Efficient Genetic Algorithms for Arabic Handwritten Characters Recognition

Laheeb Mohammad Ibrahim
College of Computer Sciences and Mathematics
University of Mosul, Iraq

Received on: 29/04/2008 **Accepted on: 03/09/2008**

ABSTRACT

The main challenge in Arabic handwritten character recognition involves the development of a method that can generate descriptions of the handwritten objects in a short period of time high recognition rate. Due to its low computational requirement, genetic algorithm is probably the most efficient method available for character recognition. In this research we use objective of genetic algorithm where the minimization of the number of features and a validity index that measures the quality of clusters have been used to guide the search towards the more discriminate features and the best number of clusters, and use Hopfield Neural Network as recognizer.

In this research Arabic handwritten characters recognition is applied. Experiments show the efficiency and flexibility of the proposed system, and show that Genetic Algorithm (GA) and Hopfield neural network are applied here to improve the recognition accuracy and make the recognition operation faster.

Keywords: Arabic handwritten characters recognition, Genetic Algorithm (GA), Feature Extracted, Feature Selection, Hopfield Neural Network.

الخوارزميات الجينية الفعالة لتمييز الحروف العربية المكتوبة بخط اليد

لهيب محمد ابراهيم
كليه علوم الحاسوب والرياضيات، جامعه الموصل

تاريخ استلام البحث: 2008/4/29 **تاريخ قبول البحث: 2008/9/3**

الملخص

يتمثل التحدي الرئيسي في تمييز الوثائق العربية المكتوبة بخط اليد بتطوير الطريقة التي يمكن ان تستخدم لاستخلاص أهم الخواص للحرف العربي بزمن تنفيذ قليل ودقة عالية في التمييز وبناء قاعدة بيانات تساعد على ذلك . ونظرا لقلة العمليات الحسابية عند تنفيذ الخوارزمية الجينية (Genetic Algorithm) ومساعدتها في الوصول الى الحل الأمثل اعتبرت هذه الطريقة أكفأ طريقة متاحة للحصول على أفضل النماذج لبناء قاعدة البيانات للحروف العربية المكتوبة بخط اليد. وقد استخدمنا في هذه البحث مميزات الخوارزمية الجينية لتقليل عدد الخواص (features) و الحصول على الحد الادنى منها في قاعدة البيانات واستخدمنا الشبكة العصبية الاصطناعية Hopfield neural network في توجيه البحث نحو تميز أدق للحروف العربية وميزات أفضل في عدد النماذج . أثبتت التجارب التي اجريت باستخدام الطريقة المقترحة في هذا البحث لبناء قاعدة بيانات للحروف العربية المكتوبة بخط اليد والتي تم الحصول عليها باستخدام الخوارزمية الجينية و الشبكة العصبية الاصطناعية Hopfield neural network لتمييز هذه الحروف جدوى المنهجية المقترحة وكفاءتها باستخدام الخورارزمية الجينية وشبكة hopfield في تحسين دقة عملية التمييز للحروف المكتوبة بخط اليد وسرعتها.

الكلمات المفتاحية: تمييز الحروف العربيه، المكتوبه باليد، الخوارزميه الجينيه، استخلاص الخواص، اختيار الخواص، الشبكه العصبيه الاصطناعيه هوبفيلد

1. Introduction

In areas of pattern recognition, features can be characterized as a way to distinguish one class of objects from another in a more concise and meaningful manner than is offered by the raw representations. Therefore, it is of crucial importance to define meaningful features when we plan to develop a good recognizer, although it has been known that a general solution has not been found. In many cases, features are generally defined by hand based on the experience and intuition of the designer. Depending on problems given, there are a number and variety of features that can be defined in terms of extracting methods and ways of representation. In many practical applications, it is not unusual to encounter problems involving hundreds of features. The designer usually believes that every feature is meaningful for at least some of the discriminations. However, it has been observed in practice that, beyond certain point, the inclusion of additional features leads to worse rather than better performance. Furthermore, including more features means simply increasing processing time. This apparent paradox presents a genuine and serious problem for classifier design [7,8,10,19].

Classification of Arabic handwritten characters, which has been a typical example of pattern recognition, contains the same problem. Due to diversity in ones written by a single person. In order to deal with such a wide range of diversity existing in handwritings, recognizers often employ several hundreds of features [6,11].

In this research, we introduce a feature selection method, which can minimize most of the problems and can be found in the conventional approaches, by applying genetic algorithms(GA) which recently received considerable attention regarding their potential as an optimization technique for complex problems. Genetic algorithms are stochastic search technique based on the mechanism of natural selection and natural genetics [15].

The objective is to find a set of non dominant solutions which contain the more discriminate features. We have used one criteria to guide the search by minimization of the number of features. Afterwards, Hopfield Artificial Neural network is applied for handwritten Arabic characters recognition in order to optimize the character classifiers. Experimental results show the efficiency of the proposed methodology

2. Genetic Algorithms (GA)

Genetic algorithms are a part of evolutionary computing, which is a rapidly growing area of artificial intelligence. Genetic algorithms are inspired by Darwin's theory of evolution. Simply said, problems are solved by an evolutionary process resulting in a best (fittest) solution (survivor) - in other words, the solution is evolved. All living organisms consist of cells. In each cell there is the same set of chromosomes. Chromosomes are strings of DNA and serve as a model for the whole organism. A chromosome consists of genes, blocks of DNA. Each gene encodes a particular protein. Basically, it can be said that each gene encodes a trait. Possible settings for a trait are called alleles. Each gene has its own position in the chromosome. This position is called locus. Complete set of genetic material (all chromosomes) is called genome. Particular set of genes in genome is called genotype. The genotype is with later development after birth base for the organism's phenotype, its physical and mental characteristics, such as eye color, intelligence etc [3,5,9].

Algorithm begins with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are then selected to form new solutions (offspring) are selected according to their fitness - the more suitable they are the more chances they have to reproduce. This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied [3,5].

There are two important steps in GA, recombination (or crossover) first occurs. Genes from parents combine to form a whole new chromosome. The newly created offspring can then be mutated. Mutation means that the elements of DNA are a bit changed. This change is mainly caused by errors in copying genes from parents. The fitness of an organism is measured by success of the organism in its life (survival) [3,5].

Algorithm of GA [3,5]

1. **[Start]** Generate random population of *n* chromosomes (suitable solutions for the problem)
2. **[Fitness]** Evaluate the fitness *f(x)* of each chromosome *x* in the population.
3. **[New population]** Create a new population by repeating following steps until the new population is complete

- **[Selection]** Select two parent chromosomes from a population according to their fitness (the better the fitness, the bigger chance to be selected)
- **[Crossover]** With a crossover probability cross over the parents to form a new offspring (children). If no crossover was performed, offspring is the exact copy of parents.
- **[Mutation]** With a mutation probability mutate a new offspring at each locus (position in chromosome).

- **[Accepting]** Place a new offspring in the new population
4. **[Replace]** Use a new generated population for a further run of the algorithm
5. **[Test]** If the end condition is satisfied, **stop**, and return the best solution in current population
6. **[loop]** Go to step **2**

The performance of genetic algorithm depends on a number of factors including: the choice of genetic representation and operators, the fitness function, the details of the fitness-dependent selection procedure, and the various user-determined parameters such as population size, probability of application of different genetic operators, etc. The specific choices made in the experiments reported, (see Figure 1) depicts a GA cycle.

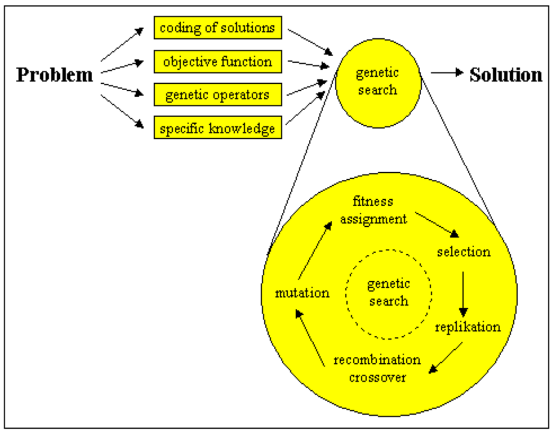


Figure 1. Genetic Algorithm (GA) Cycle

Since genetic algorithms were designed to efficiently search large spaces, they have been used for a number of different application areas, and genetic algorithms mainly deal with optimization problems. The applications include job scheduling, TSP (Traveling Salesman Problem), communication network design, image restoration, data clustering, and feature selection for speaker identification, camera calibration [18], signature verification [18], medical diagnosis [20], facial modeling [7] and handwritten recognition [9].

3. The Characteristic of an Arabic Text

The most obvious characteristics of the Arabic language is that the Arabic scripts are inherently cursive. Arabic script, which comprises 28 main characters and which is written from right to left, is used by many nations: Arabs, kurds and Urdus.

The shape of the letters is context sensitive, depending on their position within a word (beginning, middle, end, isolated). In Arabic, the letters represent only consonants or long vowels. Short vowels are represented by 14 optional diacritical marks written over or under the letters. Table 1 outlines a comparison between the various characteristics of Arabic and Latin scripts [14].

Table 1 : Comparison of various scripts

Characteristics	Arabic	Latin
Justification	R – to - L	L – to - R
Cursive	Yes	No
Diacritics	Yes	No
Number of vowels	2	5
Letters shapes	1 – 4	2
Number of letters	28	26
Complementary characters	3	-

4. Recognition System

Figure (2) briefly shows the system flow of a general handwriting recognizer. An input image is encoded for saving space and easier manipulation in the subsequent steps. Several steps of preprocessing, such as segmentation, noise removal, slant correction, and smoothing are applied to the encoded image. Defined features are extracted from the processed image and the recognition process is performed using the features [11].

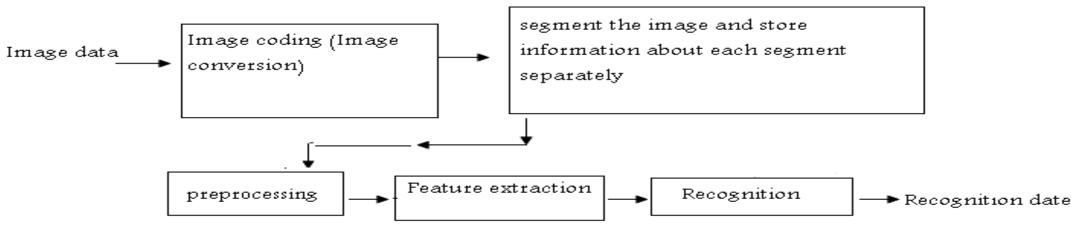


Image data
Image coding (Image conversion)
segment the image and store information about each segment separately
preprocessing
Feature extraction
Recognition
Recognition date

Figure 2. Handwritten Recognition System

5. Proposed Arabic Recognition System

The recognition system is divided into three phases: segmentation of a word into characters and preprocessing each character, extracted and selected features of characters and then word recognition, (see figure 3)

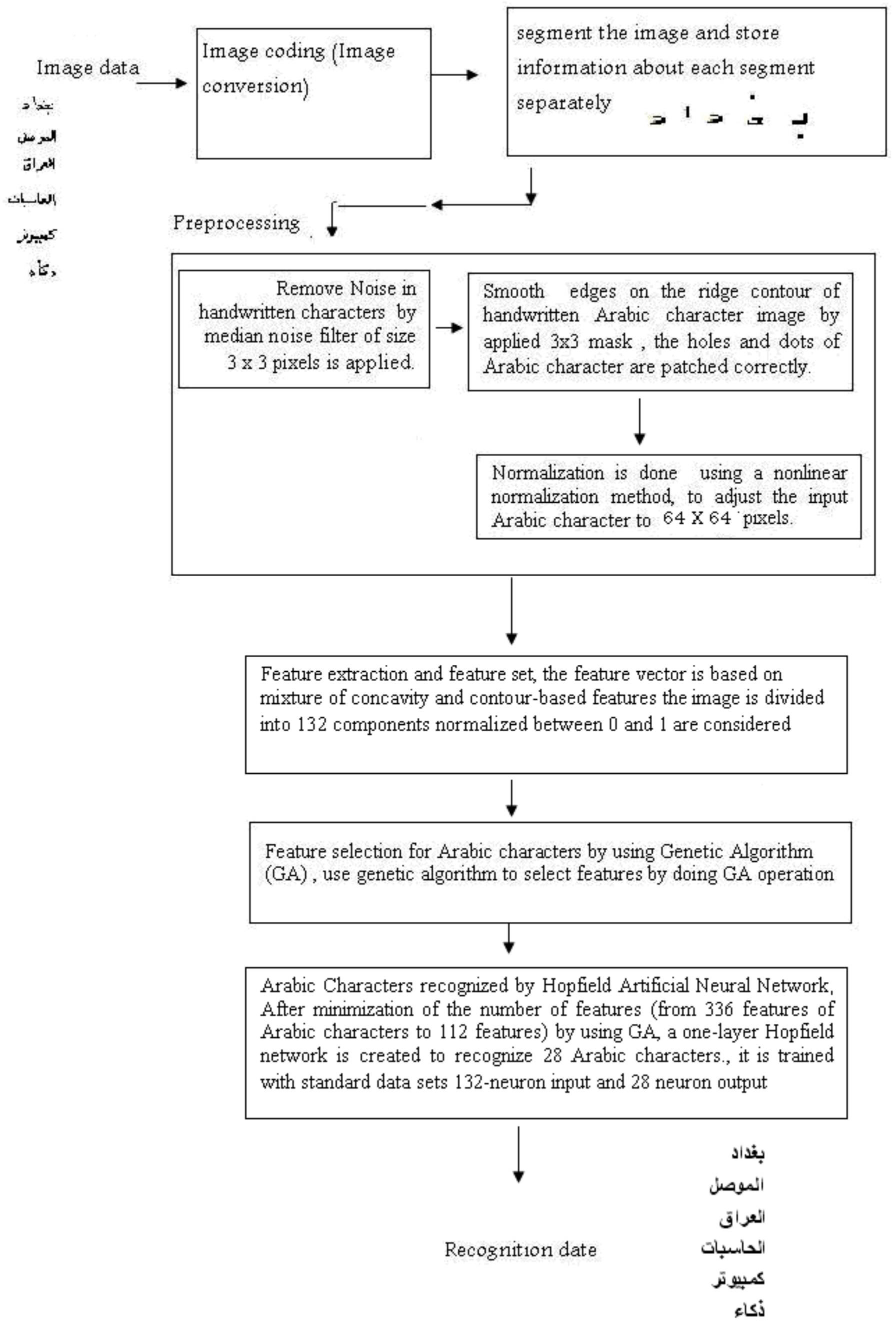


Image data
Image coding (Image conversion)
segment the image and store information about each segment separately
Preprocessing
Remove Noise in handwritten characters by median noise filter of size 3 x 3 pixels is applied.
Smooth edges on the ridge contour of handwritten Arabic character image by applied 3x3 mask , the holes and dots of Arabic character are patched correctly.
Normalization is done using a nonlinear normalization method, to adjust the input Arabic character to 64 X 64 pixels.
Feature extraction and feature set, the feature vector is based on mixture of concavity and contour-based features the image is divided into 132 components normalized between 0 and 1 are considered
Feature selection for Arabic characters by using Genetic Algorithm (GA) , use genetic algorithm to select features by doing GA operation
Arabic Characters recognized by Hopfield Artificial Neural Network, After minimization of the number of features (from 336 features of Arabic characters to 112 features) by using GA, a one-layer Hopfield network is created to recognize 28 Arabic characters., it is trained with standard data sets 132-neuron input and 28 neuron output
Recognition date
بغداد
الموصل
العراق
الحاسبات
كمبيوتر
ذكاء

Figure 3. Proposed Arabic Recognition System

5.1 Segmentation and Preprocessing

The idea of segmentation of cursive word into characters is to find the start point using the algorithm below [14]:

[Find] find the right most point x1

```
 If x1 is line end
   [Start]
   follow the curve until a branch point x2
   If x2 is found
      [ Start]
            If the segment between x1 is a short line
              [Start ]
              Turn to the right
              Follow the curve until a feature point x3
              Start point = x3
              [End]
            Else
              Start point = x1
      [End]
   Else
      [Start]
              Turn to the right
               Follow the curve until a feature point x2
                Start point = x2
      [End]
[End]
```

The different characters divided into five classes

Classes No.	Examples of Arabic letters	Description
Class 1	.	Points defined as isolated segments with a very small length
Class 2	ء	Hamza defined as a small isolated zig zag
Class 3	م و ح هـ	Segments with loops
Class 4	د عـ	Segments without loops and which finish with connections or intersection
Class 5	ل ا	Segments without loops and which finish with end point

The image of segmented Arabic character is preprocessed by performing the following operation removing distortion and noise, smoothing and normalization [1], preprocessing is done by performing Noise removal operation on segmented image where the noise in handwritten characters produces non-uniform ink density, non-printed area, and the presence of distortion and noise, to overcome these problems, median noise filter of size 3 x 3 pixels is applied [4], see Appendix (A), after that perform smooth edges on the ridge contour of handwritten Arabic character image by applying 3x3 mask , the holes and dots of Arabic character are patched correctly [4], at the end Normalization is achieved by stretching or compressing the characters. This is achieved by using a nonlinear normalization method to adjust the input Arabic character to 64 x 64 pixels.

5.2 Feature Extraction and Feature Set

In this section we describe both feature extraction and feature set used in our experiments. The feature vector is based on a mixture of concavity and contour-based features [15,16,17].

The basic idea of concavity measurements [6] is the following: for each white pixel in the component, we verify in each possible direction (Figure 4a), if a black pixel can be reached. The number of times as well as the directions leading to the black pixels are computed and stored in a vector. When black pixels are reached in four directions, we branch out in four auxiliary directions in order to confirm if the current white pixel is really inside a closed contour. Those pixels that reach just one black pixel are discarded. Therefore, the concavity measurements are represented by 13 components. The second part of the vector contains contour information, which is extracted from a histogram of contour directions. Taking into account 8-Freeman directions [12,21] (Figure 4b), we have 8 more components in our feature vector. The last component of this vector corresponds to the character surface. Finally, the image is divided into four regions and 132 components normalized between 0 and 1 are considered.

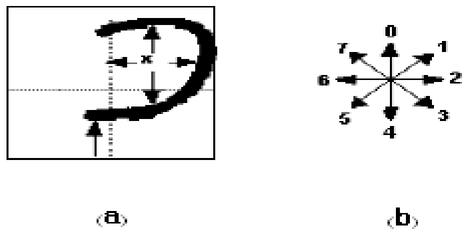


Figure. 4. Feature set for Arabic character Dal (د) : (a) Concavities, (b) 8-Freeman directions

5.3 Feature selection for Arabic characters by Genetic Algorithm (GA)

After the Arabic words are segmented, preprocessed, and feature extracted we get Feature set of segmented Arabic characters, we use genetic algorithm to select features by doing the following steps.

1. **[Start]** Generate random population of *n* chromosomes (*in our research n*=336), each individual in the population represents a candidate solution

to the feature subset selection problem. Let *m* be the total number of genes available to choose from to represent the pattern to be classifier (*in our research m*=132). The individual (chromosome) is represented by a binary vector of dimension *m*. Since we are representing a chromosome through a **binary string**, it is most used way of encoding in GA. A chromosome then could look like this:

Chromosome 1	1101100100110110 … 101

Our experiments used the following parameter settings: Chromosome size: 132, Population size: 336, Number of generation:1500, Probability of crossover: 0.8, Probability of mutation: 0.007
The parameter settings were based on results of several preliminary runs.

2. **[Fitness]** Evaluate the fitness *f(x)* of each chromosome *x* in the population.
The fitness evaluation is a mechanism used to determine the confidence level of the optimized solutions to the problem. Usually, there is a fitness value associated with each chromosome. In our research Distance between the centroid of a cluster and the feature is computed using equation (1), in order to consider the features selected only:

$$\text{Dist} = \sum_{i=0}^{p} (x_i - y_i)^2 \qquad \text{where } s_i = 1 \qquad \text{...(1)}$$

where, x_i is the i-th feature extracted from testing data, y_i is the i-th feature of any cluster in the feature set , p is the number of features before the selection is performed, and s_i indicates whether the i-th feature is selected (1) or not (0).

3. **[New population]** Create a new population by repeating following steps until the new population is complete
   - **[Selection]** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected). Chromosomes are selected from the population to be parents for crossover. In this work we are using the roulette wheel selection which is one of the most common and easy-to implement selection mechanism. Basically it works as follows:

   Parents are selected according to their fitness. The better the chromosomes are, the more chances to be selected they have. Imagine a **roulette wheel** where all the chromosomes in the population are placed. The size of the section in the roulette wheel is proportional to the value of the fitness function of every

sample 1	0 1 1 0 0 0 1 1 0 1 0
sample 2	1 0 0 1 1 1 0 0 1 0 1

chromosome - the bigger the value is, the larger the section is. The individuals are mapped to contiguous segments of a line, such that each individual's segment is equal in size to its fitness. A random number is generated and the individual whose segment spans the random number is selected. The process is repeated until the desired number of individuals is obtained (called mating population). This technique is analogous to a roulette wheel with each slice proportional in size to the fitness [5, 13].

This process can be described by the following algorithm. Programmed by using matlab GA functions, See Apendix (B)

1. **[Sum]** Calculate the sum of all chromosome fitnesses in population - sum ***S***.
2. **[Select]** Generate random number from the interval ***(0,S)*** - ***r***.
3. **[Loop]** Go through the population and sum the fitnesses from ***0*** - sum ***s***. When the sum ***s*** is greater then ***r***, stop and return the chromosome where you are.

Of course, the step **1** is performed only once for each population.

- **[Crossover]** With a crossover probability crossover the parents to form new offspring (children). After we have decided what encoding we will use, we can proceed to crossover operation. In our research we used **Uniform crossover** [5] generalizes this scheme to make every locus a potential crossover point. A crossover mask, the same length as the individual structure is created at random and the parity of the bits in the mask indicates which parent will supply the offspring with which bits. Consider the following two individuals with 11 binary variables each:

individual 1	0 1 1 1 0 0 1 1 0 1 0
individual 2	1 0 1 0 1 1 0 0 1 0 1

For each variable the parent who contributes its variable to the offspring is chosen randomly with equal probability. Here, the offspring 1 is produced by taking the bit from parent 1 if the corresponding mask bit is 1 or the bit from parent 2 if the corresponding mask bit is 0. Offspring 2 is created using the inverse of the mask, usually.

After crossover the new individuals are created:

offspring 1	1 1 1 0 1 1 1 1 1 1 1
offspring 2	0 0 1 1 0 0 0 0 0 0 0

Uniform crossover, like multi-point crossover, has been claimed to reduce the bias associated with the length of the binary representation used and the particular coding for a given parameter set. This helps to overcome the bias in single-point crossover towards short substrings without requiring precise understanding of the significance of the individual bits in the individual's representation, uniform crossover may be parameterized by applying a probability to the swapping of bits. This extra parameter can be used to control the amount of disruption during recombination without introducing a bias towards the length of the representation used [3, 5].

- [**Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome), we used in our research **binary valued individuals mutation** this mean flipping of variable values. For every individual the variable value to change is a chosen uniform at random.
- **[Accepting]** Place new offspring in the new population

4. **[Replace]** Use new generated population for a further run of the algorithm
5. **[Test]** If the end condition is satisfied, **stop**, and return the best solution in current population
6. **[loop]** Go to step 2.

GA Algorithm is programmed using GATOOL Matlab , see Appendix (C)

5.4 Recognizer for Arabic characters by Hopfield Artificial Neural Network

Our problem consists of optimizing two objectives: minimization of the number of features (The experimental results prove that feature selection using the GA reduces 66% of features from 336 features of Arabic characters to 112 features) and minimization of the error rate of the recognizer.

Using genetic algorithms for feature set selection contains the running of a genetic algorithm for several generations. In each generation, training the corresponding neural network on feature set and computing its accuracy. Hopfield artificial neural network algorithm is used as recognizer. Such a recognition module has been successfully applied on handwritten Arabic characters recognition

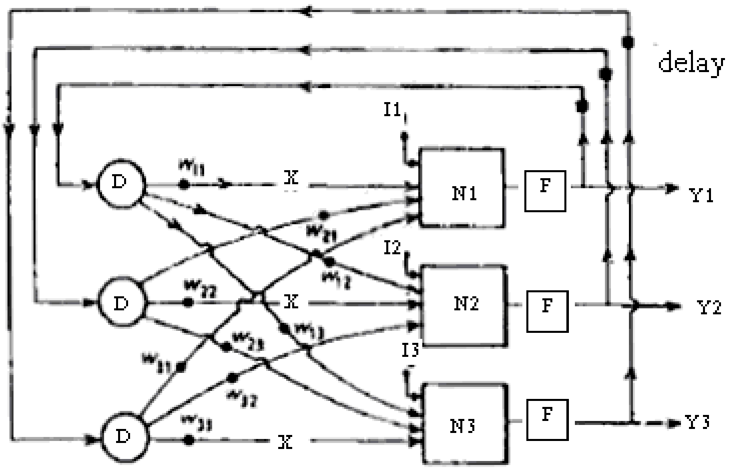
Figure 5 illustrates a recurrent single layer Hopfield network. Though it is basically a single-layer network, its feedback structure makes it effective to behave as a multi-layer network. The delay in the feedback will be shown to play a major role in its stability. Such a delay is natural to biological neural networks, noting the delay in the synaptic gap and the finite rate of neuronal firing. Whereas Hopfield networks can be of continuous or of binary output, we consider first a binary Hopfield network, to introduce the concepts of the Hopfield network.

The network of Figure 5 thus satisfies [2].

$$Z_j=\sum_{i\neq j} w_{ij}y_i(n) + I_j,\ n=0,1,2 \ldots \quad \ldots(2)$$

$$y_j(n+1) = \begin{cases} 1 \ \forall \ z_j >= Th_j \\ \\ 0 \ \forall \ z_j < Th_j \end{cases} \quad \begin{cases} 1 \quad \forall \quad z_j(n) > Th_j \\ \text{or} \quad y_j(n) \quad \forall \quad z_j = Th_j \\ 0 \quad \forall \quad z_j < Th_j \end{cases} \quad \ldots(3)$$

The ii weight in equation 2 is zero to indicate no self feedback. By equation 2 and 3, the Hopfield network employs the basic structure of individual neurons as in the Perceptron or the Adaline. However, by Figure 5, it departs from the previous designs in its feedback structure. Note that a two neuron binary Hopfield network can be considered as a 2^n state system, with outputs belonging to the four states set {00, 01, 10, 11}. The network, when inputted by an input vector, will stabilize at one of the above states as determined by its weight configurations. A partially incorrect input vector may lead the network to the nearest state to the desired one (to the one related) to the correct input vector) [2].



key: Ni = ith neuron ; D= distribution node (external input x1 ... x3 also entered to D : no shown

F = activation function ; Ii= bias input ; wij = weight

Figure. 5 Structure of Hopfield Network

5.4.1 Algorithm of Hopfield network [2]

Let the weight matrix of the Hopfield network satisfy:

$$W = \sum_{i=1}^{L} (2\bar{x}_i - 1)(2\bar{x}_i - 1)^T \quad ...(4)$$

L = numbers of training sets:

The computation of the Hopfield network with BAM memory as in. (4) Above, will then proceed according to the following procedure.

(1) Assign weights w_{ij} of matrix W according to (4), with $w_{ii} = 0 \ \forall \ i$ and x_i being the training vectors of the network.

(2) Enter an unknown input pattern x and set:

$$\mathbf{y_i (0) = x_i} \quad ...(5)$$

where x_i is the ith element of vector x considered.

(3) Subsequently, iterate:

$$\mathbf{y_i (n+1) = } f_N \mathbf{[z_i (n)]} \quad ...(6)$$

where f_N denotes the activation function,

$$\begin{cases} 1 \ldots \forall z > Th \\ \end{cases}$$

$$f_N(z) = \begin{cases} \text{unchanged} \ldots \ \forall \ z = Th \\ -1 \ldots \ \forall z < Th \end{cases} \quad \ldots(7)$$

And where

$$z_i(n) = \sum_{i=1} w_i y_i (n) \quad \ldots(8)$$

n being all integers denoting the number of the iteration (n = 0; 1; 2; ...). Continue iteration until convergence is reached, namely, until changes in $y_i(n+1)$ as compared with $y_i(n)$ are below some low threshold value.

(4) The process is repeated for all elements of the unknown vector above by going back to Step (2) while choosing the next element until all elements of the vector have been so processed.

(5) As long as new (unknown) input vectors exist for a given problem, go to the next input vector x and return to Step (2) above.

The node outputs to which y(n) converge per each unknown input vector x represent the exemplar (training) vector which best represents (best matches) the unknown input. For each element of input of the Hopfield network there is an output. Hence, for a character recognition problem there are 132 inputs and 28 outputs.

6. Experimental Results (Arabic Characters recognized by Hopfield Artificial Neural Network)

After minimization of the number of features (from 336 features of Arabic characters to 112 features) by using GA, a one-layer Hopfield network is created to recognize 28 Arabic characters., it is trained with standard data sets 132-neuron input and 28 neuron output make the algorithm converge and it tested the network with a set of test data.

The Hopfield neural network is designed and applied to the present work by using Matlab function ***newhop*** , (see figure 5), where Neuron input Number = 132, neuron output number =28, weight Function = `rand', Create Hopfield recurrent network by using matlab function
net = newhop(T) ; where T represent input vector

The training data set applied to the Hopfield network is illustrated as follows:

trainingData(1).input = [1101100100110110 … 101];
trainingData(2).input = [1100000100110011 … 111];
trainingData(3).input = [1101101100000110 … 001];
.

.
.
trainingData(n).input = [1001101100000110 … 001];
Initial Weight is computed as follows :

(1) Get all training data vectors X_i, i = 1; 2 . . . L
(2) Compute the weight matrix $W=\Sigma X_i X_i^T$ I over L vectors
(3) Set $w_{ii} = 0$, for all i, where w_{ii} is the ith diagonal element of the weight matrix
(4) Assign the jth row vector of the weight matrix to the j-th neuron as its initial weights.

The test data set is generated by volunteers writing samples of handwritten document as seen in figure (6).

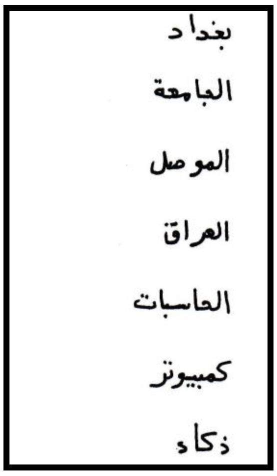

بغداد
الجامعة
الموصل
العراق
الحاسبات
كمبيوتر
ذكاء

Figure. 6 Volunteers Writing Samples of Handwritten Document

In order to train and test Hopfield neural network, we have used documents written by hand (scanned by scanner and convert to images with format .bmp) performed image conversion, segmentation, preprocessing, feature extraction operations on these documents. These documents are separated in the following way:

Training data set	: 1000 documents
Validation data set	: 500 documents
Testing data set	: 400 documents

The recognition rates by the Hopfield ANN classifier are 99 %, 92.13% and 90.52% on the training, validation and testing sets respectively. These results show the fact that this feature set achieves good recognition rates on database and feature selection method based on genetic algorithms used to recognized Handwritten Arabic characters reduces 66% of features.

7. Conclusion

In this research, a feature selection method based on genetic algorithms is used. The experiment shows that feature selection using the GA reduces features with good recognition accuracy, using GA during the feature selection process improves the recognition accuracy, and the feature selection is working effectively for features with larger dimension. The results obtained in this research were very promising and accuracy was as high as 99 %, 92.13% and 90.52% on the training, validation and testing sets by using Hopfield network, Hopfield network is robust with high convergence rate and has high success rate even if in the case of large errors.

REFERENCES

[1] Al-Nassiri, A.. (2005), Recognizing Isolated Handwritten Arabic Characters Using Hybrid Modified Directional Element Feature and General Autoassociative Memory, ACIT'2005, Proceedings of the 2005 International Arab Conference on Information Technology, December 6-8.Amman , Jordan.

[2] Daniel, G. (2007), Principle of Artificial Neural Network, World Scientific.

[3] Davis, L. (1991), Handbook on Genetic Algorithms, Van Nostrand, Reinhold

[4] Gonzalez, R. C. (2004), Digital Image Processing Using MATLAB, Pearson, Prentice Hall.

[5] Hartmut, P. (1997), Geatbx: Genetic Evalutionary Algorithms Toolbox for Use with Matlab.

[6] Heutte, L., Moreau J., Plessis B., Plagmaud J., and Lecourtier Y. (1993). Handwritten Numeral Recognition based on Multiple Feature Extractors. In 2* ICDAR, 167–170.

[7] Ho, S.Y. & Huang H.L. (2001), Facial Modeling From an Uncalibrated Face Image Using a Coarse-to-Fine Genetic Algorithm, Pattern Recognition, Vol. 34, No. 5.

[8] Kim, G. & Govindaraju V. (1997), A Lexicon Driven Approach to Handwritten Word Recognition for Real-Time Applications, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 4.

[9] Kim, G. &, Kim S. (2000), Feature Selection Using Genetic Algorithms for Handwritten Character Recognition. In 7th IWFHR Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition September 11-13, Amsterdam.

[10] Kim, B.S. & Song H.-J. (1998), An Efficient Preprocessing and Feature Extraction Method Based on Chain Code for Recognizing Handwritten Characters (in Korean), Journal of KISS(B): Software and Applications, Vol. 25, No. 12.

[11] Klassen, T. (2001), Towards Neural Network Recognition Of Handwritten Arabic Letters. Masters Thesis, Dalhousie University, Halifax, U.S.A.

[12] Madhvanath, S., Kim G., & Govindaraju V. (1999), Chain code Contour Processing for Handwritten Word Recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 9.

[13] Mitchell, M. (1996), An Introduction to Genetic Algorithms. MIT Press.

[14] Naili, Z. N., Ramadani, M., Bedda, M. (2005), Arabic Handwriting Recognition with Grapheme Nueral Networks.

[15] Oliveira ,L. S., Benahmed N. , Sabourin R. , Bortolozzi F. , Suen C. Y. (2000), Feature Subset Selection Using Genetic Algorithms for Handwritten Digit Recognition.

[16] Oliveira, L.S., Sabourin R., Bortolozzi F., and Suen C.Y. (2001), A modular system to recognize numerical amounts on brazilian bank cheques. in 6*, ICDAR, Seattle-USA, September.

[17] Oliveira, L. S., Sabourin R. (2003), A Methodology for Feature Selection Using Multiobjective Genetic Algorithms for Handwritten Digit String Recognition, International Journal of Pattern Recognition and Artificial intelligence Vol. 17, No. 6, 903-929.

[18] Ramesh, V.E. & Murty N.(1999), Off-line Signature Verification Using Genetically Optimized Weighted Features, Pattern Recognition, Vol. 32, No. 2.

[19] Shi, D. (1998), Feature Selection for Handwritten Chinese Character Recognition Based on Genetic Algorithms, International Conference of Systems, Man and Cybernetics.

[20] Yang, J. & Honavar V. (1998), Feature Subset Selection Using a Genetic Algorithm, IEEE Intelligent Systems, Vol. 13, No. 1.

[21] Yimei, D., Fumitaka, K.; Yasuji, M. (2000), Slant Estimation for Handwritten Words by Directionally Refined Chain Code, Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition, September 11-13 2000, Amsterdam.

<u>Appendix (A) : Median Filter [4]</u>

The median filter is normally used to reduce noise in an image, the median filter considers each pixel in the image in turn and looks at its nearby neighbors to decide whether it is representative of its surroundings or not. Instead of simply replacing the pixel value with the *mean* of neighboring pixel values, it replaces it with the *median* of those values. The median is calculated by first sorting all the pixel values from the surrounding neighborhood into numerical order and then replacing the pixel being considered with the middle pixel value. (If the neighborhood under consideration contains an even number of pixels, the average of the two middle pixel values is used.). The following Figure illustrates an example calculation.

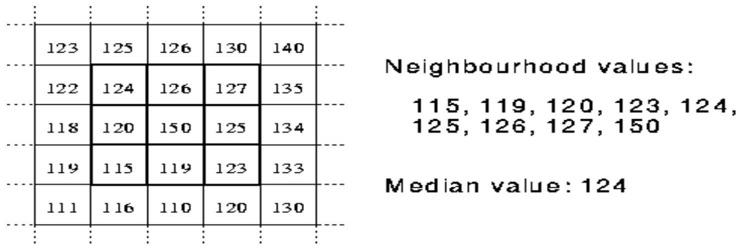


Figure Calculating the median value of a pixel neighborhood.

As can be seen, the central pixel value of 150 is rather unrepresentative of the surrounding pixels and is replaced with the median value: 124. A 3×3 square neighborhood is used here --- larger neighborhoods will produce more severe smoothing. The median filter has two main advantages over the mean filter: The median is a more robust average than the mean and so a single very unrepresentative pixel in a neighborhood will not affect the median value significantly. Since the median value must actually be the value of one of the pixels in the neighborhood, the median filter does not create new unrealistic pixel values when the filter straddles an edge. For this reason the median filter is much better at preserving sharp edges than the mean filter. In general, the median filter allows a great deal of high spatial frequency detail to pass while remaining very effective at removing noise on images where less than half of the pixels in a smoothing neighborhood have been affected. Median filter function in matlab :

```
y = medfilt1(x,n)
```

`where y = medfilt1(x,n)` applies an order `n` one-dimensional median filter to vector `x`; the function considers the signal to be 0 beyond the end points. Output `y` has the same length as `x`.

Appendix (B) : Roulette Wheel Selection with matlab [5]

NewChromIx = selrws(FitnV, Nsel, Dummy);

This function performs selection with Roulette wheel selection. This function selects a given number of individuals Nsel from a population. FitnV is a column vector containing the fitness values of the individuals in the population.

Syntax: NewChrIx = selrws(FitnV, Nsel)

Input parameters:

FitnV - Column vector containing the fitness values of the individuals in the population.

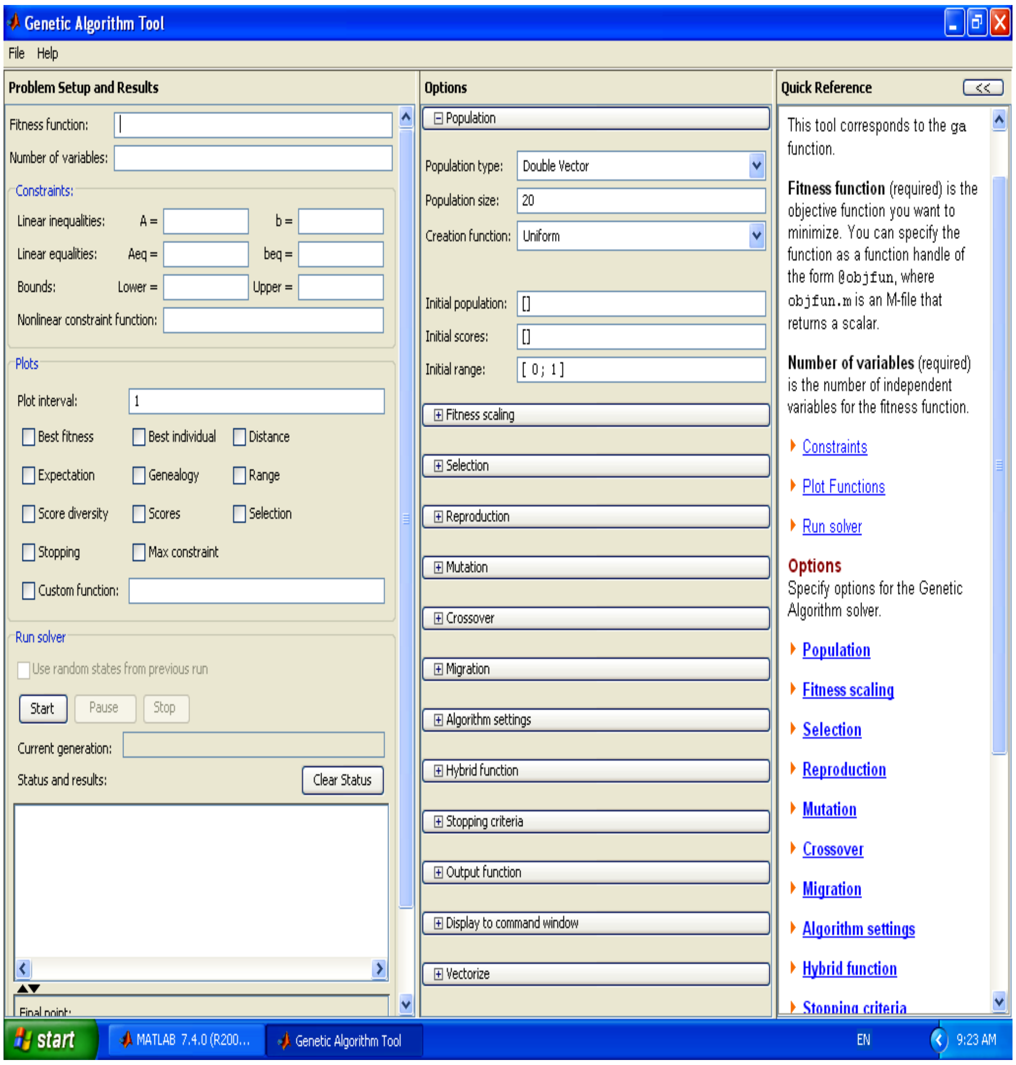
Nsel - Number of individuals to be selected

Output parameters:

NewChromIx- Column vector containing the indexes of the selected individuals relative to the original population, shuffeld. The new population, ready for mating, can be obtained by calculating OldChrom(NewChromIx,:).

Appendix (C)

gatool opens the Genetic Algorithm Tool, a graphical user interface (GUI) to the genetic algorithm, as shown in the figure below.



Fitness function:

Number of variables:

Constraints:

Linear inequalities: A = b =

Linear equalities: Aeq = beq =

Bounds: Lower = Upper =

Nonlinear constraint function:

Plots

Plot interval: 1

☐ Best fitness ☐ Best individual ☐ Distance

☐ Expectation ☐ Genealogy ☐ Range

☐ Score diversity ☐ Scores ☐ Selection

☐ Stopping ☐ Max constraint

☐ Custom function:

Run solver

☐ Use random states from previous run

Current generation:

Status and results:

Final point:

⊟ Population

Population type: Double Vector

Population size: 20

Creation function: Uniform

Initial population: []

Initial scores: []

Initial range: [0; 1]

⊞ Fitness scaling

⊞ Selection

⊞ Reproduction

⊞ Mutation

⊞ Crossover

⊞ Migration

⊞ Algorithm settings

⊞ Hybrid function

⊞ Stopping criteria

⊞ Output function

⊞ Display to command window

⊞ Vectorize

This tool corresponds to the `ga` function.

Fitness function (required) is the objective function you want to minimize. You can specify the function as a function handle of the form `@objfun`, where `objfun.m` is an M-file that returns a scalar.

Number of variables (required) is the number of independent variables for the fitness function.

- Constraints
- Plot Functions
- Run solver

Options
Specify options for the Genetic Algorithm solver.

- Population
- Fitness scaling
- Selection
- Reproduction
- Mutation
- Crossover
- Migration
- Algorithm settings
- Hybrid function
- Stopping criteria