DOI: 10.14421/ijid.2021.2360 IJID (International Journal on Informatics for Development), *e-ISSN: 2549-7448*
Vol. 10, No. 2, 2021, Pp. 62-71

Handwriting Arabic Character Recognition Using Features Combination

Fitriyatul Qomariyah
Faculty of Tarbiyah
IAIN Madura
Pamekasan, Indonesia
fitriyatulqomariyah@iainmadura.ac.id

Fitri Utaminingrum
Faculty of Computer Science
Brawijaya University
Malang, Indonesia
f3_ningrum@ub.ac.id

Muchlas
Faculty of Tarbiyah and Teaching Training
Maulana Malik Ibrahim State Islamic University
Malang, Indonesia
acculaszuby@gmail.com

Article History
Received Feb 20th, 2021
Revised Oct 2th, 2021
Accepted Oct 3th, 2021
Published Oct, 2021

***Abstract*— The recognition of Arabic handwriting is a challenging problem to solve. The similarity among the fonts appears as a problem in the recognition processing. Various styles, shapes, and sizes which are personal and different across individuals make the Arabic handwriting recognition process even harder. In this paper, the data used are Arabic handwritten images with 101 sample characters, each of which is written by 15 different handwritten characters (total sample 101x15) with the same size (81x81 pixels). A well-chosen feature is crucial for making good recognition results. In this study, the researcher proposed a method of new features extraction to recognize Arabic handwriting. The features extraction was done by grabbing the value of similar features among various types of font writing, to be used as a new feature of the font. Then, City Block was used to compare the obtained feature to other features of the sample for classification. The Average accuracy value obtained in this study was up to 82%.**

Keywords — arabic character recognition; handwriting arabic character; OCR; features extraction; features combination

1 INTRODUCTION

Optical Character Recognition (OCR) is an electronic machine used to convert handwriting to text that is readable and editable using the computer[1]. The main function of Optical Character Recognition (OCR) is to resemble a human's reading ability. Thus, the OCR system is a part of artificial intelligence which is also a branch of computer science. The recognition of Arabic handwriting using Optical Character Recognition has been a popular interest among researchers presently[2].

However, the recognition of Arabic handwriting is more difficult and more complex than recognition processing of other characters from other languages[3]. The shapes and sizes of every font in Arabic characters can be different among individuals which is the main challenge of this study. Besides, unlimited variations of styles in Arabic handwriting that influence the accuracy of the recognition processing[4], several methods have been deployed in the recognition of Arabic handwriting[5].

One of the methods is by using features extraction [6]. The selection of appropriate features influences the recognition processing especially, the recognition of Arabic handwriting. Various Arabic fonts that have intense similarity among each other and a variety of writing styles across individuals are complex problems in determining the best feature to be used for the recognition processing to get the most salutary result[7].

Several studies have been conducted regarding this issue. Lawgali and Bouridane have conducted a study to recognize Arabic handwriting using Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT) extraction features [8]. The accuracy of using DCT was 59.81% for DWT on a figure of 64x64. M.A. Abdullah and H.H Al-Fraidi et.al also administered a study to recognize Arabic handwriting [9]. In their study, the recognition process was conducted in several phases; extracting the shapes and features of the fonts. Using this method, the accuracy was 81%. Another study was also done by Sahlol and Suen who used integral projection (vertical and horizontal projection) and some other features with an accuracy score of 88% [10]. But this research is not good for the recognition of Arabic characters.

In this study, a new approach of utilizing the form of images method was proposed. This approach was constructed by integrating similar features of a character from a different person's handwriting to make a new feature. The new feature was then used as a feature for certain characters. There were 101 different shapes of characters from 15 different handwriting of each of the characters. In summary, there were a total of 1.515 samples used in this study. A detailed explanation is appended below in Fig. 1.

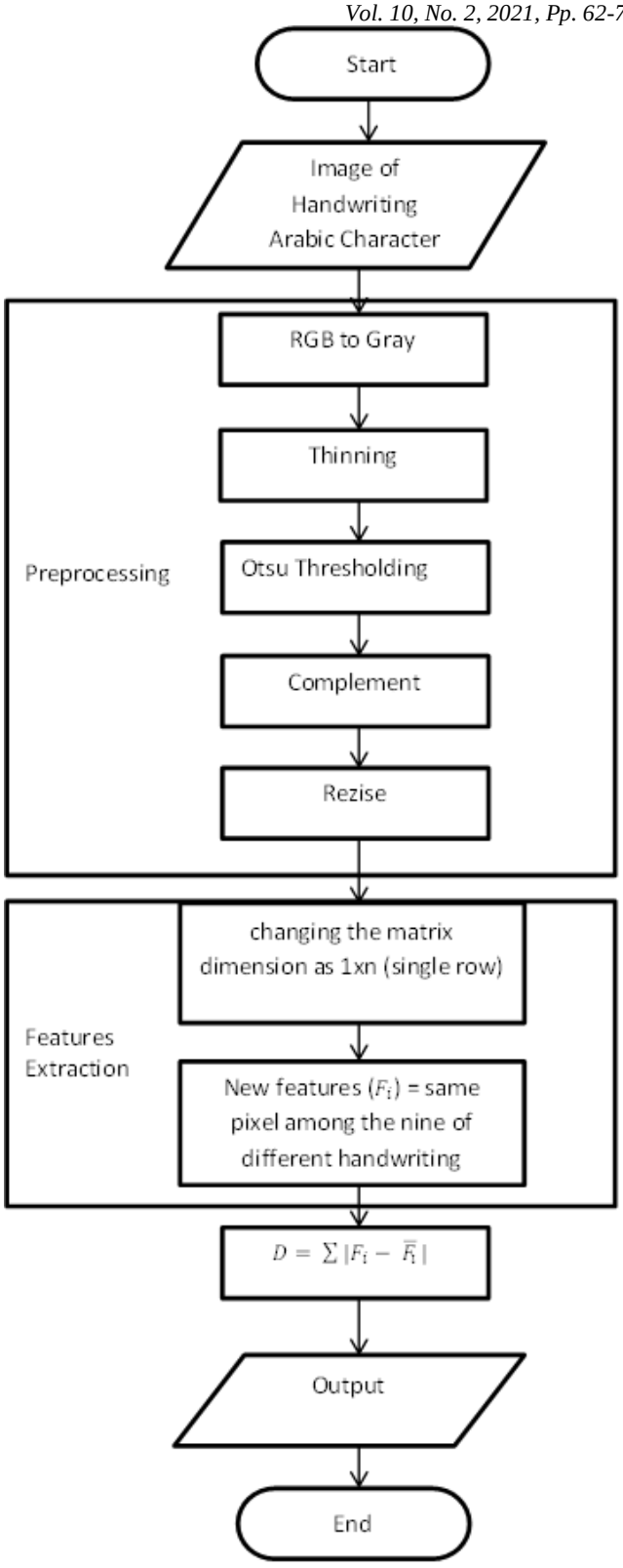


Figure 1. Design system

The details of the recognition process will be explained in the second part. Whilst, for the overcome and conclusion will be explained in the next part.

2 METHOD

Preprocessing is the initial process of improving an image to remove noise. As stated by Bahri and Maliki, preprocessing is a process to remove parts that are not needed in the input image for further processing [11]. preprocessing in this study through several processes: convert the element of RGB (red, green, and blue) to grayscale, differentiate the foreground from the background using a threshold value, the change value 0 is turned to 1 and vise versa, thin out the image pixel and resize the image sizes to 81x81 pixels. Preprocessing is to facilitate researchers to identify each handwritten Arabic character.

2.1. Arabic Characters

The Arabic character is also called the Hijaiyyah letter which consists of 28 letters, (29 if hamzah is considered to be a separate letter). Alif, waw, and ya' are used as long vowels (such as aa, ii, uu), Translation results as a diphthong or a double sound (such as ai, au), and also as a weak consonant.[12]

The Arabic alphabet is written from right to left. Each letter has several different forms according to the position of the letter in a word. There are 84 different shapes according to the position of the letter (beginning, middle, end, and independent).[10]

Additional components that are connected to the body of Arabic letters are crucial pieces of information such as additional components of dots [13]. Note that, same shapes with different additional components produce different letters, see Fig. 2.

No.	Character	Start	Middle	End	Isolated	No. of shapes
1	Alef	-	ـا, ـأ, ـإ	ـى	ا, أ, إ, آ, ى	9
2	Ba	بـ	ـبـ	ـب	ب	4
3	Ta	تـ	ـتـ	ـتـ, ـت	ة, ت	6
4	Tha	ثـ	ـثـ	ـث	ث	4
5	Geem	جـ	ـجـ	ـج	ج	4
6	Ha'a	حـ	ـحـ	ـح	ح	4
7	Kh'a	خـ	ـخـ	ـخ	خ	4
8	Dal	-	-	ـد	د	2
9	Dhal	-	-	ـذ	ذ	2
10	Ra	-	-	ـر	ر	2
11	Zeen	-	-	ـز	ز	2
12	Seen	سـ	ـسـ	ـس	س	4
13	Sheen	شـ	ـشـ	ـش	ش	4
14	Sad	صـ	ـصـ	ـص	ص	4
15	Dad	ضـ	ـضـ	ـض	ض	4
16	Tah	طـ	ـطـ	ـط	ط	4
17	Dhad	ظـ	ـظـ	ـظ	ظ	4
18	Aen	عـ	ـعـ	ـع	ع	4
19	Ghen	غـ	ـغـ	ـغ	غ	4
20	Fa	فـ	ـفـ	ـف	ف	4
21	Qaf	قـ	ـقـ	ـق	ق	4
22	Kaf	كـ	ـكـ	ـك	ك	4
23	Lam	لـ, لا, لأ, لإ, لآ	ـلـ	ـل, ـلا, ـلأ, ـلإ, ـلآ	ل, لا, لأ, لإ, لآ	8
24	Meem	مـ	ـمـ	ـم	م	4
25	Noon	نـ	ـنـ	ـن	ن	4
26	Ha	هـ	ـهـ	ـه	ه	4
27	Waw	-	-	ـو	و	2
28	Ya	يـ	ـيـ	ـي	ي	4
				Total number of shapes (templates)		113

Figure 2. Arabic Characters

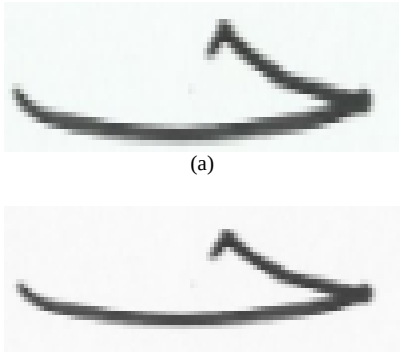
2.2. RGB to Gray

The RGB image is an image with each pixel consisting of three colors (red, green, and blue)[14]. The RGB color model is a color model based on the concept of adding strong primary light, namely red, green and blue. In a room where there is absolutely no light, the room is completely dark. No lightwave signal is absorbed by our eyes or RGB (0, 0, 0)[15]. If we add red light to the room, then the room will change color to red. For instance, in RGB (255, 0, 0) all objects in the room will only be seen in red when our light is replaced with green or blue. As we know, the RGB is a coloring system for digital appearance and is widely used for computer monitors, videos, cellphone screens, etc. The RGB color system consists of 100% Red, 100% Green and 100% Blue which results in 100% white. There is no black in RGB[16].

A grayscale image is a monochrome image consisting of brightness information only [17]. RGB to Gray is a process to convert the element of RGB to grayscale, see Fig. 3. Luminance is a standard algorithm of image processing. It is exercised by MATLAB with "rgb2gray" function which is often used in computer vision. Luminance [18] is designed to match the human's perception of brightness through the weight combination of RGB by the following Equation 1

$$G_{Luminance} = 0.2989 * R + 0.5870 * G + 0.1140 * B \quad (1)$$

In which R is the red element, G is the green element, and B is the blue element.



(a)

(b)

Figure 3. RBG Image (a) and Gray image (b)

2.3. Otsu Thresholding

Thresholding is the simplest method of image segmentation. From the grayscale image, thresholding can be used to form a binary image. A binary image is a digital image that has only two possible values for each pixel. The two colors are black and white or 0 and 1.[19]

Otsu thresholding is a simple segmentation method in segmentation techniques, so it can be easier to divide homogeneous areas based on similarity criteria to identify objects. Otsu Thresholding is a method that differentiates the foreground from the background using a threshold value.

However, poor brightness and different background colors are the traits to determine the threshold value. For instance, it is assumed that C is an original image that has bright (C_1) and dark (C_2) sides, while $P(s)$ is the probability of every side which can be formulated as follows this Equation 2 and Equation 3 [20]:

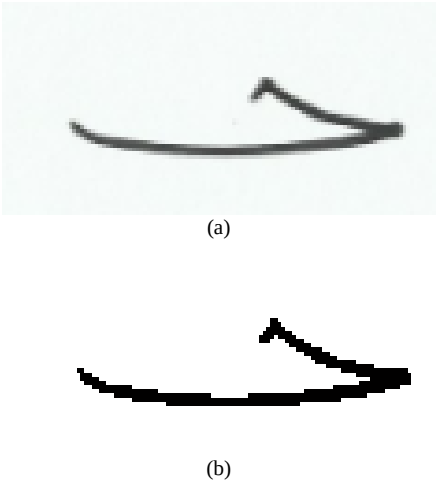
$$P_1(s) = \sum_{i=0}^{s} P_i \quad for \quad C_1[s > 0] \tag{2}$$

$$P_2(s) = \sum_{i=s+1}^{L-1} P_i = 1 - P_1(s) \quad for \quad C_2[s+1, L-1] \tag{3}$$

P_1(s) and P_2(s) are probabilities of the dark and the bright side of C_1 and C_2, whilst P_i is the normal histogram of the overall size of the image. To determine the threshold value k, bimodal histogram, statistics stationery, hence, adaptive and uniform illumination area can be modified to obtain the threshold value using this Equation 4:

$$k = \frac{\sigma^2 B}{\sigma^2 G} \tag{4}$$

In which k is threshold value, $\sigma^2 B$ is the global variant of the overall picture, while $\sigma^2 G$ is the variant across classes. Fig. 4 describes the result of converting RGB image to binary image by using otsu.



(a)

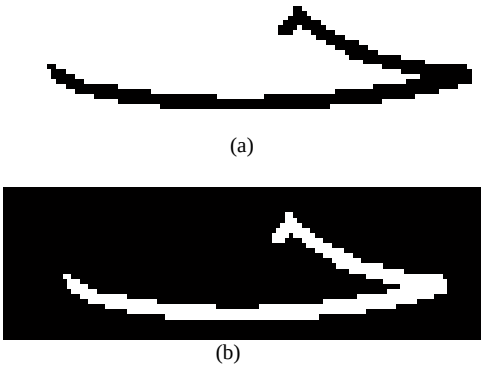
(b)

Figure 4. RGB Image (a) and Binary image (b)

1	1	1
0	0	1
0	1	1

0	0	0
1	1	0
1	0	0

Figure 5. Example of Complement



(a)

(b)

Figure 6. Binary Image (a) and Complement image (b)

2.4. Complement

In complementary grayscale or color images, each pixel value is subtracted from the maximum pixel value supported by the class (or 1.0 for double-precision images). The difference in pixel values is used as the image output pixel value. In the output image, light areas become darker and dark areas become lighter. For color images, green becomes magenta, blue becomes yellow, red becomes cyan, and vice versa [21]. In this process, the pixel value 0 is turned to 1 and vice versa. See the example in Fig. 5 and Fig. 6.

2.5. Thinning

Thinning algorithm is an algorithm that slenderizes the pixel of an image[22]. Thinning is an operation of image processing that operates on a binary image by scraping parts of the object into a line that represents the framework of the object, it can be seen in Fig. 8. This algorithm works by doing reduplication of the pixel layer until there is no more pixel to delete[23]. This method performs pixel removal. Usually, the checking process is done twice, and if there is no more change, then the reduplication stops. Algorithm ZS is a thinning algorithm proposed by Zhang and Suen in 1984 that uses a 3x3 system as shown in Fig. 7 [24].

P8	P2	P3
P7	P1	P4
P6	P1	P5

Figure 7. ZS 3x3 neighborhood

There are two sub-iterations in this algorithm: to remove the restraint pixel of South-East and the pixel point of North-West. The second is to remove the restraint pixel of North-West and the pixel point of South-East.

At first sub-iteration (odd iteration), p_1 is removed from the pattern if p_1 is following the conditions which can be written by Equation 5, Equation 6, Equation 7 and Equation 8:

$$2 \leq B(p_1) \leq 6 \quad (5)$$
$$A(p_1) = 1 \quad (6)$$
$$p_2 \times p_4 \times p_6 = 0 \quad (7)$$
$$p_4 \times p_6 \times p_8 = 0 \quad (8)$$

In the second sub-iteration (even iteration), p_1 is removed from the pattern if p_1 is following the conditions that can put in writing in Equation 9, Equation 10, Equation 11 and Equation 12:

$$2 \leq B(p_1) \leq 6 \quad (9)$$
$$A(p_1) = 1 \quad (10)$$
$$p_2 \times p_4 \times p_6 = 0 \quad (11)$$
$$p_4 \times p_6 \times p_8 = 0 \quad (12)$$

Where A (p_1) is the number pairs of 0-1 (white-black) in transverse clockwise from 8 neighborhood in (p_1) like (p_2), (p_3), (p_4) ,..., (p_8), (p_9). B (p_1) is the number of which not zero in 8 neighbors in the following Equation 13:

$$B(p_1) = \sum_{i=2}^{9} P_i \quad (13)$$

If no conditions are matched, p_1 is not removed from the foreground.

(a)

(b)

Figure 8. Binary Image (a) and image after the thinning process (b)

2.6. Resize Image

Image resizing means changing the size of the digital image in pixels. There are times when the size changes to be smaller or larger. When the size is small or large, it then gets to be done proportionally both on the length and width of the image. In this study, we resize the image sizes to 81x81 pixels. This process is done to equalize all the image sizes so that the feature extraction process can be effective. Fig. 9, Fig. 10. Fig. 11, and Fig. 12 are example image with different Arabic letters.

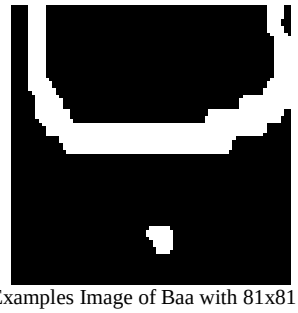


Figure 9. Examples Image of Baa with 81x81 pixels

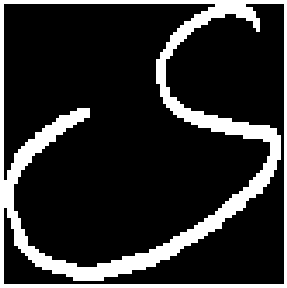


Figure 10. Examples Image of alif with 81x81 pixels

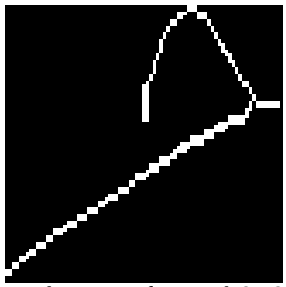


Figure 11. Examples Image of Haa with 81x81 pixels

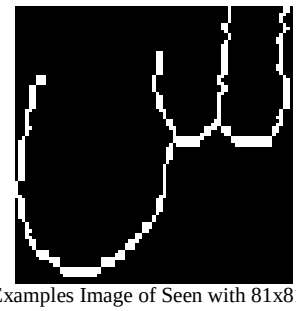
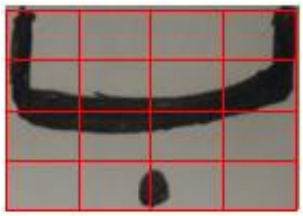


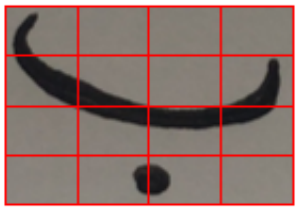
Figure 12. Examples Image of Seen with 81x81 pixels

Arabic characters are quite different from other characters of other languages. In the recognition processing of Arabic handwriting in computer vision, similar features of some

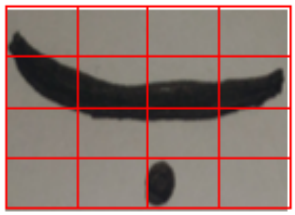
characters are the traits to solve. In addition to this complex problem, another challenging trait is also triggered by various styles, shapes, and sizes of handwriting among individuals. Therefore, the selection of the feature is the key to the success of recognition processing. In this study, a new features extraction was proposed to be used in Arabic handwriting recognition. This study involved 101 samples of characters which each was written by 15 different handwriting characters (the total of the sample was 101x15) with the same image sizes of 81x81 pixels as shown in Table 1. Firstly, data of 15x30 Arabic handwriting images were coded and changed to binary codes (consisting of 0 and 1) where foreground means 1 and background means 0 as shown in Fig. 5 and Fig. 6. 15 of different writing styles of a character were used and analyzed. Then, features of the data were extracted by changing the matrix dimension, see Fig. 14 which illustrates pixel value from Fig. 13 in m x n, then converted to 1xn as shown in Fig. 15. After that, the next step was to find the same pixel among the nine different handwriting to be used as the new feature of the character as illustrated in Fig. 16 in which the pixel in pink shows a similar pixel value.



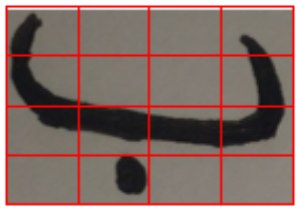
(a)



(b)



(c)



(d)

Figure 13. Example of Baa

1	0	0	1
1	1	1	1
1	1	1	1
0	1	1	0

(a)

1	0	0	0
1	1	0	1
0	1	1	1
0	1	1	0

(b)

1	0	0	0
1	1	1	1
0	1	1	1
0	0	1	0

(c)

1	0	0	1
1	1	0	1
1	1	1	1
0	1	0	0

(d)

Figure 14. Binary image of Baa

1	0	0	1	1	1	1	1	1	1	1	1	0	1	1	0

(a)

1	0	0	0	1	1	0	1	0	1	1	1	0	1	1	0

(b)

1	0	0	0	1	1	1	1	0	1	1	1	0	0	1	0

(c)

1	0	0	1	1	1	0	1	1	1	1	1	0	1	0	0

(d)

Figure 15. One Dimension of Binary Images

1	0	0	1	1	1	1	1	1	1	1	1	0	1	1	0

(a)

1	0	0	0	1	1	0	1	0	1	1	1	0	1	1	0

(b)

1	0	0	0	1	1	1	1	0	1	1	1	0	0	1	0

(c)

1	0	0	1	1	1	0	1	1	1	1	1	0	1	0	0

(d)

1				1	1		1		1	1	1		1	1	

(e)

Figure 16. The Same Pixels Value of Baa One Dimension

After all the features from the samples of the previous processing were obtained, the next step was to compare each of the features with all the samples using City Block. City Block is an algorithm that is used to measure the gap or similarity between the two objects. The formulation of the City Block is presented as Equation 14 [25]:

$$D = \sum |F_i - \overline{F_i}| \quad (14)$$

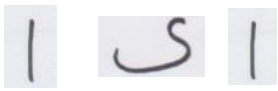
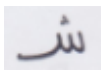
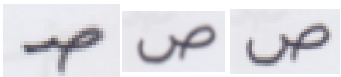
In which D = is the degree of distinction between two objects, $\overline{F_i}$ is input object, and $\overline{F_i}$ is compared object/output

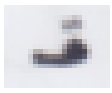
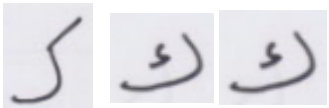
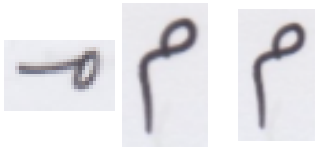
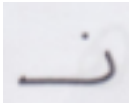
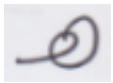
3 RESULT AND DISCUSSION

a. Trial

This study involved 101 samples of characters which each was written by 15 different handwriting characters (the total of the sample was 101x15). The data used in this study is preprocessed first, the preprocessing carried out includes convert the element of RGB (red, green, and blue) to grayscale using equation (1), differs the foreground from the background using threshold value (2-4), changing the value 0 to 1 or vice versa using the complement algorithm, attenuation image pixels using a thinning algorithm (5-13), the last preprocessing resize image data of handwriting Arabic character with the same size of 81x81. The example of data used in this study is shown in Table 1.

Table 1. Handwriting Arabic Character Data

No	Character	First	Middle	End	Isolated
1	Alif	-			
2	Baa				
3	Ta				
4	Tha				
5	Gheem				
6	Ha'a				
7	Kh'a				
8	Dal	-	-		
9	Dhal	-	-		
10	Ra	-	-		
11	Zeen	-	-		
12	Seen				
13	Sheen				
14	Sad				
15	Dad				
16	Tah				

17	Dhad			27	Waw	-	-	
18	Aen			28	Ya			
19	Ghen							
20	Fa							
21	Qaf							
22	Kaf							
23	Lam							
24	Meem							
25	Noon							
26	Ha							

The table above shows Arabic characters in various forms including the form when the letters stand alone, at the beginning of a word, in the middle, and at the end of a word. The Arabic characters used as the sample is the image of the handwritten hijaiyah Arabic Characters.

First, perform preprocessing on all samples of Arabic handwriting. After preprocessing, the next process is features extraction and recognition using City_Block (14). The result of handwriting Arabic character is presented using the calculation accuracy as shown below.

$$System\ Performance = \frac{The\ Same\ Pixels\ of\ input\ and\ fitur}{All\ Pixels\ of\ input\ and\ fitur} \times 100\% \quad (14)$$

Table 2. The Experiment Result of Handwriting Arabic Character Data

No	Character	Accuracy
1	Alif	85,51%
2	Baa	84,41%
3	Ta	80%
4	Tha	82%
5	Gheem	81,1%
6	Ha'a	82,30%
7	Kh'a	80%
8	Dal	82%
9	Dhal	81%

10	Ra	85%
11	Zeen	81%
12	Seen	85%
13	Sheen	80%
14	Sad	81%
15	Dad	80%
16	Tah	86,2%
17	Dhad	81,1%
18	Aen	83%
19	Ghen	81,3%
20	Fa	80,5%
21	Qaf	80%
22	Kaf	82,7%
23	Lam	83,1%
24	Meem	82,9%
25	Noon	81,5%
26	Ha	79%
27	Waw	86%
28	Ya	78%

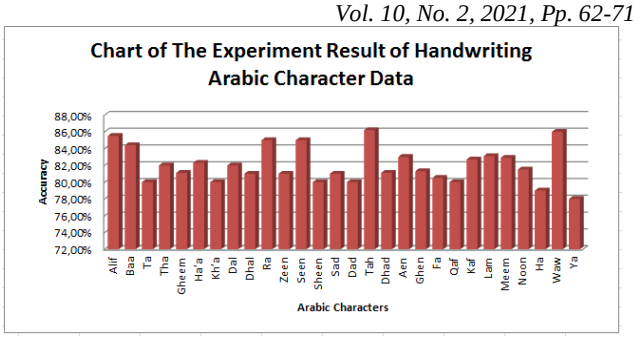


Figure 17. Chart of The Experiment Result of Handwriting Arabic Character Data

From the results obtained, by using Features Combination, the letters Ya and Ha have the smallest accuracy values compared to other letters, which are about 78% and 79%. This is because the shape of the letters is almost similar to other letters, the level of inclination of the writing, the position of the points that are not the same from the 10 samples, different writing styles. In addition, the shape of the writing similarity, the slope, and the size of the writing greatly affects the level of accuracy, the detail of each letter can be seen in Fig. 17.

4 CONCLUSION

The results, based on the experiment can be concluded that the recognition of handwriting Arabic characters using features combination method is very good for recognizing handwriting Arabic characters. In this study, we used image sizes of 81x81 pixels, we got a fairly good accuracy value of up to 80 %. The great quality of data will increase the accuracy. Accuracy is influenced by several factors, such as the degree of similarity, slope, and size.

AUTHOR'S CONTRIBUTION

As the first author, Fitriyatul Qomariyah contributed technically and wrote to this research. She did the research with the supervision from the second author namely Fitri Utaminingrum for ideas and technical aspects. Last, the research got theoretically supervision from the third author with Muchlas.

COMPETING INTERESTS

Comply with the publication ethics of this journal, Fitriyatul Qomariyah, Fitri Utaminingrum and Muchlas as the authors of this article declare that this article is free from conflict of interest (COI) or competing interest (CI).

REFERENCES

[1] A. Farhat *et al.*, "Optical character recognition on heterogeneous SoC for HD automatic number plate recognition system," *Eurasip J. Image Video Process.*,

vol. 2018, no. 1, 2018.
[2] J. Memon, M. Sami, R. A. Khan, and M. Uddin, "Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)," *IEEE Access*, vol. 8, pp. 142642–142668, 2020.
[3] H. Ghaleb, P. Nagabhushan, and U. Pal, "Segmentation of offline handwritten Arabic text," pp. 41–45, 2017.
[4] F. Qomariyah and F. Utaminingrum, "The Segmentation of Printed Arabic Characters Based on Interest Point," *Advencement Res. Circuit Syst. Int. Conf.*, pp. 2–6, 2016.
[5] M. O. Assayony and S. A. Mahmoud, "Integration of Gabor Features with Bag-of-Features Framework for Arabic Handwritten Word Recognition," *2017 9th IEEE-GCC Conf. Exhib. GCCCE 2017*, pp. 1–4, 2018.
[6] S. Ahmed Medjahed, "A Comparative Study of Feature Extraction Methods in Images Classification," *Int. J. Image, Graph. Signal Process.*, vol. 7, no. 3, pp. 16–23, 2015.
[7] M. Abdelnafea and S. Heshmat, "Efficient Preprocessing Algorithm for Online Handwritten Arabic Strokes," *Proc. 2019 Int. Conf. Innov. Trends Comput. Eng. ITCE 2019*, no. February, pp. 64–69, 2019.
[8] A. Lawgali and A. Bouridane, "Handwritten Arabic Character Recognition: Which feature extraction method," *Int. J. Adv. Sci. Technol.*, vol. 34, no. September, pp. 1–8, 2011.
[9] M. A. Abdullah, L. M. Al-Harigy, H. H. Al-Fraidi, M. a Abdullah, L. M. Al-harigy, and H. H. Al-fraidi, "Off-Line Arabic Handwriting Character Recognition Using Word Segmentation," *J. Comput.*, vol. 4, no. 3, pp. 40–44, 2012.
[10] A. Sahlol and C. Suen, "A Novel Method for the Recognition of Isolated Handwritten Arabic Characters," *arXiv Prepr. arXiv1402.6650*, 2014.
[11] R. S. Bahri and I. Maliki, "Perbandingan Algoritma Template Matching Dan Feature Extraction Pada Optical Character Recognition," *J. Komput. dan Inform.*, vol. 1, no. 1, pp. 29–35, 2012.
[12] R. I. O. Ahmed and M. E. M. Musa, "Preprocessing Phase for Offline Arabic Handwritten Character Recognition," *Int. J. Comput. Appl. Technol. Res.*, vol. 5, no. 12, pp. 760–763, 2016.
[13] D. M. Jacquerye, "Proposal to encode Javanese and Sundanese Arabic characters," 2019.
[14] F. Qomariyah, F. Utaminingrum, and W. F. Mahmudy, "The segmentation of printed Arabic characters based on interest point," *J. Telecommun. Electron. Comput. Eng.*, vol. 9, no. 2–8, pp. 19–24, 2017.
[15] F. Damayanti, S. Herawati, Imamah, F. A. M, and A. Rachmad, "Indonesian license plate recognition based on area feature extraction," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 17, no. 2, pp. 620–627, 2019.
[16] D. A. Prabowo and D. Abdullah, "Deteksi dan Perhitungan Objek Berdasarkan Warna Menggunakan Color Object Tracking," *Pseudocode*, vol. 5, no. 2, pp. 85–91, 2018.
[17] K. Kumar, R. K. Mishra, and D. Nandan, "Efficient Hardware of RGB to Gray Conversion Realized on FPGA and ASIC," *Procedia Comput. Sci.*, vol. 171, no. 2019, pp. 2008–2015, 2020.
[18] M. Balsky, "Luminance values extraction from digital images," *Proc. 2016 IEEE Light. Conf. Visegr. Countries, Lumen V4 2016*, pp. 1–3, 2016.
[19] S. Bhahri and Rachmat, "Transformasi Citra Biner Menggunakan," *J. Sist. Inf. dan Teknol. Inf.*, vol. 7, no. 2, pp. 195–203, 2018.
[20] J. Yoo, I. Karpov, S. Lee, J. Jung, H. S. Kim, and H. Hwang, "Threshold Voltage Drift in Te-Based Ovonic Threshold Switch Devices under Various Operation Conditions," *IEEE Electron Device Lett.*, vol. 41, no. 1, pp. 191–194, 2020.
[21] Y. Kim, Y. Lee, and M. Jeon, "Imbalanced image classification with complement cross entropy," *Pattern Recognit. Lett.*, vol. 151, pp. 33–40, 2021.
[22] M. Wang, L. Zhenglin, S. Fuyuan, and G. Lei, "An Improved Image Thinning Algorithm and Its Application in Chinese Character Image Refining," no. Itnec, pp. 1870–1874, 2019.
[23] Abhisek and K. Lakshmesha, "Thinning approach in digital image processing," *Last Accessed April*, pp. 326–330, 2018.
[24] R. M. Haralick, "A Fast Parallel Algorithm for Thinning Digital Patterns," vol. 27, no. 3, pp. 236–239, 1984.
[25] T. Vardoulaki, M. Vakalopoulou, and K. Karantzalos, "Per city-block, density estimation at build-up areas from aerial RGB imagery with deep learning," *2017 Jt. Urban Remote Sens. Event, JURSE 2017*, pp. 0–3, 2017.