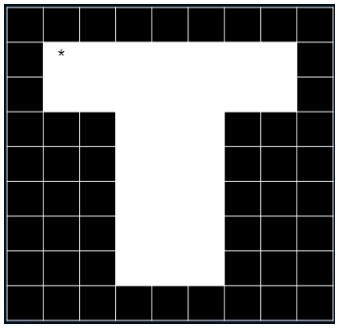


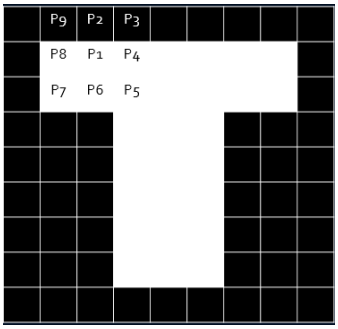
Figure 8. Character extraction in form of matrix



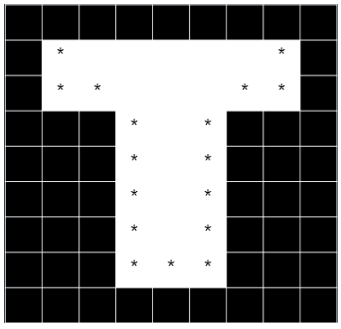
(a) An example of a zhang-suen algorithm



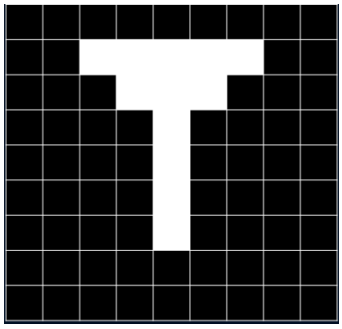
(b) Marking on pixel *P*1



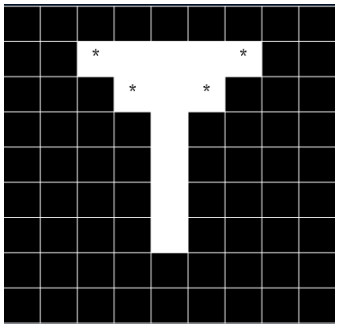
(c) Checking on the next pixel



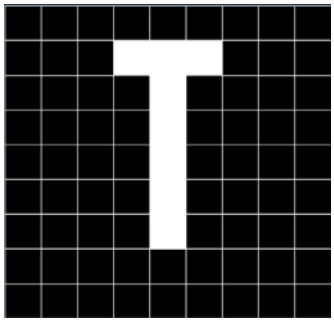
(d) The result of marking all pixels in figure 22 because all zhang-suen algorithm conditions are met.



(e) Change pixel the value of marked pixel object to 0



(g) There are several marked pixels from the results of the re-examination in Figure 26



(h) Changed result of marked pixel value

(i) Result of thinning

(j) The boundary of each thinned characters

Figure 7. Thinning process using Zhang-Suen (ZS) algorithm

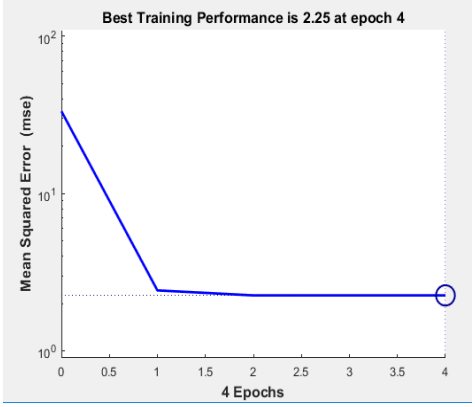


Figure 9. Training OCR using ANN

The obtained vector patterns can be classified and trained then stored in the model or can be tested with available patterns in the model for recognition process. Training in OCR can use artificial neural networks (ANN). The result of the training is shown in Figure 9. It consists of 10 characters, namely "T", "R", "I", "L", "O", "G", "2", "0", "1" and "7". The training was completed in 4 epochs. In the first epoch, almost all of the 10 characters have been studied from the pattern. Second and third epochs experienced significant obstacles in their learning, the fourth epoch the entire character pattern was identified.

D. Recognizer

Characters that have been obtained from the pattern of the previous process are matched with the pattern of characters stored in the model. To understand the pattern matching process see the explanation below[19]:

a. Take value VI.
b. Matches the pattern VI with all values of the model vector pattern (VM) stored to calculate the weighted vector value (VB).
c. If each value of VI and VM is equal, then the value of VB increases by one, and if the value is not equal, then the value of VB is reduced by 1 with the initial VB value is 0.
   For example, 4 vector VI values of Figure 7(e) are VI = 1 1 1 0 and the character "T" in the model has VM = 1111 and VB = 0.
   Vector 1 : VI(0) = 1 and VM(0) = 1. Both values are the same, so VB = 0 + 1 = 1.
   Vector 2 : VI(1) = 1 dan VM(1) = 1. Both values are the same, so VB = 1 + 1 = 2.
   Vector 3 : VI(2) = 1 dan VM(2) = 1. Both values are the same, so VB = 2 + 1 = 3.
   Vector 4 : VI(3) = 0 dan VM(3) = 1. Both values are not the same, so VB = 3 - 1 = 2.
   The above calculation shows that the character "T" has a weight of 2, then VI compared again with each character stored in the model to calculate each weight on the character.
d. Character prediction results by taking the largest weighting value that has been calculated from the model.

The recognition using ANN approach consists of several layers as presented in Figure 10. There are three

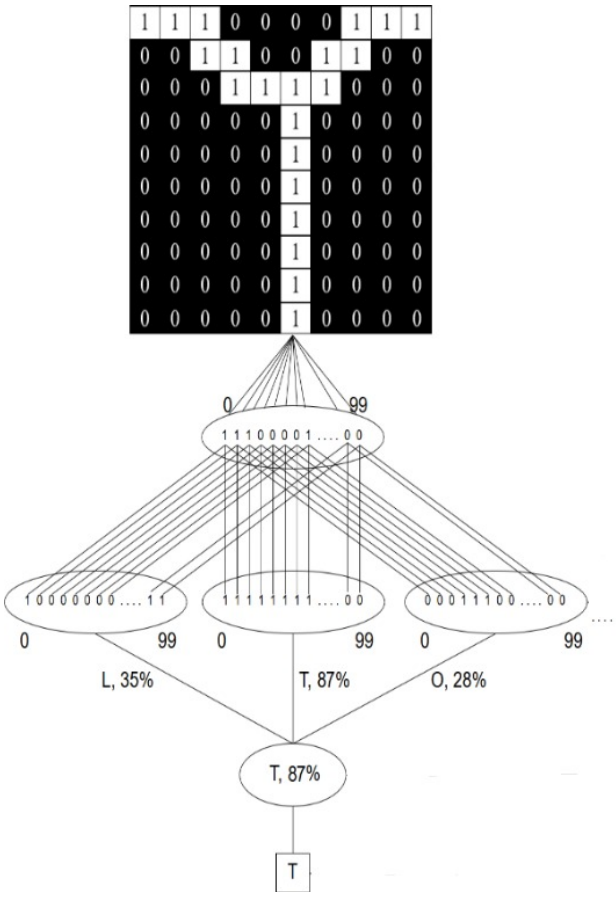


Figure 10. ANN layers of character recognition.

layers including input layer, hidden layer and output layer. The input layer is a v ector value obtained from the character pattern in the image to be recognized. Hidden layer is a p attern matching process obtained from the input layer with the pattern available on the model, for example in the model consisting of 100 character patterns then the input layer will match as many as 100 character objects and each vector value, the last output layer is the result of a character in the model Has been classified by the greatest weight of the result of matching the value of both patterns.

The result of OCR implementation with predictions of characters shows in Figure 11. The output of the character in the form of text is quite appropriate with the input image. The output can be copied and then taped into word processing applications such as notepad or microsoft word to facilitate text-based information transfer from printed media to digital format.

Figure 11. The result of OCR implementaion shows text output

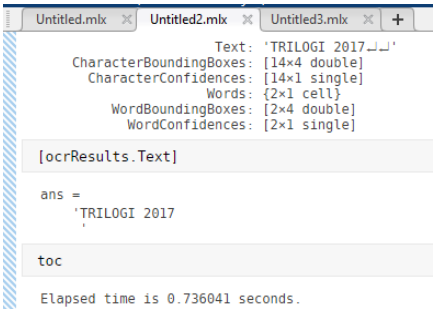


Figure 12. Time taken to recognize words with the original image

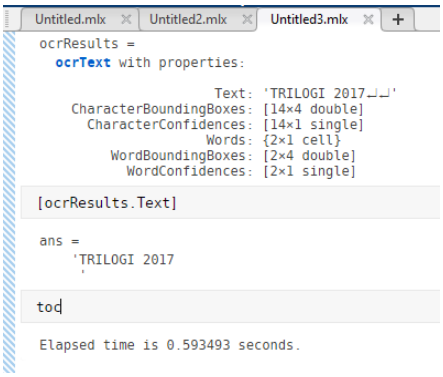


Figure 13. Time taken to recognize words based Figure 7(i)

The final output image in Figure 7(i) that has been through all the steps and algorithms described above has the advantage of faster processing time compared to the initial image without going through the above process. Figures 12 and 13 show the time difference for original and the final result image in figure 7(i) to recognize words.

CONCLUSION

Based on the result we can describe every step of standard OCR system. Standard OCR has noisier in feature extraction so we proposed MFP to refining the noise and create better performance in the outcome. The noise reduction stage improves the image quality by eliminating noise so that the character objects contained in the image become clear. Clearer image quality will reduce training time in ANN not only that it will reduce time to recognize the words so as the result, we could reach better quality output with less time performance to gain it.

REFERENCE

[1] J. Chandarana, and M. Kapadia, "Optical character recognition," *International Journal of Emerging Technology and Advanced Enginnering*, 4(5), pp. 219–223, 2014.

[2] S. Chopra, and A. Ghadge, "Optical Character Recognition," *International Journal of Advanced Research in Computer and Communication Engineering*, 3(1), pp. 4956-4958, 2014.

[3] A. Sharma, and D.R. Chaudhary, "Character Recognition Using Neural Network. *International Journal of Engineering Trends and Technology (IJETT)*, *4*(April), pp. 662–667, Apr. 2013

[4] A.T. Birhanu, and R. Sethuraman, "Artificial Neural Network Approach to the Development of OCR for Real Life Amharic Documents," *International Journal of Science, Engineering and Technology Research*, 4(1), pp. 141–147, 2015.

[5] E.S.A. Ahmed, R. E.A Elatif, Z.T. Alser, "Median Filter Performance Based on Different Window Sizes for Salt and Pepper Noise Removal in Gray and RGB Images," *International Journal of Signal Processing Image Processing and Pattern Recognition*, 8(10), pp. 343-352, 2015.

[6] B.K Pal, P.S. Tiwari, P.S. Kumar, "Efficient Small and Capital Handwritten Character Recognition with Noise Reduction," *International Journal of Emerging Technology and Advanced Engineering*, *3*(8), pp. 408-413, 2013.

[7] N. Chaki, S.H. Shaikh, K. Saeed, "Exploring image binarization techniques," *Studies in Computational Intelligence*, 560, pp. 5–16, 2014.

[8] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, *9*(1), pp. 62-66, 1979.

[9] G. Mehul, P. Ankita, D. Namrata, G. Rahul, S. Sheth, "Text-Based Image Segmentation Methodology," *Procedia Technology*, vol. 14, pp. 465-472, 2014.

[10] D.N. Hakro, S.A. Awan, M. Memon, A. Aamur, G. Mojai, "Interactive thinning for segmentation-based and segmentation-free Sindhi OCR," *Sindh University Research Journal-SURJ (Science Series)*, 47(3), pp. 395-398, 2015.

[11] M.S. Nixon, and A.S Aguado, *Feature Extraction and Image Processing,* Academic Press, 2008.

[12] S. Kumar, "A Brief Review of Classifiers used in OCR Applications," *International Journal of Computer Trends and Technology (IJCTT)*, 34(2), pp. 80-88, 2016.

[13] R. Buse, Z.-Q. Liu, J. Bezdek, "Word Recognition Using Fuzzy Logic," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 1, pp. 65-76, 2002.

[14] N. Bhagava, A. Kumawat, R. Bhargava, "Threshold and binarization for document image analysis using otsu's Algorithm," *International Journal of Computer Trends and Technology (IJCTT)*, 17(5), pp. 272-275, 2014.

[15] S. Imam Syafi, R. Tri Wahyuningrum, D. Arif Muntasa, "Segmentasi Obyek Pada Citra Digital Menggunakan Metode Otsu Thresholding," *Jurnal Informatika*, 13(1), pp. 1-8, 2015.

[16] K. B. Yogha, M. Cendana, R. Lipikorn, "Non-deterministic finite state automata as termites swarm agent model," in *Proc. of 7th International Workshop on Computer Science and Engineering (WCSE)*, pp. 318-325, 2017.

[17] P. Pooja, R. Phalak, W. Jayashri, S. Yugandhara, "Text extraction from English comic images using connected component algorithm," in *Proc. of 4th IRF International Conference*, Pune, pp. 166-169, 2014.

[18] A. S. Karne and S. S. Navalgund, "Implementation of an Image Thinning Algorithm using Verilog and MATLAB," *International Journal of Current Engineering and Technology (Ncwse)*, pp. 333-337, 2013.

[19] Hendri, "Character Recognition Dengan Menggunakan Jaringan Syaraf Tiruan," *Jurnal TIMES*, III(2), pp. 1–5, 2014.