International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 2, Issue 2, ISSN No. 2455-2143, Pages 76-80
Published Online December-January 2016 in IJEAST (http://www.ijeast.com)

OPTIMIZED VORONOI IMAGE ZONING FOR HANDWRITTEN CHARACTER RECOGNITION BASED ON KOHONEN NEURAL NETWORKS

V L Padmalatha
Adhoc Lecturer, Department of CSE,
JNTUA College of Engineering, Pulivendula

M Sampoorna
Assistant Professor, Department of CSE
Sri Indu College of Engineering, Hyderabad

Abstract — In Handwritten Character Recognition, zoning is rightly regarded as one of the best feature extraction techniques. In the past, many zoning methods have been suggested, predicated on dynamic and static zoning design techniques. However, little attention is paid so far to the position of kohonen neural networks; define the method by which a feature influences various zones of the pattern. In this paper the effectiveness of kohonen networks for zoning-based classification is examined. For that purpose, a useful illustration of zoning strategies based on Voronoi Diagram is adopted according to measurement-levels, ranked- and abstract– approaches. The process devised here can be tagged as Optimized Voronoi Image zoning for Handwritten Character Recognition (OVIZ-HCR). The experimental tests, carried out within the field of hand-written number recognition, show the superiority of OVIZ-HCR in comparison with regular functions, whatever zoning approach is used.

Keywords- **Component Formatting, Handwritten Character Recognition, Feature Extraction, Zoning, Voronoi Diagrams**.

I. INTRODUCTION

In the field of handwritten character recognition, zoning is measured as one of the most effective feature extraction method able to handle handwritten pattern variability, due to dissimilar writing styles and personal changeability in writing. Strictly speaking, given a model image B, a zoning $Z_M = \{z_1, z_2, ..., z_M\}$ of B is a separation of B into M sub images, named zones, each one providing in sequence related to a specific part of the pattern [1].

Traditional ways use static zoning practices, in which zoning style is obtained by common grids that are superimposed on pattern images. In this case, no a-priori information on element distribution can be used for defining the method. Recently, energetic zoning techniques have already been proposed, in which zoning design is recognized as an optimization problem and the optimal zoning process is available as the zoning which maximizes the classification performance, estimated with a well-defined cost function associated for the classification process [2, 3]. For this purpose, Voronoi Diagrams have already been lately proposed for zoning explanation, since they provide, given a set of points called Voronoi points in continuous space, a way of partitioning the space into sub-regions (named zones) according to proximity relationships among the set of points [4].

Unfortunately, even though zoning is largely used and its success is commonly demonstrated, factors associated with the option of feature-zone membership functions have not been addressed yet. However, account characteristics play a vital role. in analyzing the potential of a zoning strategy given that they ought to be able to design the way in that the features detected in a pattern influence different zones. Ergo, when zoning can be used, the decision of a membership function requires particular Giuseppe Pirlo et al [10], proposed a technique for zoning design, which can be applied to any zoning-based classifier, without limitations in terms of type of features and classification method. Actually depending on the requirements of the specific application that are formalized through the CF associated to the classification performance the new technique is able to determine the optimal zoning topology and to select the best adaptive membership functions depending on the feature set and classification technique considered. The main weakness of the proposed approach is that the number of zones must be defined a priori. Therefore, an important advancement in the technique can be certainly achieved by optimizing also the number of zones of the zoning method.

Starting from this consideration, this paper investigates the potency of conventional membership functions and introduces a fresh membership function with adaptive features. Moreover, the paper devised a real-coded genetic algorithm for determining both the optimal zoning approach, centered on Voronoi Diagram, and the adaptive account function most profitable for confirmed classification problem.

The outcomes show the performance of a method clearly depends on the membership function considered. Moreover, they show that adaptive membership functions are more advanced than conventional functions, whatever zoning approach is employed.

II. KOHONEN NETWORKS

The Kohonen neural network contains only an input and output layer of neurons. There is no hidden layer in a Kohonen neural network. First we will examine the input and output to a Kohonen neural network. The input to a Kohonen neural network is given to the neural network using the input neurons. These input neurons are each given the floating point numbers that make up the input pattern to the network. A Kohonen neural network requires that these inputs be normalized to the range between -1 and 1. Presenting an input pattern to the network will cause a reaction from the output neurons. In a Kohonen neural network only one of the output neurons actually produces a value. Additionally, this single value is either true or false. When the pattern is presented to the Kohonen neural network, one single output neuron is chosen as the output neuron. Therefore, the output from the Kohonen neural network is usually the index of the neuron that fired. The structure of a typical Kohonen neural network is shown in Figure 1.

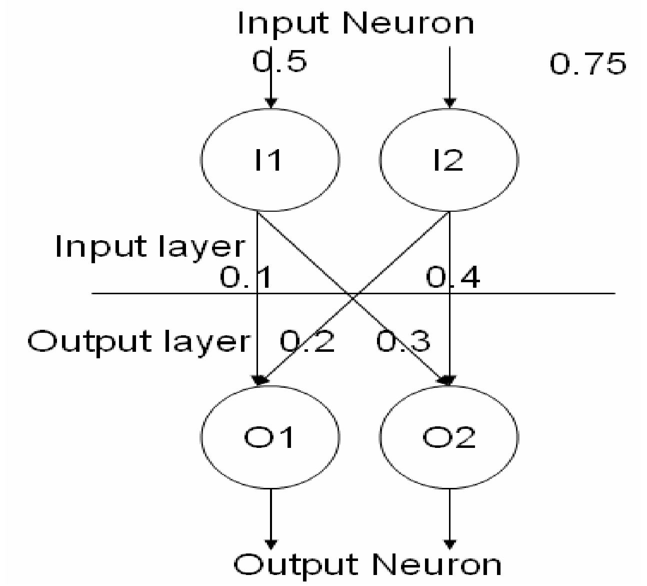


Figure 1: A Simple Kohonen Neural Network with 2 input and 2 output neurons

A. Sample Input to Kohonen Network

As we understand the structure of the Kohonen neural network we will examine how the network processes information. To examine this process we will step through the calculation process. For this example we will consider a very simple Kohonen neural network. This network will have only two input and two output neurons. The input given to the two input neurons is shown in Table 1.

Table 1: Sample Inputs to a Kohonen Neural Network

Input Neuron 1 (I1)	0.5
Input Neuron 2 (I2)	0.75

We must also know the connection weights between the neurons. These connections weights are given in Table 2.

Table 2: Connection weights in the sample Kohonen neural network

I1->O1	0.1
I2->O1	0.2
I1->O2	0.3
I2->O2	0.4

Using these values we will now examine which neuron would win and produce output. We will begin by normalizing the input.

B. Normalizing the input

The requirements that the Kohonen neural network places on its input data are one of the most severe limitations of the Kohonen neural network. Input to the Kohonen neural network should be between the values -1 and 1. In addition, each of the inputs should fully use the range. If one, or more, of the input neurons were to use only the numbers between 0 and 1, the performance of the neural network would suffer. To normalize the input we must first calculate the "vector length" of the input data, or vector. This is done by summing the squares of the input vector. In this case it would be. (0.5 * 0.5) + (0.75 * 0.75). This would result in a "vector length" of 0.8125. If the length becomes too small, say less than the length is set to that same arbitrarily small value. In this case the "vector length" is a sufficiently large number. Using this length we can now determine the normalization factor. The normalization factor is the reciprocal of the square root of the length. For our value the normalization factor is calculated as follows, 0.8125 1 and this results in a normalization factor of 1.1094. This normalization process will be used in the next step where the output layer is calculated.

C. Calculating each neuron's output

To calculate the output the input vector and neuron connection weights must both be considered. First the "dot product" of the input neurons and their connection weights must be calculated.

To calculate the dot product between two vectors you must multiply each of the elements in the two vectors. We will now examine how this is done. The Kohonen algorithm specifies that we must take the dot product of the input vector and the weights between the input neurons and the output neurons. The result of this is as follows.

$$\begin{vmatrix}0.5 & 0.75\end{vmatrix} \bullet \begin{vmatrix}0.1 & 0.2\end{vmatrix} = (0.5*0.75)+(0.1*0.2)$$

As we can see from the above calculation the dot product would be 0.395. This calculation will be performed for the first output neuron. This calculation will have to be done for each of the output neurons. Through this example we will only examine the calculations for the first output neuron. The calculations necessary for the second output neuron are calculated in the same way. This output must now be normalized by multiplying it by the normalization factor that was determined in the previous step. We must now multiply the dot product of 0.395 by the normalization factor of 1.1094. This results in an output of 0.438213. Now that the output has been calculated and normalized it must be mapped to a bipolar number.

D. Mapping to bipolar

In the bipolar system the binary zero maps to -1 and the binary remains a 1. Because the input to the neural network normalized to this range we must perform a similar normalization to the output of the neurons. To make this mapping we add one and divide the result in half. For the output of 0.438213 this would result in a final output of 0.7191065. The value 0.7191065 is the output of the first neuron. This value will be compared with the outputs of the other neuron. By comparing these values we can determine a "winning" neuron.

E. Choosing a winner

We have seen how to calculate the value for the first output neuron. If we are to determine a winning output neuron we must also calculate the value for the second output neuron. We will now quickly review the process to calculate the second neuron. For a more detailed description you should refer to the previous section. The second output neuron will use exactly the same normalization factor as was used to calculate the first output neuron. As you recall from the previous section the normalization factor is 1.1094. If we apply the dot product for the weights of the second output neuron and the input vector we get a value of 0.45. This value is multiplied by the normalization factor of 1.1094 to give the value of 0.0465948. We can now calculate the final output for neuron 2 by converting the output of 0.0465948 to bipolar yields 0.49923.

As we can see we now have an output value for each of the neurons. The first neuron has an output value of 0.7191065 and the second neuron has an output value of 0.49923. To choose the winning neuron we choose the output that has the largest output value. In this case the winning neuron is the first output neuron with an output of 0.7191065, which beats neuron two's output of 0.49923. We have now seen how the output of the Kohonen neural network was derived. As we can see the weights between the input and output neurons determine this output.

F. Kohonen Network learning procedure

The training process for the Kohonen neural network is competitive. For each training set one neuron will "win". This winning neuron will have its weight adjusted so that it will react even more strongly to the input the next time. As different neurons win for different patterns, their ability to recognize that particular pattern will be increased.

IV. DYNAMIC ZONING DESCRIPTION BY VORONOI DIAGRAM

Voronoi Diagram is a widespread technique of computational geometry that has been applied to more than a few fields, ranging from biology to chemistry, from VLSI chip design to medicine, form physics to cartography [4]. Strictly speaking, given a set of a restricted number of M distinct points $p_1, p_2, ..., \mathrm{p}_M$ in the Euclidean plane, the Voronoi Diagram is the separation of the plane into M zones $z_1, z_2, ..., \mathrm{z}_M$ that reflects proximity relationships between the set of points. In other words, each point p_i determines a region z_i that is the locus of points which are closer to p_i than to any other point of the set, according to the Euclidean expanse [4]. Voronoi Diagrams have been newly used for zoning description [3, 5]. Static zoning methods are distinct without using a-priori information on characteristic distributions. They are designed according to personal knowledge of the designer and experimental tests. As illustrate in the literature, static zoning normally use regular partitioning criteria of the pattern image [1, 2]. Typical static zonings divide a $D_x \times D_y$ pattern image (D_x : image width, D_y : image height} into r × s equal rectangles [2]. A regular $r \times s$ zoning is represented by a Voronoi Diagram based on the set of M (M= r.s) Voronoi points $P = \{p_1, p_2, ..., \mathrm{p}_M\}$, with

$$p_i = (px_i, py_i) \qquad \mathrm{i}=1,2,...,\mathrm{M} \quad (1)$$

Where:

$$px_i = (1/2 + k_x)\cdot D_x / s \quad , \mathrm{k}_x = 0,1,2,...,s-1$$

$$py_i = (1/2 + k_y)\cdot D_y / r \quad , \mathrm{k}_y = 0,1,2,...,r-1 .$$

Being $i = 1 + k_x.r + k_y$.

Dynamic zoning methods are distinct on the basis of a priori information on feature distributions, according to explicit optimality criterion [3]. Unlike static methods, that use typical partitioning criteria of the pattern image, dynamic zoning methods regard as zoning design as an optimization difficulty and the optimal zoning $Z*_M = \{z*_1,\ z*_2, ..., \ \mathrm{z}*_M\}$ is found as the zoning for which the cost function $CF(Z_M)$ connected to classification is minimum [3, 5]. Also in this case, if a zoning method $Z_M = \{z_1, z_2,\ ...,\mathrm{z}_M\}$ is illustrate by the set of Voronoi points $P = \{p_1, p_2,\ ...,\mathrm{p}_M\}$, the optimal zoning design can be re-formulated as the difficulty of finding the set of Voronoi points $P* = \{p*_1,\ \mathrm{p}*_2, ..., \ \mathrm{p}*_M\}$ so that the equivalent zoning method $Z*_M = \{z*_1,\ z*_2, ..., \ \mathrm{z}*_M\}$ (i.e. $Z*_M$ is the Voronoi Diagram distinct from the set $P*$) leads to the minimum of the cost function connected to classification performance.

V. IMPLEMENTATION

Given a user-drawn image representing a character, we must convert this image into a format a neural network can understand, feed this data into a neural network constructed and trained such that it can recognize characters, and then transform the output of the neural network into something useful. We have used Java to create an application and an extensible framework to be used for character recognition. We construct an extension of the Java SWING JPanel to support user drawing, and then store the user-drawn character as a Java Image object. We then construct a bounding box around the character to do away with excess white space, and then we down sample the image to a double array of Boolean values (here "true" represents a pixel that has been drawn in) to feed to the neural network. Experimentally, we found that a 5×7 grid has the lowest error rate. Actual implementation issues, which comprised the vast majority of time spent on the project, are beyond the scope of this discussion. This Boolean double-array is passed to a Kohonen neural network, which was trained on similar Boolean arrays to recognize characters of the alphabet. As described in the previous section, the winning output neuron map to the character that the neural network believes most closely resembles what the user drew. This backend was attached to a Java SWING graphical user interface for the purposes of demonstrating the capabilities of the neural network at this stage in the project.

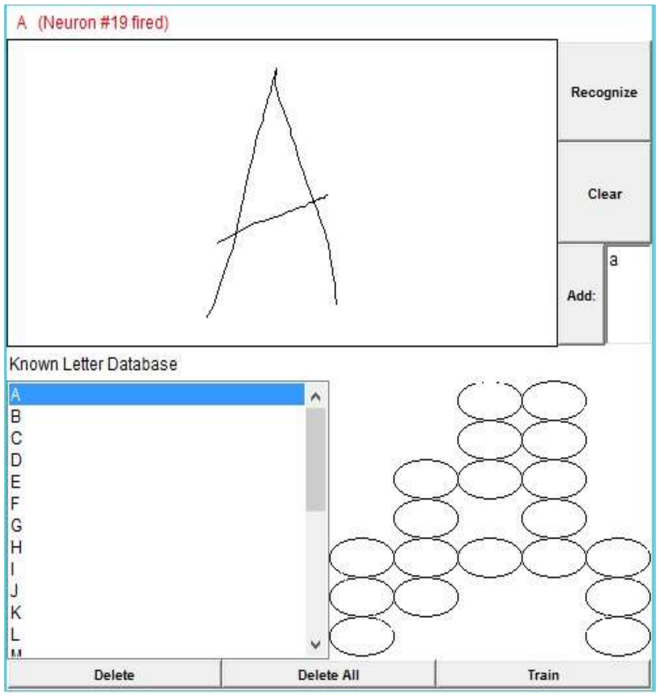


Fig 2: Recognition process and displaying the corresponding dynamic zones for character "A".

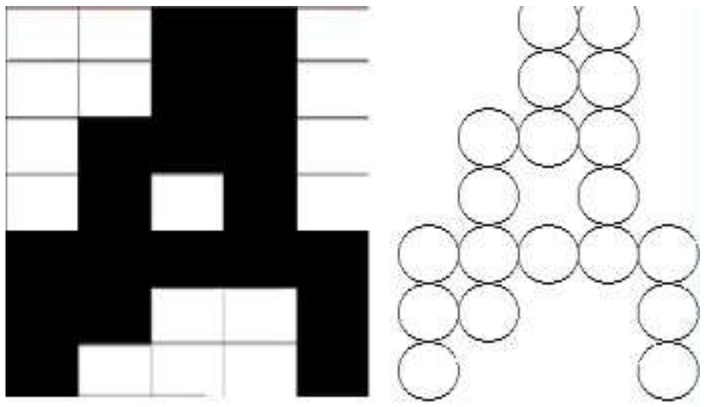


Fig 3: Comparison of Static Zones and Dynamic Zones

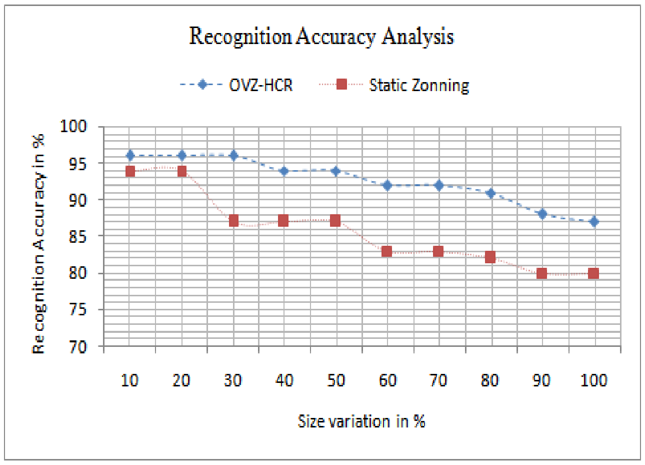
VI. EXPERIMENTAL RESULTS

The experiments have been carried out by means of the set of handwritten numeral digits $\Omega_1 = \{0,1,2,3,4,5,6,7,8,9\}$ extracted from the CEDAR database (BR directory) [7]. After normalization of each the prototype image to a size of 72×54 pixels, the skeleton of the prototype is derived through the Safe Point Thinning Algorithm [8]. Successively, the feature set

$F = \{ f_1, \ldots\ldots, f_9 \}$ is considered for pattern explanation, where (see [9] for more details): f_1 - holes; f_2 - vertical-up cavities; f_3 - vertical-down cavities; f_4 - horizontal-right cavities; f_5 - horizontal-left cavities; f_6 - vertical-up endpoints; f_7 - vertical-down end-points; f_8 - horizontal-right end-points; f_9 - horizontal-left end-points.

In order to pre-estimate the most commercial parameter values for the Genetic Algorithm, some beginning pilot tests have been conducted. According to previous studies in the literature [5, 6], the following restriction values have been considered: $N_{Pop} = 10$; $N^{iter} = 300$; $Mut\_prob = 0.35$; δ_displ ; $b = 1.0$; $displ = 0.5$, $c = 3.0$.

Figure 4: Recognition Accuracy Analysis



VII. CONCLUSION

This paper proposes a dynamic zoning scheme for handwritten character recognition using kohonen network. The experimental results, completed in the area of handwritten numeral recognition, show the flexible kohonen networks leads to the very best classification results, when compared to static zoning scheme. The accuracy of recognition is compared with the previous zoning technique (static) and results tend to be good and further training is needed for more accuracy rates.

VIII. REFERENCES

[1] A. Ferrante, S. Impedovo, R. Modugno, and G. Pirlo, “Zoning methods for hand-written character recognition: An overview,” inProc. Int. Conf. Frontiers Handwritten Recognit., Kolkata, India, Nov. 2010, pp. 329– 334.

[2] A. L. Koerich and P. R. Kalva, “Unconstrained handwritten character recognition using metaclasses of characters,” in Proc. IEEE Int. Conf. Image Process., Sep. 2005, pp. 542–545.

[3] P. Phokharatkul, K. Sankhuangaw, S. Phaiboon, S. Somkuarnpanit, and C. Kimpan, “Off-line hand written thai character recognition using antminer algorithm,”Trans. Enformatika Syst. Sci. Eng., vol. 8, pp. 276– 281, Oct. 2005.

[4] P. Xiang, Y. Xiuzi, and Z. Sanyuan, “A hybrid method for robust car plate character recognition,”J. Eng. Appl. Artif. Intell., vol. 18, no. 8, pp. 963–972, Dec. 2005.

[5] D. Sharma and D. Gupta, “Isolated handwritten digit recognition using adaptive unsupervised incremental learning technique,”Int. J. Comput. Appl., vol. 7, no. 4, pp. 27–33, Sep. 2010.

[6] G. Vamvakas, B. Gatos, S. Petridis, and N. Stamatopoulos, “An efficient feature extraction and dimensionality reduction scheme for isolated Greek handwritten character recognition,” inProc. 9th Int. Conf. Document Anal. Recognit., 2007, pp. 1073–1077.

[7] J. B. K. Aires, C. O. A. Freitas, F. Bortolozzi, and R. Sabourin, “Perceptual zoning for handwriting character recognition,” in Proc. 12th Conf. IGS, 2005, pp. 178–182.

[8] C. O. A. Freitas, L. S. Oliveira, and F. Bortolozzi, “Handwritten character recognition using nonsymmetrical perceptual zoning,”Int. J. Pattern Recognit. Artif. Intell., vol. 21, no. 1, pp. 135–155, 2007.

[9] C. O. A. Freitas, L. S. Oliveira, S. B. K. Aires, and F. Bortolozzi, “Metaclasses and zoning mechanism applied to handwritten recognition,” J. Universal Comput. Sci., vol. 14, no. 2, pp. 211–223, 2008.

[10] G. Pirlo and D. Impedovo, “Fuzzy zoning-based classification for handwritten characters,” IEEE Trans. Fuzzy Syst., vol. 19, no. 4, pp. 780–784, Aug. 2011.