2020 17th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)
Mexico City, Mexico November 11-13, 2020

Proposal for training a Cellular Neural Network using a Hybrid Artificial Bee Colony and Nelder-Mead Algorithms

J.J. Morales-Romero[1], M.A. Reyes-Barranca[1], L.M. Flores-Nava[1], E.R. Espinosa-García[1]
[1]Department of Electrical Engineering, CINVESTAV-IPN
Mexico City, Mexico
Phone: (52) 55 5747 3800 Ext: 6261
E-mail: {jmoralesr, mreyes, lmflores, espinosa} @cinvestav.mx

***Abstract*—In this work, we propose a methodology for training a Cellular Neural Network based on the Artificial Bee Colony Algorithm and the Nelder-Mead Algorithm. The results of this proposal are compared with training using only the Artificial Bee Colony algorithm and with the Nelder-Mead algorithm, also with related works. The Cellular Neural Network is directed and trained for image processing, and the selected tasks for the performance evaluation of the Cellular Neural Network are edge detection and noise remover.**

***Index Terms*—Cellular Neural Network, Artificial Bee Colony, Nelder-Mead algorithm, image processing, edge detection, noise remover.**

I. INTRODUCTION

Chua and Yang on the year of 1998, proposed a new architecture of neural network called Cellular Neural Network (CNN) [1]. Since then, there have been several studies about the search and optimization of templates [2]–[4].

CNNs can be used for different tasks. Nevertheless, one appealing assignment is for image processing [5]. There are distinct image processing such as edge detection, noise remover, global connectivity detector, and shadow detector.

Some works, which use CNNs for image processing, can be found in the following: in [4] a CNN has been used for edge detection, in [6] a CNN has been used for video processing, and finally, in [7] a CNN has been used for edge detection in noisy images.

As was previously mentioned, there are several proposals to search the cloning templates of a CNN. These proposals can be divided into two categories: analytic methods [3], [8] and heuristic methods [4], [9], [10].

Heuristic methods are an interesting approach to optimization problems, and also they have been used for training neural networks. In heuristic methods, we can find Particle Swarms Optimization (PSO) [10], Genetic Algorithms (GA) [2], Artificial Bee Colony (ABC) [4], [11], and Nelder-Mead method [12].

Since Karaboga proposed the ABC algorithm [13], it has been used to carry out different optimization tasks. In the context of CNNs, the ABC algorithm has been proposed as a good alternative for searching cloning templates, as in [4] and [14]. However, there is no report on how to improve the results obtained with this method for searching CNN templates, and there is also no report on the use of this method to process other image processing tasks.

In this work, we present a modification of the ABC algorithm using the Nelder-Mead algorithm for training a CNN, which can process images for noise removal and as an edge detector, as well. The results obtained are then compared with other works, particularly compared with training CNNs using the ABC algorithm and Nelder-Mead methods individually.

The rest of the work is as follow: the theory of the CNN is presented in section II; the ABC and the Nelder-Mead algorithms are explained in section III; the proposed method is explained in section IV, and finally, in section V, the results of the proposed method and the comparisons with other works are presented.

II. CELLULAR NEURAL NETWORK

A CNN is built based on a unity called *cell*. This cell is cloned to construct a structure of one, two, or more dimensions. For instance, in image processing in grayscale, a two-dimensional array of cells is used. In Fig. 1, it is shown a CNN of two dimensions.

The cell connected directly to the cell $C(i,j)$ is called neighborhood $N_r(i,j)$. The *radius* r defines the set of cells in the neighborhood and is referred to as $(2r+1)\times(2r+1)$ for the case of a two-dimensional CNN.

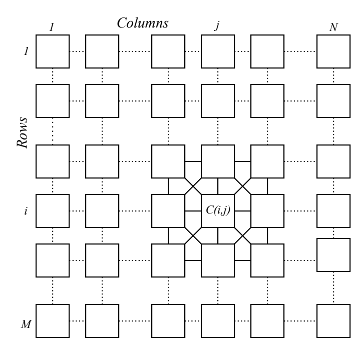


Fig. 1. Two-dimensional CNN.

The behavior of a cell within the CNN is defined in (1), which is called the state equation [1]. Equation (1) has been approximated by the Euler method.

$$v_{xij}(n+1) = v_{xij}(n) + h[-v_{xij}(n) + f_{ij}(n) + I_{ij}] \quad (1)$$

Where:

$$f_{ij}(t) = \sum_{C(k,l)\in N_r(i,j)} A(i,j;k,l)v_{ykl}(n)$$

and

$$I_{ij} = \sum_{C(k,l)\in N_r(i,j)} B(i,j;k,l)v_{ukl} + I$$

From the *state equation*, n is the integration step, h is the time step constant in the Euler method, and v_{xij} is the state of the cell $C(i,j)$.), $A(i,j;k,l)$ is called the feedback template and v_{ykl} are the outputs of the boundary cells in the neighborhood, $B(i,j;k,l)$ is called the control template, v_{ukl} is the input of the boundary cells and finally, I is the bias of the cell $C(i,j)$.

As in neural networks, an activation function defines the output of a cell. The activation function of a cell on CNN is the symmetric saturating linear transfer function (Satlins). In (2), the satlins function is defined.

$$v_{yij} = 0.5(|v_{xij}(n) + 1| - |v_{xij} - 1|) \quad (2)$$

The templates **A** and **B** with the bias I defines the task to be performed by the CNN. Considering a radius $r = 1$, we can define the templates as shown in (3) and (4).

$$\mathbf{A} = \begin{bmatrix} a_{-1,-1} & a_{-1,0} & a_{-1,1} \\ a_{0,-1} & a_{0,0} & a_{0,1} \\ a_{1,-1} & a_{1,0} & a_{1,1} \end{bmatrix} \quad (3)$$

$$\mathbf{B} = \begin{bmatrix} b_{-1,-1} & b_{-1,0} & b_{-1,1} \\ b_{0,-1} & b_{0,0} & b_{0,1} \\ b_{1,-1} & b_{1,0} & b_{1,1} \end{bmatrix} \quad (4)$$

However, considering the stability restrictions mentioned in [1] and [5], it is possible to redefine the templates as in (5).

$$\mathbf{A} = \begin{bmatrix} a_0 & a_1 & a_2 \\ a_3 & a_4 & a_3 \\ a_2 & a_1 & a_0 \end{bmatrix}; \mathbf{B} = \begin{bmatrix} b_0 & b_1 & b_2 \\ b_3 & b_4 & b_3 \\ b_2 & b_1 & b_0 \end{bmatrix} \quad (5)$$

III. Theory of Artificial Bee Colony and Nelder-Mead Algorithms

Both the ABC and the Nelder-Mead algorithms are considered heuristic methodologies. The following explains the theory of both the ABC algorithm and the Nelder-Mead algorithm (also known as the Downhill Simplex method).

A. Artificial Bee Colony Algorithm

The ABC algorithm belongs to the so-called bio-inspired algorithms because it is based on the foraging behavior of honeybees. In this algorithm, three groups of honeybees within a colony are defined: the first group is the employed bees, the second group is the onlooker bees and, finally, the third group is the scout bees. The behavior of each group of bees is explained below:

The Employed Bees (*EBs*), which are the first half of the colony, are sent to the sources, and they calculate their nectar amount. The Onlooker Bees (*OBs*), which are the second half of the colony, are sent to their sources and calculate their nectar amount. Finally, the Scout Bees (*SBs*), which represents the nectar sources, are sent for searching new sources, this is made based on a parameter call *Limit* (*L*) of the SB. Here, the nectar sources represent possible solutions to the optimization problem.

The *Limit* of the Scout Bees determines when a Scout Bee abandons its source, and it is sent to search for a new one. The general ABC algorithm is shown in Algorithm 1 [13].

Algorithm 1 ABC Algorithm

```
1: Assign control parameters.
2: Initialize the EBs, OBs, SBs, and the Limit parameters
3: Calculate the initial nectar amount of the EBs and memorize the best result.
4: while the stop criterion is not met do
5:     Send the EBs onto the food sources and calculate their nectar amount.
6:     Send the OBs around the food sources and calculate their nectar amount. Update the best result in applying a greedy selection.
7:     Send the SBs into the search are for new food sources, randomly. This is done according to the Limit value.
8:     Memorize the best food sources found so far.
9: end while
```

In Algorithm 1, the first step is assigned the parameters of the ABC algorithm. In the second step, the EBs, OBs, SBs, and the *Limit* value are initialized. In the third step, the first nectar amount of the EB is calculated, and it is saved the best solution of these bees. Step 4 is the main loop, here, it is made the steps for searching the solution of the problem using the different bees.

In the mathematical context, the colony size is made up of half of the EBs, and the other half is the OBs. The set of food sources, $(x_1, \cdots, x_{SB})$, is produced randomly using (6).

$$x_{i,j} = x_j^{min} + \phi(x_j^{max} - x_j^{min}) \quad (6)$$

Where $i = 1, \cdots, SB$; $j = 1, \cdots, D$; D is the dimension of the optimization problem; ϕ is a random number with uniform distribution in the interval $[0, 1]$; x_j^{min} and x_j^{max} are the boundaries of the search area.

The location of each EB is changed from the position of each SB to neighboring sources using (7).

$$v_{i,j} = x_{i,j} + \varphi(x_{i,j} - x_{k,l}) \tag{7}$$

Where $k = 1, \cdots, SB$ and $k \neq j$; j and k are indexes chosen randomly; φ is a random number with uniform distribution in the interval $[-1, 1]$. The value $v_(i, j)$ must be into the boundary of the search area.

The OB choose their food sources based on a probability distribution, this is defined in (8).

$$P_i = \frac{fit(x_i)}{\sum_{i=1}^{SB} fit(x_i)} \tag{8}$$

Where $fit(x_i)$ is the fitness function, and it is proportional to the amount source of x_i. The fitness function depends on the optimization problem. In this step, if the fitness value increases, the visiting probability of a new source is increased too. Here, it is applied to a greedy selection to memorize the OB.

During the phase of SBs, if an EB reaches its *Limit*, the food source is abandoned and begins a search of a new one using (6).

B. Nelder-Mead Algorithm

The Nelder-Mead algorithm, also known as the Downhill Simplex Method, belongs to heuristic methods [15]. As was mentioned before, this algorithm, as in the ABC algorithm, has been used to search cloning templates [12].

Before starting, we must initialize the algorithm with at least $n+1$ points. For example, in a two-dimensional problem, we need three points, in a three-dimensional problem we need four points, etc.

Next, the following points are defined: the first point which value gives the maximum (x_{h1}), the second point which value gives the second maximum (x_{h2}), the third point which value gives the minimum (x_{min}), and finally, an average of all points, also known as centroid (x_0) is calculated.

Finally, three operations are defined: a *reflection* which is shown in (9), an *expansion* which is shown in (10), and a *contraction* which is shown in (11).

$$x_r = x_0 + \alpha(x_0 - x_{h1}) \tag{9}$$

$$x_e = x_0 + \gamma(x_r - x_0) \tag{10}$$

$$x_c = x_0 + \beta(x_{h1} - x_0) \tag{11}$$

The values α, β, and γ are set heuristically. The algorithm for the Nelder-Mead method is shown in Fig. 2.

IV. The proposed method

ABC and Nelder-Mead algorithms are well suitable for minimization problems. ABC algorithm can search the global minimum, and the Nelder-Mead algorithm is fast to converge to a minimum. However, ABC requires many random points to be evaluated, and Nelder-Mead has the disadvantage of converging to a local minimum.

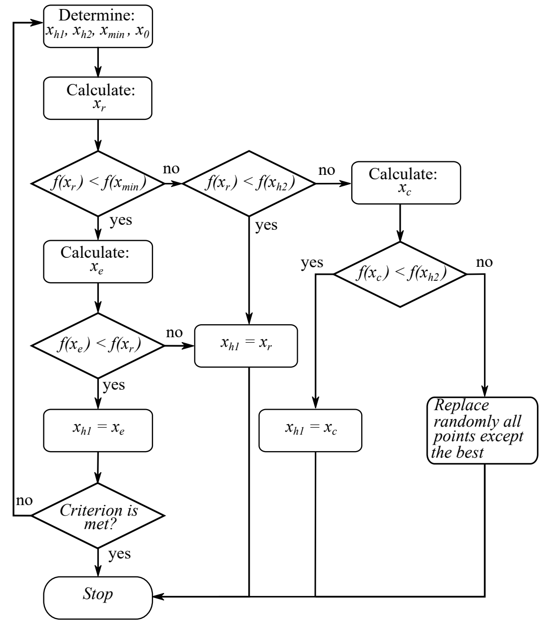


Fig. 2. Nelder-Mead Algorithm.

Considering the advantage of both algorithms, we can combine them to search a global minimum, overcoming the disadvantage of both algorithms.

The algorithm of the proposed method is shown in Algorithm 2. In such an algorithm, the phase of the OBs have been changed, instead of sending them into the area search randomly, they have been sent into the area search using the Nelder-Mead algorithm. This proposal increases the effectiveness of the searching for a minimum in training a CNN for image processing.

Algorithm 2 Hybrid Artificial Bee Colony and Nelder-Mead Algorithm

```
Assign the control parameters of ABC and Nelder-Mead.
Initialize the EBs, SBs, and the Limit parameters.
Calculate the initial nectar amount of the EBs and memorize the best result.
while the stop criterion is not met do
    Send the EBs onto the food sources and calculate their nectar amount.
    Apply a greedy selection.
    while OBs do
        Apply Nelder-Mead algorithm using the data for OBs.
        Apply a greedy selection.
    end while
    Send the SBs into the search are for new food sources, randomly. This is done according to the Limit value.
    Memorize the best food sources found so far.
end while
```

As it was noticed, in the Algorithm 2, the probability distribution used by the OBs has been avoided.

In the step of replacing randomly all points except the best in the Nelder-Mead algorithm, it has been changed by the best point found so far and it has been updated using (12), in the proposed method.

$$x_{h1,j} = x_{best} + rand(-1.0, 1.0) \tag{12}$$

To train a CNN for image processing, the parameters of the CNN, this is the values for the cloning template, have been treat as a vector. The vector used, considering (5) and the value of bias, is shown in (13).

$$x_i = [a_0, a_1, a_2, a_3, a_4, b_0, b_1, b_2, b_3, b_4, I] \tag{13}$$

The method proposed has been used to search cloning templates of a CNN for image processing. The block diagram used for training a CNN using the proposed method is shown in Fig. 3.

A correlation between the output image from the CNN and the ideal objective image is made using (14). The fitness value required in the optimization problem is calculated using (15).

$$C = \frac{\sum_i^M \sum_j^N (xP_{i,j} - \overline{xP})(yP_{i,j} - \overline{yP})}{\sqrt{(\sum_i^M \sum_j^N (xP_{i,j} - \overline{xP})^2)(\sum_i^M \sum_j^N (yP_{i,j} - \overline{yP})^2)}} \tag{14}$$

$$fit(C) = \begin{cases} 0.5 + |C| & C < 0 \\ \frac{1}{1+C} - 0.5 & C \geq 0 \end{cases} \tag{15}$$

In (14), $xP_{i,j}$ is the value of the pixel of the output image from CNN, $\overline{xP}$ is the mean of the output image from CNN, $yP_{i,j}$ is the value of the pixel of the objective image, and $\overline{yP}$ is the mean of the objective image. In (15), C is the correlation value obtain in (14).

V. Results

To show the effectiveness of the proposed method, CNN for image processing has been used. The tasks of the CNN are for edge detection and noise-remover.

A. CNN for Edge-detection

To train a CNN for edge-detection, an artificial image has been used as an input image. The parameters used in the method are the following: a colony size of 24 bees was used, where 12 bees were used for the EBs, 12 bees were used for the OBs, 12 bees were used for the SBs, the *Limit* parameter was set to 15, the values for the phase of the OBs were set to $\alpha = 1.0$, $\gamma = 2.0$ and $\beta = 0.5$.

The search area was set in the interval of $[-64.0, 64.0]$. As was mentioned before, the values remain within the search area. The set parameters for the CNN were the following, the time step, h, was set to $0.03125s$ and the maximum time for simulation was set to $1.5s$. One hundred cycles and five hundred cycles were set as a stop criterion. In Table I, the best values obtained with multiple runs for Nelder-Mead, ABC, and the proposed method are shown.

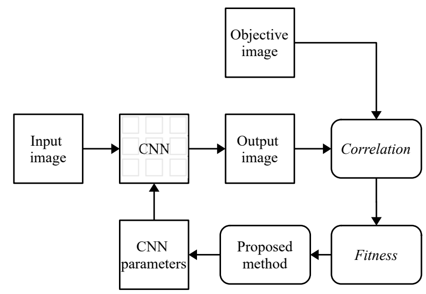


Fig. 3. Block diagram for training a CNN using the proposed method.

TABLE I
Performance of Nelder-Mead, ABC and Hybrid method for edge-detection

Cycles	Nelder-Mead		ABC		Proposed method	
	C	*fit*	*C*	*fit*	*C*	*fit*
100	0.6602	0.1023	0.6860	0.0931	0.8500	0.0405
500	0.8625	0.0368	0.7644	0.0667	**0.9006**	**0.0261**

The best cloning templates found have a correlation value of 0.9006 and a fitness value of 0.0261 and they are shown in vector form in (16):

$$\begin{aligned} x = [&0.1700, -1.6643, -2.2639, 1.6445, \\ &58.3330, 0.5411, -21.2155, 2.3303, \\ &-32.5841, 49.1672, -1.6426] \end{aligned} \tag{16}$$

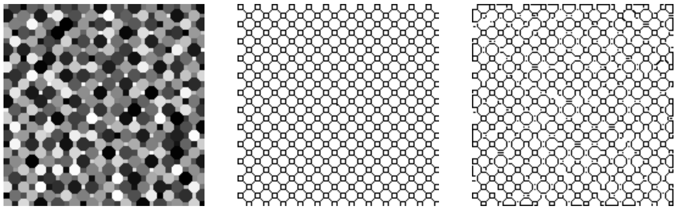
The input image used to train covered 57.81% of grayscale and size of 165×165 pixels. In Fig. 4, it is shown the input image, the objective image, and the output image processed by the CNN.

To validate the templates found, they were compared with the works [4], [5], [16], and [17]. The images obtained by the respective parameters of the different CNNs are shown in Fig. 5. The input and objective image used is shown in Fig. 4a and Fig. 4b, respectively.

In Fig. 5, it is shown the different results for an artificial image processing. Fig. 5a is the input image; Fig. 5b is the output image using the proposed method; Fig. 5c is the output image using the work [4]; Fig. 5d is the output image using the work [5]; Fig. 5e is the ouput image using the work [16]; and Fig. 5f is the output image using the work [17]. In Table II, it is shown the correlation obtained by each one of the works.

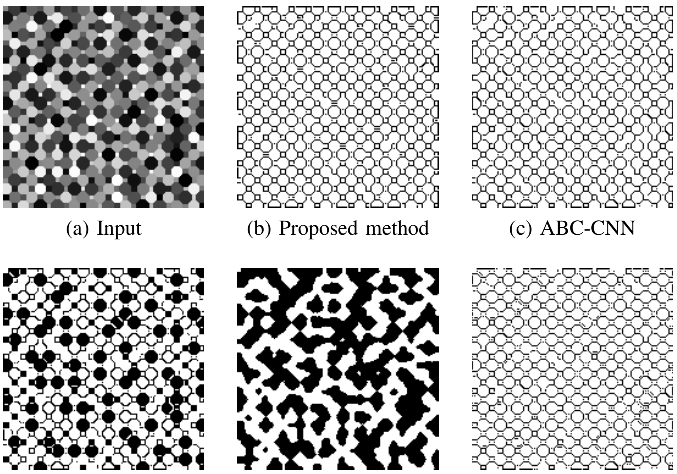
TABLE II
Comparative of correlation for diferents works

	ABC-CNN	Chua-CNN	DE-CNN	ES-CNN	Proposed method
C	0.8765	0.4805	0.0785	0.8375	**0.9006**



(a) Input (b) Objective (c) Output

Fig. 4. Images for edge detection.



(a) Input (b) Proposed method (c) ABC-CNN

(d) Chua-CNN (e) DE-CNN (f) ES-CNN

Fig. 5. Comparative of different works using an artificial image

In Fig. 6, it is shown the different results of processing a real image using the same parameters as in Fig. 5. The real image has a size of 384×384 pixels.

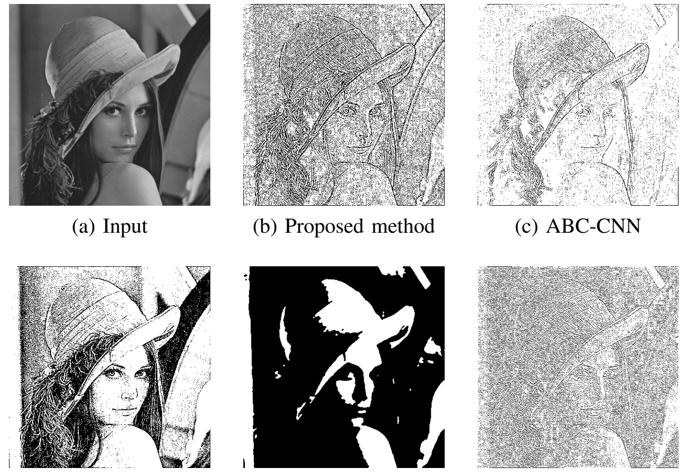
In Fig. 6a, it is shown the input image; in Fig. 6b, it is shown the output image using the proposed method; in Fig. 6c, it is shown the output image using ABC-CNN; in Fig. 6d, it is shown the output image using Chua-CNN; in Fig. 6e, it is shown the output image using DE-CNN, and finally, in Fig. 6f, it is shown the output image using ES-CNN.

B. CNN for Noise-remover

To train a CNN for noise remover, an artificial image has been used. The parameters for the proposed method were set equal as in the edge-detection subsection.

The search area was set in the interval of $[-64.0, 64.0]$. The parameters of the CNN were set as follows: time step, h, was set to $0.03125s$, the maximum time for simulation was set to $1.1s$. As in edge-detection training, one hundred and five hundred cycles were set as a stop criterion. In Table III, it is shown the best results obtained.

As can be seen in Table III, the best cloning templates found are for the proposed method. The best cloning templates for the task of noise-remover found are shown in vector form in (17).



(a) Input (b) Proposed method (c) ABC-CNN

(d) Chua-CNN (e) DE-CNN (f) ES-CNN

Fig. 6. Real image for edge-detection

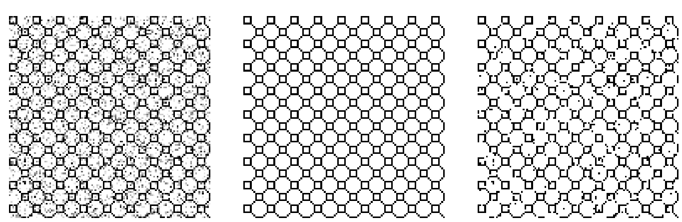
TABLE III
PERFORMANCE OF NELDER-MEAD, ABC AND HYBRID METHOD FOR NOISE REMOVER

Cycles	Nelder-Mead		ABC		Proposed method	
	C	*fit*	*C*	*fit*	*C*	*fit*
100	0.8616	0.0371	0.8691	0.0350	0.8976	0.0269
500	0.8982	0.0268	0.8535	0.0309	**0.9000**	**0.0263**

$$\begin{aligned} x = & [-0.1067, 20.3887, -21.4537, 12.5994 \\ & 54.3960, -1.0752, 2.6313, 27.0788, \\ & 3.4722, 59.0601, -18.9781] \end{aligned} \tag{17}$$

In Fig. 7, it is shown the input image, the objective, and the image processed by the CNN using the templates found. The input image has a size of 120×120 pixels and has a covered of 50% of noise. A grayscale noise was added to the image.

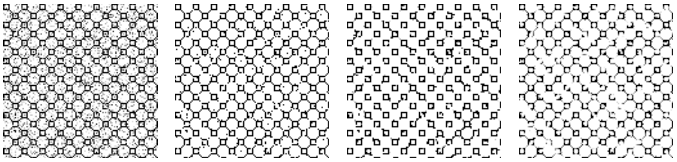
To validate the templates found, it has been compared with the works [12] and [18]. In Fig. 8, it is shown the output images for the validation.



(a) Input (b) Objective (c) Output

Fig. 7. Images for noise-remover.

In Fig. 8a, it is shown the artificial input image; in Fig. 8b, it is shown the output image using the proposed method; in 8c, it is shown the output image using the work [12]; and in



(a) Input (b) Proposed method (c) Nakai (d) Tsuruta

Fig. 8. Comparative of results of images for noise-remover.

Fig. 8d, it is shown the output image using the work [18]. In Table IV, it is shown the correlation of these images.

TABLE IV
PERFORMANCE OF THE PROPOSED METHOD WITH NAKAI AND TSURUTA WORKS

	Tsuruta-CNN	Nakai-CNN	Proposed method
C	0.8338	0.8788	**0.9000**

In Fig. 9, it is shown the different results of processing a real image with noise. The real image has a size of 300×300 pixels.

(a) Input (b) Proposed method (c) Nakai (d) Tsuruta

Fig. 9. Images for noise-remover.

In Fig. 9a,it is shown the input image; in Fig. 9b, it is shown the output image using the proposed method; in Fig. 9c, it is shown the output image using the Nakai-CNN; and in Fig. 9d, it is the output image using Tsuruta-CNN.

VI. Conclusion

As can be seen in the results section, the searching cloning templates found, for the tasks of edge-detection and noisy-images, were successfully achieved using the proposed methodology. Using the proposed method, for this search gives a better result than using the ABC and Nelder-Mead algorithm individually.

The quality of the image processed using the cloning templates found with the hybrid method is better compared with the quality of the image processed using the cloning templates found with ABC and Nelder-Mead algorithms alone, using the same parameters.

Compared to the results of this work with others, it can observe that the cloning templates found using the proposed method gives better results for artificial and real images.

Not only 100 and 500 cycles were used to train the CNN, but also 1000 and 2000 were used; however, it was observed that after 500 cycles, and the correlation did not increase its quality.

References

[1] Leon O. Chua and L. Yang, ”Cellular Neural Networks: Theory and Applications” IEEE Trans. on Circuits and Systems, vol. CA-35, pp. 1257-1290, 1988.

[2] T. Kozet, T. Roska and Leon O. Chua, “Genetic Algorithm for CNN Template Learning” IEEE Trans. on Circuits and Systems I, vol. 40, pp. 392-402, June 1993.

[3] M. Hanggi and G. S. Moschytz, “An exact and direct analytical method for the design of optimally robust CNN templates”, in IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 46, no. 2, pp. 304-311, Feb. 1999.

[4] Selami Parmaksızoğlu and Mustafa Alçı, “A Novel Cloning Template Designing Method by Using an Artificial Bee Colony Algorithm for Edge Detection of CNN Based Imaging Sensors”, Sensors, pp. 5337-5359, 2011.

[5] Leon O. Chua and Tamás Roska, “Cellular neural networks and visual computing: Foundations and Applications”, Cambridge U.K., Cambridge University Press, pp. 7-34, 2004.

[6] K. Kayaer and V. Tavsanoglu, “A new approach to emulate CNN on FPGAs for real-time video processing”, The 11th International Workshop on Cellular Nanoscale Networks and their Applications, 2008.

[7] V. Murugesh, V. Arthy, and V. Agalya, ”Edge detection of noisy images based on time-multiplexing CNN simulator,” Fifth International Conference on Computing, Communications and Networking Technologies ICCCNT), Hefei, pp. 1-5, 2014.

[8] L. O. Chua and P. Thiran, “An analytic method for designing simple cellular neural networks”, in IEEE Transactions on Circuits and Systems, vol. 38, no. 11, pp. 1332-1341, Nov. 1991.

[9] C. -. Chang, K. -. Fan, I. -. Chung and C. -. Lin, ”A Recurrent Fuzzy Coupled Cellular Neural Network System With Automatic Structure and Template Learning,” in IEEE Transactions on Circuits and Systems I: Express Briefs, vol. 53, no. 8, pp. 602-606, Aug. 2006.

[10] A. Giaquinto and G. Fornarelli, ”PSO-Based Cloning Template Design for CNN Associative Memories, in IEEE Transactions on Neural Networks, vol. 20, no. 11, pp. 1837-1841, Nov. 2009.

[11] J. Enriquez-Gaytan, F. Gómez-Castañeda, J. A. Moreno-Cadenas, and L. M. Flores-Nava, ”Spiking Neural network trained by metaheuristics for gas sensing,” 2017 International Conference on Electronics, Communications and Computers (CONIELECOMP), Cholula, pp. 1-6, 2017.

[12] Nakai, K. and Ushida, A. “Design technique of cellular neural network”, Electron. Comm. Jpn. Pt II, vol. 78, pp. 97-107, 1995.

[13] Karaboga, D. “An Idea Based On Honey Bee Swarm for Numerical Optimization”, Technical Report - F06; Computer Engineering Department, Erciyes University: Kayseri, Turkey, 2005.

[14] M. Duraisamy and F. M. M. Jane, ”cellular neural network-based medical image segmentation using Artificial bee colony algorithm,” 2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE), Coimbatore, pp. 1-6, 2014.

[15] J. A. Nelder, R. Mead, “A Simplex Method for Function Minimization”, The Computer Journal, Vol.7, Issue 4, pp. 308-313, Jan 1965.

[16] Basturk A., Gunay E., “Efficient Edge Detection in Digital Images Using a Cellular Neural Network Optimized by Differential Evolution Algorithm”, Expert Syst. Appl., pp. 2645-2650, 2009.

[17] Parmaksizoglu S., Alci M., “Designing CNN Templates for Edge Detection by Using Evolution Strategies”, In Proceedings of International Symposium on Innovations in Intelligent Systems and Applications, Kayseri, Turkey, pp. 396-399, June 2010.

[18] K. Tsuruta, Z. Yang, Y. Nishio, A. Ushida, “Small-World Cellular Networks for Image Processing Applications”, ECCTD – European Conference on Circuit Theory and Design, pp. 225 – 228, September 2003.