Single Spiking Neuron Multi-Objective Optimization for Pattern Classification

Submitted: 20th December 2019; accepted: 30th March 2020

Carlos Juarez-Santini, Manuel Ornelas-Rodriguez, Jorge Alberto Soria-Alcaraz, Alfonso Rojas-Domínguez, Hector J. Puga-Soberanes, Andrés Espinal, Horacio Rostro-Gonzalez

DOI: 10.14313/JAMRIS/1-2020/9

Abstract: *As neuron models become more plausible, fewer computing units may be required to solve some problems; such as static pattern classification. Herein, this problem is solved by using a single spiking neuron with rate coding scheme. The spiking neuron is trained by a variant of Multi-objective Particle Swarm Optimization algorithm known as OMOPSO. There were carried out two kind of experiments: the first one deals with neuron trained by maximizing the inter distance of mean firing rates among classes and minimizing standard deviation of the intra firing rate of each class; the second one deals with dimension reduction of input vector besides of neuron training. The results of two kind of experiments are statistically analyzed and compared again a Mono-objective optimization version which uses a fitness function as a weighted sum of objectives.*

Keywords: *Multi-objective Optimization, Spiking Neuron, Pattern Classification*

1. Introduction

Artificial Neural Networks (ANNs) try to simulate the behavior of the brain when they generate, process or transform information. An ANN is a system formed of simple processing units, which offers the property, and capability of input-output mapping. ANNs learn to solve complex problems in a reasonable amount of time [1]. The ability of learning of ANNs become a powerful tool for wide applications, for instance: pattern recognition works, classifying, clustering, vision tasks and forecasting [2].

ANNs can be distinguished in three generations according to their computational units [3]. The first one is based on McCulloch-Pitts neuron as computational units that can handle digital data [3]. The second one is characterized by a multilayer architecture, connectivity separating input, intermediate, and output units and applying activation functions with a continuous set of possible output values to a weighted sum of the inputs [4]. The third generation has been developed with the purpose of design neural models more plausible to the biological neurons. These are known as Spiking Neural Networks (SNNs) [5], [6].

ANNs are conformed by neurons organized in input, hidden and output layers, which are inter-connected by synaptic weights. These simulate the neuron synapsis of the human brain. During the training process of an ANN, a set of synaptic weights constantly is changing until the knowledge acquired is enough. Once the knowledge process has finished, it is necessary to evaluate the performance of the ANN. It is expected that the ANN can classify with acceptable accuracy the patterns from a particular problem during the testing phase [7]. The training process is an optimization task since it is desired to find the optimal weight set of the ANN. Methods based on gradient-descent have been applied to the training phase [8], but these techniques can be trapped at local minima. Then to overcome this situation, the researchers have proposed different global optimization methods [9] to optimize the ANNs by Evolutionary Algorithms (EAs). These EAs can be used to calibrate the connection weights, optimize the architecture and selecting the input features of ANNs [10].

The present research proposes a method for training full and partially connected SNNs based on the Leaky Integrate and Fire (LIF) model, by using a variant of Multi-objective Particle Swarm Optimization known as OMOPSO. This methodology is designed to solve pattern recognition problems. The results are statistically analyzed and compared with a version of mono-objective optimization using the Particle Swarm Optimization algorithm (PSO).

This paper is organized as follows: Section 2 presents the theoretical fundamentals used in this work. Section 3 explains the implemented methodology. Section 4 shows the results and statistical analysis. Finally, in section 5 are presented the conclusions and future work.

2. Background

This section describes the LIF model and the Optimized Multi-objective Particle Swarm Optimization (OMOPSO), which were used in this work.

2.1. Leaky Integrate and Fire Model

The LIF neuron model is one of the most used in the field of computational neuroscience given this model has an easier implementation and a lower computational cost in comparison with other spiking neuron models [11].

The mathematical representation for this model is shown in [11], [12] and it is given by the potential dynamic of the membrane:

$$\tau \frac{dv_i}{dt} = -g_{leak}\left(v_i - E_{leak}\right) + I(t) \quad (1)$$

where g_{leak} and E_{leak} are the conductance and the reversal potential of the leak current, τ is the membrane time constant and $I(t)$ is a current injected into the neuron.

In this work, it was used the representation proposed in [11],[13] defined as:

$$v' = I + a - bv,$$
$$\text{if } v \geq v_{threshold}\text{, then } v \leftarrow c \quad (2)$$

where I is the input current of the neuron, v denotes the membrane potential, a and b are parameters to configure the behavior of the neuron, c is the rest state voltage and $v_{threshold}$ is the threshold for the spike (firing) of the neuron. Besides, an initial condition v_0 is necessary to solve the differential equation by numerical methods.

Since the input patterns cannot be directly processed by the LIF neuron, they must be transformed to input currents by means of the equation:

$$I = \bar{x} \cdot \bar{w} \cdot \theta \quad (3)$$

where $\bar{x} \in \mathbb{R}^n$ is the input pattern vector, $\bar{w} \in \mathbb{R}^n$ is the set of synaptic weights and θ is a gain factor.

Fig. 1 shows the representation of a LIF neuron. When I is computed, it continues to solve the equation (2) to obtain the output spike train belonging to the input pattern.

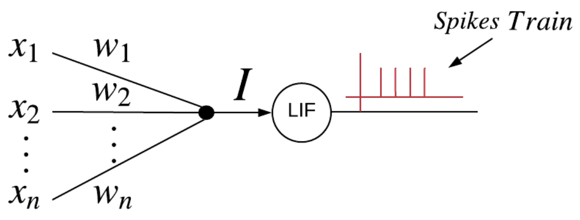

Fig. 1. Representation of a LIF neuron

2.2. Optimized Multi-Objective Particle Swarm Optimization (OMOPSO)

Regarding multi-objective optimization, a considerable number of algorithms can be found in the literature. For instance, the Multi-objective Particle Swarm (MOPSO) was proposed by Coello in [14]. In this work, we used the OMOPSO algorithm described in [15], which is based on Pareto dominance and an elitist selection through crowding factor. Beside this, the authors incorporated two mutation operators (uniform mutation and non-uniform mutation). The uniform mutation refers to variability range allowed for each decision variable, which is kept constant over generations and the non-uniform mutation has a characteristic variability range allowed for each decision variable, which decreases over time. Finally, it was added the ε-dominance concept which is the final size of the external file where stores the non-dominated solutions. Algorithm 1 shows the OMOPSO.

Algorithm 1. OMOPSO

```
Require: Initialize Swarm $P_i$, Initialize Leaders $L_i$
1:  Send $L_i$ to $\varepsilon$-file
2:  crowding($L_i$), $g = 0$
3:  while $g < g_{max}$ do
4:    for each particle
5:      Select leader
6:      Fly
7:      Mutation
8:      Evaluate
9:      Update $p_{best}$
10:   end for
11:   Update $L_i$
12:   Send $L_i$ to $\varepsilon$-file
13:   crowding($L_i$), $g = g + 1$
14: end while
15: Report results in $\varepsilon$-file
```

3. Methodology

This section shows the methodology used in our work. There were proposed two kinds of experiments: the first one treats with neuron trained by maximizing the inter distance of mean firing rates among classes and minimizing the standard deviation of the intra firing rate of each class; the second one deals with dimension reduction of input vector besides of neuron training.

The LIF neuron model was implemented into jMetal [16], [17] where is available the OMOPSO algorithm, which was used for training the LIF neuron. Furthermore, the OMOPSO algorithm was configured as a mono-objective algorithm (PSO).

The design of the methodology is shown in Fig. 2. Initially, we set up the parameters of the OMOPSO algorithm and the LIF neuron model. Next, it is necessary to initialize the particles and Leaders (L_i) with uniformly random numbers to make a swarm. Each particle represents a synaptic weight vector ($\bar{w}$) with the same size as the feature input vector ($\bar{x}$). Then, whole particles are evaluated into the LIF neuron model, by means of the objective functions. The non-dominated particles in the swarm will be L_i, which are sent to *ε-file*. Besides this, it is calculated a crowding factor for each L_i as a second discrimination criterion.

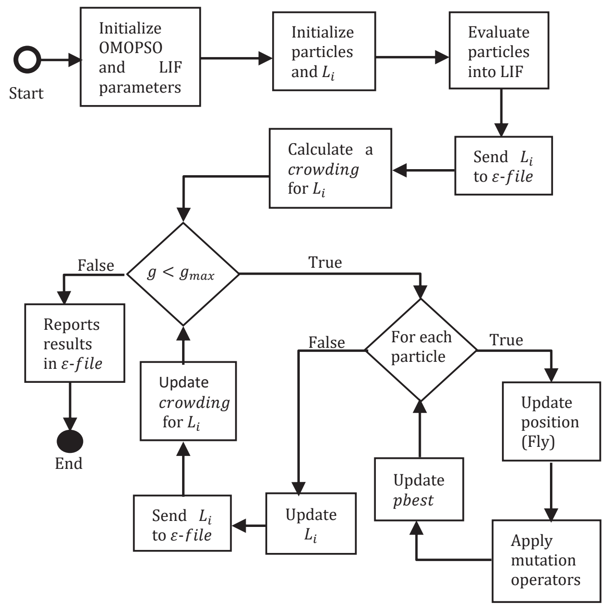
Start
Initialize OMOPSO and LIF parameters
Initialize particles and L_i
Evaluate particles into LIF
Send L_i to ε-$file$
Calculate a $crowding$ for L_i
False
$g < g_{max}$
True
Reports results in ε-$file$
End
Update $crowding$ for L_i
False
For each particle
True
Update position (Fly)
Update $pbest$
Send L_i to ε-$file$
Update L_i
Apply mutation operators

Fig. 2. Methodology schema

After it is initialized an Internal Loop into an External Loop, and each particle is modified into the Internal Loop, updating the position and applying the mutation operators. Then, each particle is evaluated and updated its personal best value (p_{best}). A new particle replaces the p_{best} if such value is dominated by the new particle or if both are non-dominated concerning each other.

When all particles have been updated, the L_i are modified in the External Loop. Only the particles that overcome their p_{best} will try to enter to L_i set. Once the L_i have been updated, they are sent to *ε-file*. Finally, the crowding values of the set of L_i is updated and we eliminate as many leaders as necessary to avoid overflow of the size of the L_i set. The process is repeated until finalizing all iterations.

3.1. Objective Functions

Three different objective functions were considered to measure the performance of the solutions (particles):

A. The Euclidean distance between the combination of AFR_i and AFR_j, where AFR is the average firing rate of each class and $i \neq j$. For this objective function, we looking for maximize the separability between the classes:

$$MAXdist\left(AFR_i, AFR_j\right) \tag{4}$$

B. The Standard Deviation of the firing rate for each pattern class $SDFR_k$, where k = 1, ..., K and K is the total of pattern classes. In this objective function, we looking for minimize the dispersion of each pattern class:

$$MIN\left(SDFR_k\right) \tag{5}$$

C. The dimension of the input feature vector ($\bar{x}$). To avoid redundancies in information, we desire to reduce the dimensionality of the feature vectors, by minimizing the total of 1's of a binary mask a binary mask ($\bar{r}$) with the same size of the input feature vector.

In our proposal, the number of objective functions is related to the number of classes of the dataset.

3.2. Experiments

Four supervised classification datasets from the UCI Machine Learning Repository [18] were employed for experimentation: Iris Plant, Wine, Glass, and SPECT. Table 1 shows the details of the datasets used.

Each dataset was randomly divided in two subsets with approximately the same size. The first one was employed as training set and the second one as testing set.

Tab. 1. Datasets employed for experimentation

Dataset	Instances	Classes	Features
Iris Plant	150	3	4
Wine	178	3	13
Glass	214	6	9
SPECT	267	2	22

With the aim to observe the performance of our proposal, four experiments were configurated according to the objective functions seen in section 3.1. The characteristics of each experiment are defined below and summarized in Table 2.

i. Experiment #1 was defined as a multi-objective problem, focusing on the A and B objective functions. The OMOPSO algorithm was used to optimize the synaptic weight vector of the LIF neuron.
ii. Experiment #2 employs the multi-objective approach, considering the A, B and C objective functions. The OMOPSO algorithm was taken to optimize the synaptic weight vector and the dimension of the input vector. Concerning the optimization of the last parameter, a binary mask ($\bar{r}$) was used in equation (3) to calculate a modified input current given by equation (6).

$$I = \bar{x} \cdot \bar{w} \cdot \bar{r} \cdot \theta \tag{6}$$

iii. Experiment #3 was designed as a mono-objective problem. The objective function (eq. 7) was formed by the weighted sum of two objective functions. The first one is the inverse of the summation of the Euclidean distances among all combinations of AFR_i and AFR_j and the second objective is the sum of the standard deviation of the firing rate for all classes as shown in equation 7 [11]. PSO algorithm was used to design the synaptic weight vectors.

$$MIN(f) = \frac{1}{dist(\boldsymbol{AFR})} + \sum_{k=1}^{K} SDFR_k \quad (7)$$

iv. Experiment #4 is a mono-objective approach that seeks to optimize the synaptic weight vector and the dimension of the input vector with the PSO algorithm. The objective function (eq. 8) is formed by the weighted sum of the equation (7) and the rate of T and D, where T is total of 1's in the binary mask $(\overline{r})$ and D is the dimension of the input feature vector.

$$MIN(f) = \frac{1}{dist(AFR)} + \sum_{k=1}^{K} SDFR_k + \frac{\boldsymbol{T}}{\boldsymbol{D}} \quad (8)$$

Tab. 2. Configuration for experimentation

	Algorithm	**Optimized Parameters**	**Objective Functions**
Exp #1	OMOPSO	synaptic weight vector	A, B
Exp #2	OMOPSO	synaptic weight vector and dimension of input vectors	A, B, C
Exp #3	PSO	synaptic weight vector	A, B
Exp #4	PSO	synaptic weight vector and dimension of input vectors	A, B, C

Table 3 shows a compendium of the number of objective functions by experiment for each dataset.

Tab. 3. Total of Objective Functions by experiment

		Objective Functions in			
Dataset	Classes	Exp #1	Exp #2	Exp #3	Exp #4
Iris Plant	3	6	7	1	1
Wine	3	6	7	1	1
Glass	6	21	22	1	1
SPECT	2	3	4	1	1

Each experiment consisted of 40 independently executions per each dataset to guarantee statistical significance. The parameter values used in the OMOPSO algorithm and the LIF neuron model [11] are detailed in Table 4 and 5 respectively.

The initial synaptic weights were generated randomly $\theta \in [0,1]$.

4. Results and Statistical Analysis

This section describes the results obtained from the experimentation proposed in section 3. The results are statistically analyzed and discussed below.

Tab. 4. Configuration OMOPSO Parameters

Max particle size:	100
Max iterations:	1000
ε-file size:	100
Uniform Mutation	
Mutation probability:	$\frac{1.0}{\textit{Number of problem variables}}$
Perturbation index:	0.5
Non-uniform Mutation	
Mutation probability:	$\frac{1.0}{\textit{Number of problem variables}}$
Perturbation index:	0.5
Max iterations:	1000

Tab. 5. Configuration LIF Parameters

a	0.5
b	-0.001
c	-50 mV
v_i	-60 mV
$v_{threshold}$	50 mV
Time	1000 ms
h	1
θ	0.1

For each execution, at the end of the training phase, the total of particles is evaluated in the LIF neuron model using the training set, and the classification accuracy is calculated for each particle. Finally, the particle with the best performance is used in the testing phase for obtaining the accuracy in the testing set.

Tab. 6. Accuracy of training phase over each experiment

	OMOPSO		**PSO**	
	Experiments		**Experiments**	
Dataset	#1	#2	#3	#4
Iris Plant	**0.9817 ± 0.0131**	0.9793 ± 0.0157	0.9 ± 0.0232	0.8987 ± 0.0219
Wine	0.7858 ± 0.0358	**0.8048 ± 0.0336**	0.6849 ± 0.0432	0.6986 ± 0.0301
Glass	**0.5050 ± 0.0441**	0.5031 ± 0.0405	0.39 ± 0.0030	0.3638 ± 0.0726
SPECT	**0.8592 ± 0.0243**	0.8286 ± 0.0227	0.7276 ± 0.0325	0.7273 ± 0.0344

Tables 6 and 7 show the results obtained from the methodology proposed. The accuracy values along with the standard deviations grade the performance of the experiments. The accuracy of the training phase corresponds to the average of the performance of the best particles obtained in each experiment, whereas that the accuracy of the testing phase is obtained from the average of the performance of these particles applied to the testing set. The highest accuracy values are remarked in bold font.

Tab. 7. Accuracy of testing phase over each experiment

	OMOPSO		PSO	
	Experiments		Experiments	
Dataset	#1	#2	#3	#4
Iris Plant	0.94 ± 0.0383	**0.9543** ± **0.0220**	0.9003 ± 0.0355	0.8883 ± 0.0303
Wine	0.7322 ± 0.0604	**0.7397** ± **0.0610**	0.6706 ± 0.0505	0.6858 ± 0.0451
Glass	0.3516 ± 0.1141	**0.3695** ± **0.1163**	0.3472 ± 0.0947	0.3594 ± 0.0960
SPECT	0.7043 ± 0.1051	0.7019 ± 0.1006	**0.7157** ± **0.0545**	0.7073 ± 0.0523

Tab. 8. Analysis of reduction of features of input vector

	Experiments			
	#2		#4	
Dataset	Average number of features employed	Rate of features used	Average number of features employed	Rate of features used
Iris Plant	2.575 ± 0.747	0.640	3.050 ± 0.221	0.760
Wine	8.475 ± 3.266	0.652	5.850 ± 1.902	0.450
Glass	5.550 ± 1.999	0.617	6.00 ± 1.377	0.667
SPECT	10.325 ± 5.677	0.469	21.675 ± 0.526	0.985

Table 8 shows the average amount of input features employed by the LIF neuron model and its corresponding rate concerning the total size of the original input feature vector.

Several statistic tests were applied to the obtained results. Firstly, Shapiro-Wilk test was executed to identify the kind of parametric or non-parametric tests to be used along with our data. Our tests were implemented using R programming language, and the CONTROLTEST package tool (available at http://sci2s.ugr.es/sicidm/) was used for non-parametric comparison between experiments. Specifically, three non-parametric tests were applied: Friedman, Friedman Aligned Ranks, and Quade.

Firstly, the results from statistic tests for the Training phase are shown and discussed. Subsequently, the results of statistic tests computed in the Testing phase are analyzed.

In the Shapiro-Wilk test, the null-hypothesis (H_0) states the samples come from a normal distribution. In Table 9, for a significance level of α = 0.05, the *P-values* obtained show that approximately half of the results do not reject H_0, but the rest of the results reject H_0. Therefore, non-parametric statistics were applied since such tests include both cases.

Tab. 9. Shapiro-Wilk test in Training phase

	OMOPSO		PSO	
	Experiments		Experiments	
Dataset	#1	#2	#3	#4
Iris Plant	0.0002868	0.000761	0.1257	0.189
	H_0 is rejected	H_0 is rejected	H_0 is not rejected	H_0 is not rejected
Wine	0.6275	0.0407	0.08713	0.3317
	H_0 is not rejected	H_0 is rejected	H_0 is not rejected	H_0 is not rejected
Glass	0.0002413	0.001126	6.64E-14	1.09E-11
	H_0 is rejected	H_0 is rejected	H_0 is rejected	H_0 is rejected
SPECT	0.06032	0.07665	0.3015	0.09427
	H_0 is not rejected	H_0 is not rejected	H_0 is not rejected	H_0 is not rejected

Friedman, Friedman Aligned Ranks, and Quade tests were applied to the obtained results. In these tests, the null-hypothesis (H_0) states that the data of the experiments follow the same distribution [19] (there is no difference in their performance).

Table 10 reports the average ranks obtained from these tests on the whole experiments. The smaller values, in bold font, indicate that Experiment #1 had consistently the best performance.

Tab. 10. Average rankings of the experiments for the Training phase

Experiment	Friedman	Friedman Aligned Ranks	Quade
#1	**1.25**	**4.0**	**1.2**
#2	1.75	5.0	1.80
#3	3.25	12.25	3.199
#4	3.75	12.75	3.8

Table 11 shows the *P-value* for each statistical test and the sentence corresponding to the status of H_0 for a significance level α = 0.05. If the *P-value* is greater than α then indicates that not exist evidence to reject H_0. Therefore, the tests Friedman and Quade rejected H_0. However, these results do not give enough information to select the best experiment, so that, a post-hoc procedure was necessary to do. From Table 10, Experiment #1 was taken as the control experiment.

Tab. 11. Contrast the null-hypothesis in Training phase

	Friedman	**Friedman Aligned Ranks**	**Quade**
P-values	0.01694	0.3806	1.04E-04
	H_0 is rejected	H_0 is not rejected	H_0 is rejected

Table 12 shows the results of the post-hoc procedure, where the *P-values* were adjusted by Holm's correction. For $\alpha = 0.05$, the adjusted *P-values* for the comparison between the control experiment and the Experiments #3 and #4 show that the Experiment #1 had better performance.

Then, it is presented the results for the Testing phase.

Table 13 shows the results of the Shapiro-Wilk test where the *P-values* were contrasted with a significance level of $\alpha = 0.05$. The *P-values* obtained show that five results reject H_0, and eleven results do not reject H_0. Next, non-parametric statistic tests were applied.

Tab. 12. Adjusted P-values for Training phase

	Friedman	**Quade**
Experiment	Holm	Holm
#1 vs #4	0.01667	0.01667
#1 vs #3	0.025	0.025
#1 vs #2	0.05	0.05

Tab. 13. Shapiro-Wilk test in Testing phase

	OMOPSO		**PSO**	
	Experiments		**Experiments**	
Dataset	#1	#2	#3	#4
Iris Plant	0.05354	0.02317	0.4062	0.2015
	H_0 is not rejected	H_0 is rejected	H_0 is not rejected	H_0 is not rejected
Wine	0.4185	0.4515	0.4542	0.2932
	H_0 is not rejected	H_0 is not rejected	H_0 is not rejected	H_0 is not rejected
Glass	0.06616	0.00249	4.17E-08	4.71E-07
	H_0 is not rejected	H_0 is rejected	H_0 is rejected	H_0 is rejected
SPECT	0.002891	0.08466	0.7195	0.07491
	H_0 is rejected	H_0 is not rejected	H_0 is not rejected	H_0 is not rejected

Table 14 shows the average ranks obtained from Friedman, Friedman Aligned Ranks and Quade tests for whole results. In the three tests, the smaller average ranks, in bold font, specify that Experiment #2 had the best performance.

Tab. 14. Average rankings of the experiments for the Testing phase

Experiment	**Friedman**	**Friedman Aligned Ranks**	**Quade**
#1	2.5	6.5	2.3
#2	**1.75**	**4.75**	**1.2999**
#3	3.0	11.75	3.4
#4	2.75	11.0	3.0

Tab. 15. Contrast the null-hypothesis in Testing phase

	Friedman	**Friedman Aligned Ranks**	**Quade**
P-values	0.5519	0.3632	0.0381
	H_0 is not rejected	H_0 is not rejected	H_0 is rejected

Table 15 shows the *P-value* for each statistical test. The significance level was set up $\alpha = 0.05$. Quade test rejects H_0. Nonetheless, this result does not present enough information to choose of the best experiment. So, a post-hoc procedure was made. From Table 14, Experiment #2 was used as the control experiment.

Tab. 16. Adjusted P-values for Testing phase

	Quade
Experiment	Holm
#2 vs #4	0.01667
#2 vs #3	0.025
#2 vs #1	0.05

Table 16 shows the results of the post-hoc procedure for Quade test, where *P-values* were adjusted by Holm's correction. The *P-values* were compared against a significance level of $\alpha = 0.05$. The *P-values* for the comparison between the control experiment and the Experiment #3 and #4 show that the Experiment #2 had better performance.

5. Conclusion

This paper presents a methodology for training full and partially connected LIF spiking neurons using the OMOPSO algorithm for solving pattern recognition problems. The experiments were designed with a multi-objective approach and their results were compared statistically with the results of mono-objective experiments. Each experiment was tested on four well-known benchmark datasets by performing 40 independently executions for each dataset.

The results have shown that the Experiments #1 and #2 had the best performances in the Training and Testing phases respectively. Therefore, the multi-objective approach provides an adequate alternative to optimize LIF spiking neurons.

One interesting characteristic of our methodology consists on the reduction of dimensionality of the input feature vectors to avoid redundancies in the input information.

As future work, we propose to include the LIF parameters into the OMOPSO algorithm to explore better non-dominated solutions and to implement more multi-objective algorithms from state of the art for training LIF spiking neurons.

ACKNOWLEDGEMENTS

The authors express their gratitude to the National Technology of Mexico and the University of Guanajuato. C. Juarez-Santini and A. Rojas-Dominguez thank the National Council of Science and Technology of Mexico (CONACYT) for the support provided by means of the Scholarship for Postgraduate Studies and research grant CATEDRAS-2598, respectively.

AUTHORS

Carlos Juarez-Santini – Postgraduate Studies and Research Division, Leon Institute of Technology – National Technology of Mexico, Leon, Guanajuato, Mexico
e-mail: jusca_94@hotmail.com.

Manuel Ornelas-Rodriguez* – Postgraduate Studies and Research Division, Leon Institute of Technology – National Technology of Mexico, Leon, Guanajuato, Mexico
e-mail: mornelas67@yahoo.com.mx.

Jorge Alberto Soria-Alcaraz – Department of Organizational Studies, DCEA-University of Guanajuato, Guanajuato, Mexico, e-mail: jorge.soria@ugto.mx.

Alfonso Rojas-Domínguez – Postgraduate Studies and Research Division, Leon Institute of Technology – National Technology of Mexico, Leon, Guanajuato, Mexico
e-mail: alfonso.rojas@gmail.com.

Hector J. Puga-Soberanes – Postgraduate Studies and Research Division, Leon Institute of Technology – National Technology of Mexico, Leon, Guanajuato, Mexico
e-mail: pugahector@yahoo.com.mx.

Andrés Espinal – Department of Organizational Studies, DCEA-University of Guanajuato, Guanajuato, Mexico, e-mail: aespinal@ugto.mx.

Horacio Rostro-Gonzalez – Department of Electronics, DICIS-University of Guanajuato, Salamanca, Guanajuato, Mexico, e-mail: hrostrog@ugto.mx.

* Corresponding author

REFERENCES

[1] M. van Gerven and S. Bohte, "Editorial: Artificial Neural Networks as Models of Neural Information Processing", *Frontiers in Computational Neuroscience*, vol. 11, 2017, 1–2, DOI: 10.3389/fncom.2017.00114.

[2] K. Soltanian, F. A. Tab, F. A. Zar and I. Tsoulos, "Artificial neural networks generation using grammatical evolution". In: *2013 21st Iranian Conference on Electrical Engineering (ICEE)*, 2013, 1–5, DOI: 10.1109/IranianCEE.2013.6599788.

[3] W. Maass, "Networks of spiking neurons: The third generation of neural network models", *Neural Networks*, vol. 10, no. 9, 1997, 1659–1671, DOI: 10.1016/S0893-6080(97)00011-7.

[4] D. Gardner, *The Neurobiology of neural networks*, MIT Press, 1993.

[5] N. G. Pavlidis, O. K. Tasoulis, V. P. Plagianakos, G. Nikiforidis and M. N. Vrahatis, "Spiking neural network training using evolutionary algorithms". In: *Proceedings. 2005 IEEE International Joint Conference on Neural Networks,* vol. 4, 2005, 2190–2194, DOI: 10.1109/IJCNN.2005.1556240.

[6] A. A. Hopgood, *Intelligent Systems for Engineers and Scientists*, CRC Press/Taylor & Francis Group, 2012.

[7] B. A. Garro and R. A. Vázquez, "Designing Artificial Neural Networks Using Particle Swarm Optimization Algorithms", *Computational Intelligence and Neuroscience*, 2015, 1–20, DOI: 10.1155/2015/369298.

[8] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning internal representations by error propagation". In: *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1*, MIT Press, 1986, 318–362.

[9] D. Karaboga, B. Akay and C. Ozturk, "Artificial Bee Colony (ABC) Optimization Algorithm for Training Feed-Forward Neural Networks". In: V. Torra, Y. Narukawa and Y. Yoshida (eds.), *Modeling Decisions for Artificial Intelligence*, vol. 4617, 2007, 318–329, DOI: 10.1007/978-3-540-73729-2_30.

[10] S. Ding, H. Li, C. Su, J. Yu and F. Jin, "Evolutionary artificial neural networks: a review", *Artificial Intelligence Review*, vol. 39, no. 3, 2013, 251–260, DOI: 10.1007/s10462-011-9270-6.

[11] R. A. Vazquez and A. Cachon, "Integrate and Fire neurons and their application in pattern recognition". In: *2010 7th International Conference on Electrical Engineering Computing Science and Automatic Control*, 2010, 424–428, DOI: 10.1109/ICEEE.2010.5608622.

[12] A. Cachón and R. A. Vázquez, "Tuning the parameters of an integrate and fire neuron via a genetic algorithm for solving pattern recognition problems", *Neurocomputing*, vol. 148, 2015, 187–197, DOI: 10.1016/j.neucom.2012.11.059.

[13] E. M. Izhikevich, "Which Model to Use for Cortical Spiking Neurons?", *IEEE Transactions on Neural Networks*, vol. 15, no. 5, 2004, 1063–1070, DOI: 10.1109/TNN.2004.832719.

[14] C. A. Coello Coello and M. S. Lechuga, "MOPSO: a proposal for multiple objective particle swarm optimization". In: *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02*, vol. 2, 2002, 1051–1056, DOI: 10.1109/CEC.2002.1004388.

[15] M. R. Sierra and C. A. Coello Coello, "Improving PSO-Based Multi-objective Optimization Using Crowding, Mutation and ∈-Dominance". In: C. A. Coello Coello, A. Hernández Aguirre and E. Zitzler (eds.), *Evolutionary Multi-Criterion Optimization*, vol. 3410, 2005, 505–519, DOI: 10.1007/978-3-540-31880-4_35.

[16] J. J. Durillo, A. J. Nebro and E. Alba, "The jMetal framework for multi-objective optimization: Design and architecture". In: *IEEE Congress on Evolutionary Computation*, 2010, 1–8, DOI: 10.1109/CEC.2010.5586354.

[17] J. J. Durillo and A. J. Nebro, "jMetal: A Java framework for multi-objective optimization", *Advances in Engineering Software*, vol. 42, no. 10, 2011, 760–771, DOI: 10.1016/j.advengsoft.2011.05.014.

[18] "UCI Machine Learning Repository, Irvine, CA: University of California, School of Information and Computer Science". D. Dua and C. Graff, http://archive.ics.uci.edu/ml. Accessed on: 2020-05-28.

[19] J. Derrac, S. García, D. Molina and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms", *Swarm and Evolutionary Computation*, vol. 1, no. 1, 2011, 3–18, DOI: 10.1016/j.swevo.2011.02.002.