Cellular Neural Networks Learning using Genetic Algorithm

E. Gómez-Ramírez & F. Mazzanti*
Laboratorio de Investigación y Desarrollo de Tecnología Avanzada
LIDETEA
UNIVERSIDAD LA SALLE,
Benjamín Franklin No. 47 Col. Condesa
CP 06140, México, D.F., México
egomez@ci.ulsa.mx

*Departament d'Electrónica, Enginyeria i Arquitectura La Salle, U. Ramon Llull,
Pg. Bonanova 8, 08022 Barcelona, España
mazzanti@salleURL.edu

Abstract

In this paper one alternative to the CNN learning using Genetic Algorithm is presented. The results show that it is possible to find different solutions for the cloning templates that fulfill the same objective condition. Different examples for image processing show the result of the proposed algorithm.

Keywords: Cellular Neural Networks, Learning, Genetic Algorithm, Image processing

1. Introduction

The Cellular Neural Networks [1,2] (CNN) have had a great development mainly on areas like image processing and pattern recognition. The most important reasons are due to its parallel nature and the simplicity of its algorithm. The parallel nature of the Artificial Neural Nets (ANN) has been employed by technologies like VLSI and optical computers [3].

It is possible to find many applications using fixed weights that come from some kind of experience, e.g., from the image processing area or from the behavior in the frequency domain. In many cases the learning problem of CNN has been solved for special and simple cases. In [4][5] the authors show a genetic algorithm to find only one of the two matrixes. In [6] the authors only show experiments for specific cases and it is possible that the examples presented only work with these constraints that off course reduce the space solution.

In this work a genetic algorithm approach is developed to compute all the parameters of a Cellular Neural Network (weights and current). The results show that it is possible to find many different matrixes that fulfill the objective condition. Different examples of image processing are presented to illustrate the procedure developed

The structure of the papers is the following: Section 2 describes the theory of Cellular Neural Networks, section 3 the genetic algorithm proposed and in section 4 the way that the algorithm was applied to find the optimal parameters for image processing.

2. CNN

From its origins in 1987 the CNN became a very powerful algorithm, and right now there are many research groups in the area. As it was defined on its origins, a CNN is an identical array of non-linear dynamic circuits. This array could be of different types depending on the way they are connected or the class of the neighborhood.

2.1 Mathematical Model

The simplified mathematical model that defines this array [7] is the following:

$$\frac{dx^c}{dt} = -x^c(t) + \sum_{d \in N_r(c)} a_d^c y^d(t) + \sum_{d \in N_r(c)} b_d^c u^d + i^c \qquad (1)$$

$$y^c(t) = \frac{1}{2}\left(\left|x^c(t)+1\right| - \left|x^c(t)-1\right|\right) \qquad (2)$$

where x^c represents the state of the cell c, y^c the output and u^c its input. The state of each cell is controlled by the inputs and outputs of the adjacent cells inside the r-neighborhood N(c). The outputs are feedback and multiplied by a^c_d and the inputs are multiplied by control parameters b^c_d. The value i^c is constant and is used for adjusting the threshold[1]. These coefficients are invariants to translations and it would be called cloning template.

The output y_c is obtained applying the equation (2). This equation limits the value of the output on the range between [-1,1], in the same way the activation function does in the classical ANN models (Fig. 1).

[1] This coefficient is equivalent to the input bias on other nets.

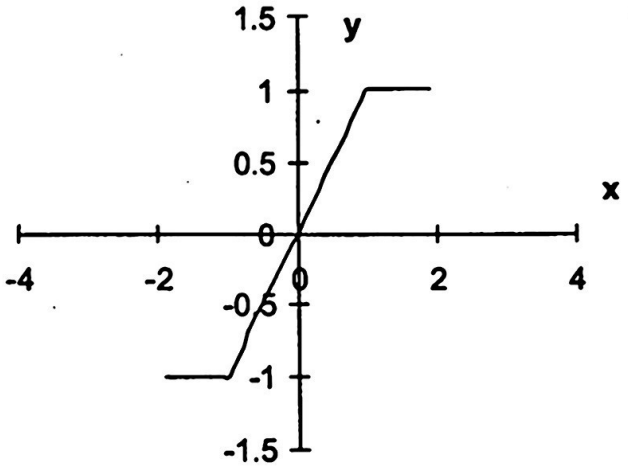

Figure 1 Output function

2.2 Discrete Mathematical Model

The mathematical model illustrated before could be represented by the following expression:

$$\frac{dx^c}{dt} = -x^c(t) + k^c \tag{3}$$

where:

$$k^c = \sum_{d \in N_r(c)} a_d^c y^d(t) + \sum_{d \in N_r(c)} b_d^c u^d + i^c$$

Because the final state is obtained evaluating x(∞), the model could be simplified as:

$$x^c(k) = \sum_{d \in N_r(c)} a_d^c y^d(k) + \sum_{d \in N_r(c)} b_d^c u^d + i^c \tag{4}$$

Some authors also considered a modification on the output (eq. 2), as follows [4]:

$$y^c(k) = f(x^c(k-1))$$

$$y^c(k) = \begin{cases} 1 & si \;\; x^c(k-1) > 0 \\ -1 & si \; x^c(k-1) < 0 \end{cases} \tag{5}$$

However, the simplification made (eq. 4) could be used with the function of the eq. 2 [8]. Equation 4 can be represented using the convolution operator like:

$$X_k = A * Y_k + B * U_k + I \tag{6}$$

where matrix A and B correspond to the rotation of 180° of the corresponding coefficients a^c_d and b^c_d. The previous step it is not necessary if the matrix has the property of $A=A^T$.

The cloning template in some cases, is a set of different combinations that fulfill the same conditions, e. g., there are different matrix that can be solve the same problem. This can be a problem with the use of gradient error techniques, because in these cases, it is not possible to find an optimal solution of the learning problem. One alternative is the use of evolutionary computation techniques that can be applied to find the different solutions or alternatives to the same problem.

The next section describes the methodology of Genetic Algorithm used in this work.

3. Genetic Algorithm

Genetic Algorithm (GA) is a knowledge model inspired on some of the evolution mechanisms that are observed in nature. GA usually follows the next cycle [9][10]:

- Generation of an initial population in a random way.
- Evaluation of the fitness or some objective function of every individual that belongs to the population.
- Creation of a new population by the execution of operations like crossover and mutation over individuals which fitness or profit value has been measured.
- Elimination of the former population and iterate using the new one until the termination criteria is fulfilled or it's reached a certain number of generations.

In this work the crossover or sexual recombination, the mutation and other special process that we call add parents [11] and add random parents are used. Next Sections describe these processes.

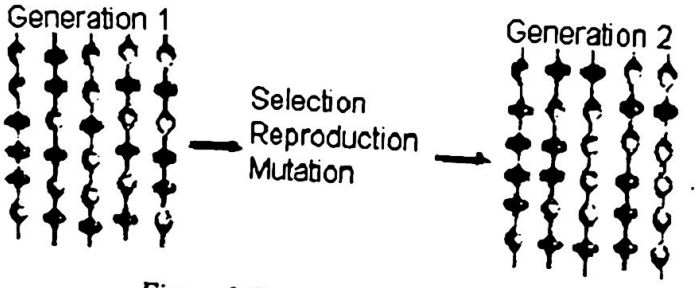

Figure 2 Operators of Genetic Algorithm

3.2 Crossover

The crossover operator is characterized by the combination of the genetic material of the individuals that are selected in function of the good performance (objective function).

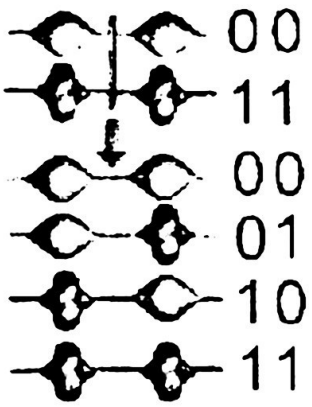

Figure 3 Multipoint Crossover Operator

To explain the multipoint crossover for each fixed number $g=1,...,n_g$, where n_g the number of total generations, let introduce the matrix F_g which is the set of parents of a given population. This matrix is boolean of dimension $F_g : n_p x n_b$, where n_p is the number of parents of the population at the generation g and n_b is the size of every array (chromosomes). Let $C(F_g, n_t)$ be the crossover operator which can be defined as the combination between the information of the parents set considering the number of intervals n_t of each individual and the number of sons n_s such that:

$$n_s = n_p^{n_t} \tag{7}$$

then $C(F_g, n_t): n_p x n_b \rightarrow n_s x n_b$. To show how the crossover operator can applied the following example is introduced. Let F_g has $n_p=2$ and $n_t=2$. This means that the array (the information of one father) is divided in 3 sections and every section is determined with a_i and b_i respectively for $i=1,...,n_t$. It's important appoint that with this operator the parents F_g of the population g are included the result of the crossover:

Example:

$$F_g = \begin{bmatrix} M_1 \\ M_2 \end{bmatrix}, M_1 = [a \quad b], M_2 = [c \quad d]$$

$$\Rightarrow C(F_g) = \begin{bmatrix} [a & b] \leftarrow \\ [a & d] \\ [c & b] \\ [c & d] \leftarrow \end{bmatrix}$$

Note that with this operator the parents F_g of the population g are included in the result of the crossover.

For the problem of CNN learning every section of one father can be represented by one column or one row of one matrix of the cloning template.

3.3 Mutation

The mutation operator just changes some bits that were selected in a random way from a fixed probability factor P_m; in other words, we just vary the components of some *genes*. This operator is extremely important, because assures the maintenance of the diversity inside the population, which is basic for the evolution. This operator $M : n_s x n_b \rightarrow n_s x n_b$ changes with probability P_m a specific population in the following way:

$$M(F_{ij}, P_m) = \begin{cases} \overline{F}_{ij} & r(\omega) \leq P_m \\ F_{ij} & r(\omega) > P_m \end{cases} \tag{8}$$

3.4 Add Parents

In this part the parents F_g are added to the result of mutation process, then the population A_g at the generation g can be obtained like:

$$A_g = \begin{bmatrix} M(C(F_g)) \\ F_g \end{bmatrix} \tag{9}$$

This step and the previous one ensure the convergence of the algorithm, because every generation at least hast the best individual obtained in the process.

With this step the original and mutated parents are included in the generation g and with some probability the individuals tend to the optimal one.

3.5 Selection Process

The Selection Process S_g computes the objective function O_g that represents a specific criterion to maximize or minimize and selects the best n_p individuals of A_g as:

$$S_g(A_g, n_p) = \min^{n_p} O_g(A_g) \tag{10}$$

Then, the parents of the next generation can be calculated by:

$$F_{g+1} = S_g(A_g, n_p) \tag{11}$$

In resume the Genetic Algorithm can be describe with the following steps:

1. For the initial condition $g=0$ select the A_0 in random way, such that $A_0 : n_s x n_b$
2. Compute $F_1 = S_0(A_0)$
3. Obtain A_g
4. Calculate S_g
5. Return to step 3 until the maximal number of generations is reached or one of the individuals of S_g obtain the minimal desire value of O_g

3.6 Add Random Parents

To avoid local minima a new scheme is introduced and it is called add random parents. If the best individual of one generation is the same than the previous one, a new random individual is included like parent of the next generation. This step increases the population because when crossover is applied with the new random parent, the number of sons increases. This step is tantamount to have a very big mutation probability and to search in new one points of the solution space.

4. Application in image processing

The procedure used to train the network was:

1. To generate the input patterns using a known matrix in image processing to find the different kind of lines (horizontal, vertical, diagonal) in different images.
2. To train the network using the methodology above explained using GA
3. Test the results using a new pattern

The results were repeated to obtain different templates for the same problem. The parameters used are: 4000 individuals in the initial population and 5 parents. The change in the values for every coefficient was discretized using an increment of 0.125 to reduce the space solution.

4.2 Example 1 Horizontal Line Detector

Figures 4 and 5 show the input and output patterns used. This patterns was obtained using a well known matrix:

$$M = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

The parameters obtained are:

$$A = \begin{bmatrix} -0.125 & 0 & 0 \\ 0 & 2.5 & 0.125 \\ -0.75 & -1.375 & 0.125 \end{bmatrix}, B = \begin{bmatrix} -0.125 & -2 & 0 \\ 0.375 & 2.75 & 0 \\ 0.375 & -0.875 & -0.25 \end{bmatrix}$$

$I=-2.875$

$$A = \begin{bmatrix} 4.5 & -2.25 & -0.5 \\ -1.625 & 0.75 & 3.375 \\ 2.125 & -0.875 & -0.875 \end{bmatrix}, B = \begin{bmatrix} 0.75 & -7.125 & -0.625 \\ 2.625 & -0.875 & 2.25 \\ -3.125 & -0.875 & 2.375 \end{bmatrix}$$

$I=0.625$

4.3 Example 2 Vertical Line Detector

Figures 6 and 7 show the input and output patterns used. This patterns was obtained using a well known matrix:

$$M = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

In this case we put constraints to select only the matrix B. The parameters obtained are:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & -0.5 & 0 \\ 0 & 0.625 & 0 \\ 0 & -0.125 & 0 \end{bmatrix}, I=0$$

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & -0.75 & 0 \\ 0 & 1.625 & 0 \\ 0 & -0.875 & 0 \end{bmatrix}, I=0$$

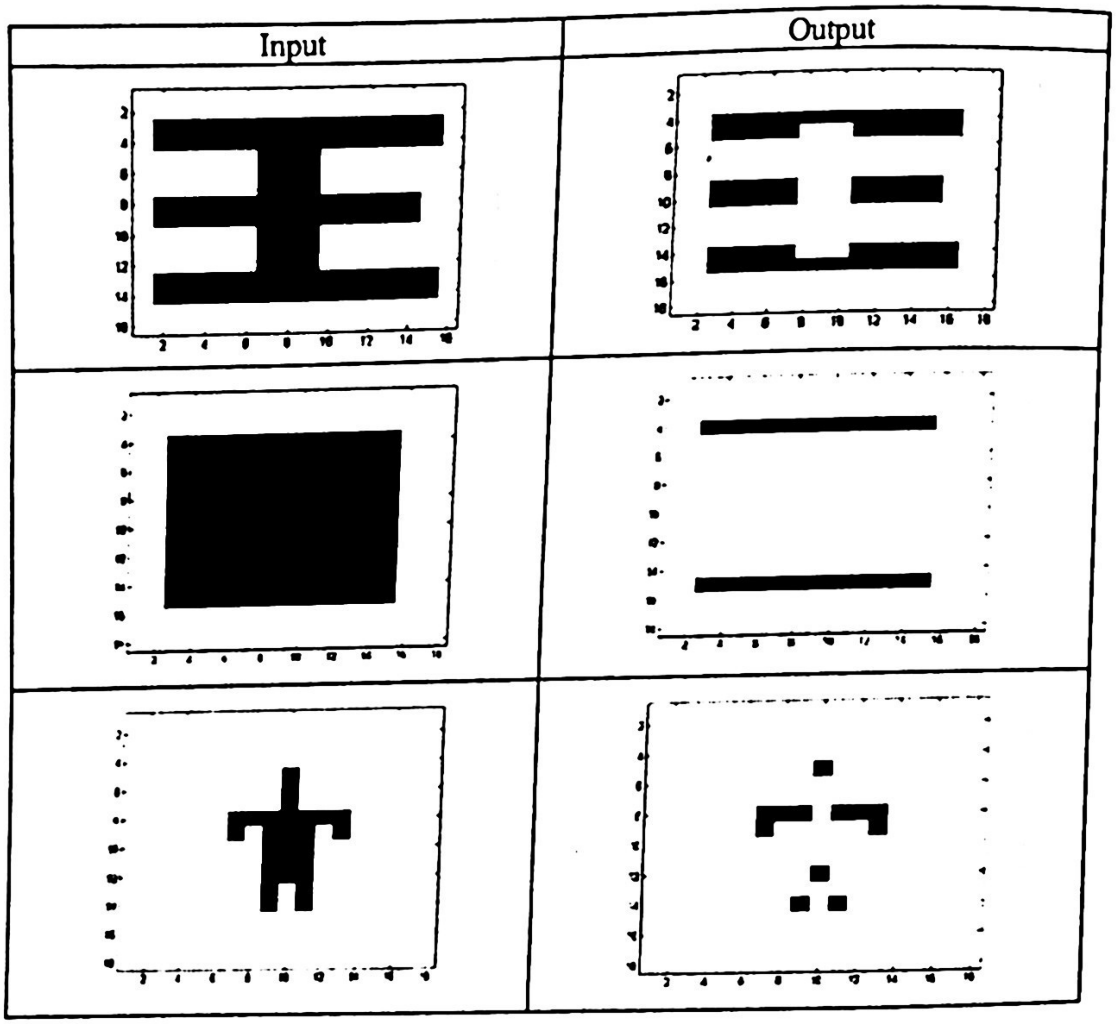

Figure 4 Input and Output patterns for CNN Learning

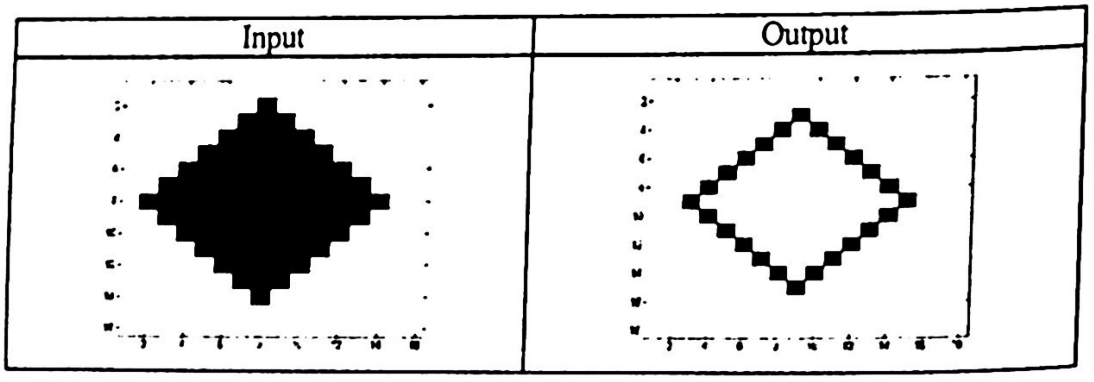

Figure 5 Input and Output Patterns for CNN Test

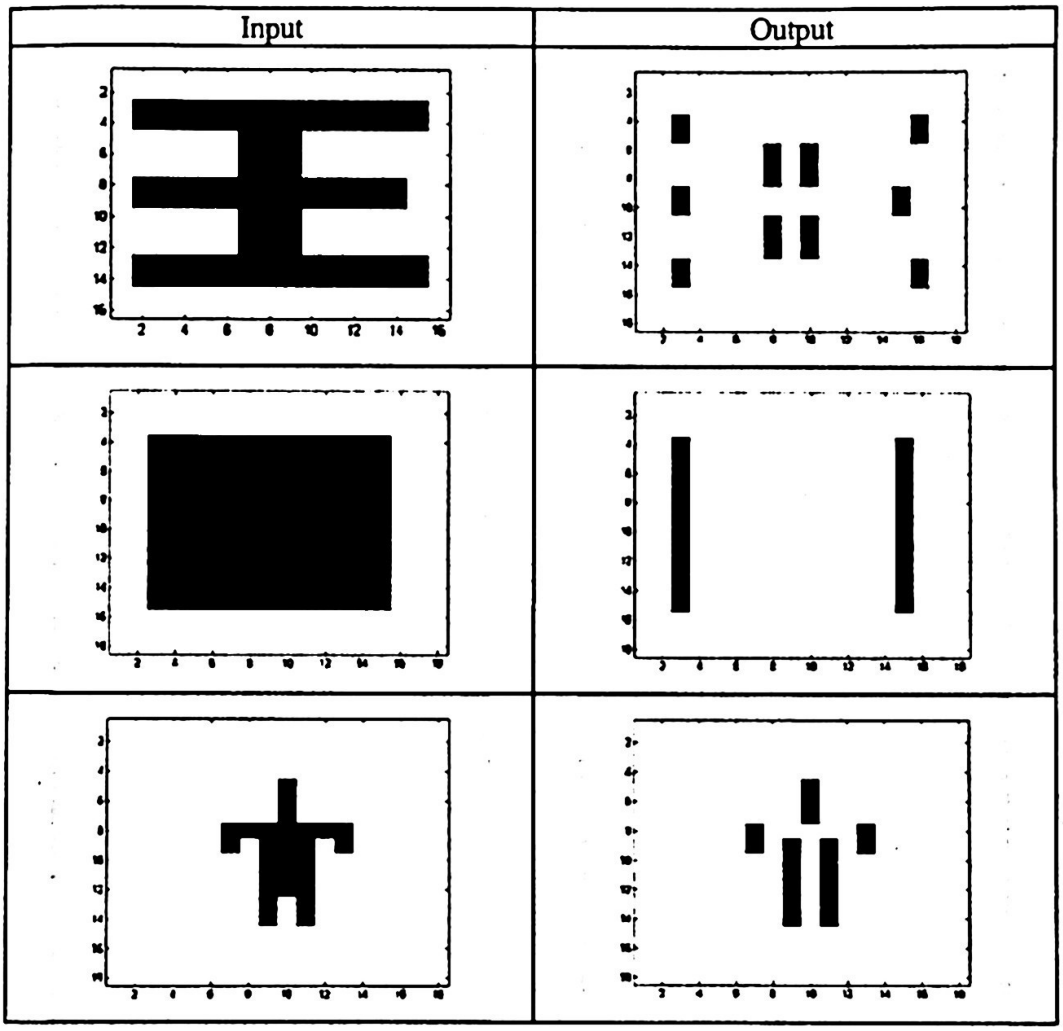

Figure 6 Input and Output patterns for CNN Learning

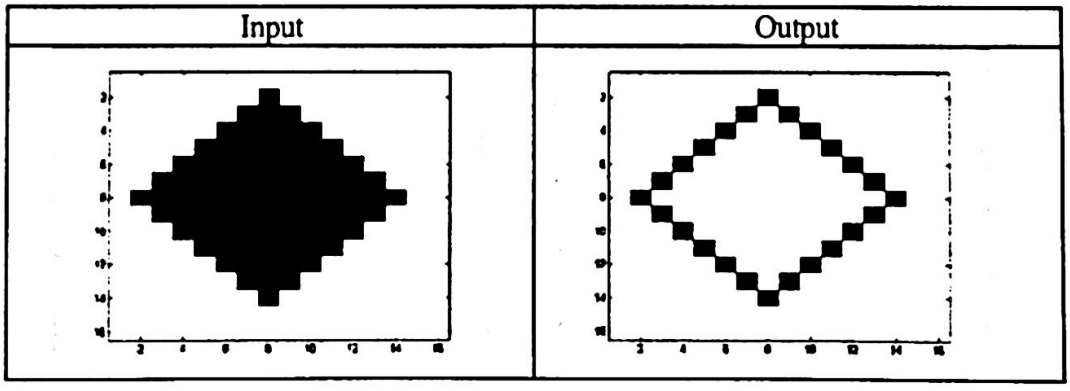

Figure 7 Input and Output Patterns for CNN Test

5. Conclusions

Genetic Algorithm is one alternative to the Learning of CNN that can be used to solve the problem of techniques that use the gradient error in the algorithm. In this work the use of add random parent avoid to reach local minima and improves the convergence of the algorithm. This is very important when the space solution includes all the elements of the cloning template.

6. References

[1] O. Chua & L. Yang (1988), "Cellular neural networks: Theory", *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1257-1272.

[2] O. Chua & L. Yang (1988),"Cellular neural networks: Applications", *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1273-1290.

[3] Mayol W, Gómez E. (1994), "2D Sparse Distruted Memory-Optical Neural Network for Pattern Recognition". *IEEE International Conference on Neural Networks*. Orlando, Florida, June28-July 2.

[4] Taraglio S. & Zanela A. (1996) , Cellular Neural Networks: a genetic algorithm for parameters optimization in artificial vision applications., *CNNA-96., Proceedings., 1996 Fourth IEEE International Workshop on*, 24-26 June 1996 , Seville, Spain, pp. 315-320.

[5] Destri, G., (1996), Discrete-time cellular neural network construction through evolution programs *Cellular Neural Networks and their Applications, CNNA-96., Proceedings, 1996 Fourth IEEE International Workshop on*, 24-26 June 1996 , Seville, Spain, pp. 473 – 478.

[6] Doan, M. D., Halgamuge, S., Glesner M. & Braunsforth, (1996), Application of Fuzzy, GA and Hybrid Methods to CNN Template Learning, *CNNA-96., Proceedings., 1996 Fourth IEEE International Workshop on*, 24-26 June 1996 , Seville, Spain pp. 327-332.

[7] Harrer H. & Nossek J., (1993), "Discrete-Time Cellular Neural Networks". *Cellular Neural Networks*. Edited by T. Roska and J. Vandewalle. John Wiley & Sons.

[8] M. Alencastre-Miranda, A. Flores-Méndez & E. Gómez Ramírez:, (1995) "Comparación entre Métodos Clásicos y Redes neuronales celulares para el análisis de imágenes". *XXXVIII Congreso Nacional de Física*. Zacatecas, Zacatecas, Mexico del 16 al 20 de octubre.

[9] Alander J. T., *An Indexed Bibliography of Genetic Algorithms*: Years 1957--1993, 1994, Art of CAD ltd.

[10] Bedner I., (1997), *Genetic Algorithms and Genetic Programming at Stanford* , Stanford Bookstore.

[11] E. Gómez-Ramírez, A. Poznyak, A. González Yunes & M. Avila-Alvarez. Adaptive Architecture of Polynomial Artificial Neural Network to Forecast Nonlinear Time Series. *CEC99 Special Session on Time Series Prediction.* Mayflower Hotel, Washington D.C., USA, July 6-9, 1999.