Integrate and Fire neurons and their application in pattern recognition

Conference Paper · September 2010

DOI: 10.1109/ICEEE.2010.5608622 · Source: DBLP

2 authors, including:

Roberto Antonio Vázquez
Universidad La Salle
79 PUBLICATIONS **642** CITATIONS

SEE PROFILE

2010 7th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE 2010)
Tuxtla Gutiérrez, Chiapas, México. September 8-10, 2010.

Integrate and Fire Neurons and their Application in Pattern Recognition

Roberto A. Vazquez, Aleister Cachón
Escuela de Ingeniería, Universidad La Salle,
Benjamín Franklin 47, Col. Condesa, CP. 06140, México D.F., México
Contact: ravem@lasallistas.org.mx

Abstract —**In this paper, it is shown how a Leaky Integrate and Fire (LIF) neuron can be applied to solve non-linear pattern recognition problems. Given a set of input patterns belonging to K classes, each input pattern is transformed into an input signal, then the LIF neuron is stimulated during T ms and finally the firing rate is computed. After adjusting the synaptic weights of the neuron model, we expect that input patterns belonging to the same class generate almost the same firing rate and input patterns belonging to different classes generate firing rates different enough to discriminate among the different classes. At last, a comparison between a feed-forward neural network and the LIF neuron is presented when applied to solve non-linear problems.**

Keywords — **Leaky Integrate and Fire Neurons, Pattern Recognition, Differential Evolution**

I. Introduction

Artificial neural networks have been applied in a wide range of problems such as pattern recognition, forecasting, and intelligent control. Perhaps, among the most popular neural network models we could mention the feed-forward neural network trained with the back-propagation algorithm [1-2]. However, this type of neural network cannot reach an optimum performance in non-linear problems if it is not well designed. The selection of a learning algorithm and parameters such as number of neurons, number of layers, and the transfer functions, determine the accuracy of the network, resulting in complex architectures to solve the problem [3].

Spiking neuron models have been called the 3rd generation of artificial neural networks [4]. These models increase the level of realism in a neural simulation and incorporate the concept of time. Spiking models have been applied in a wide range of areas from the field of computational neurosciences [5] such as: brain region modeling [6], auditory processing [7-8], visual processing [9-10], robotics [11-12] and so on.

In this paper, it is shown how a Leaky Integrate and Fire (LIF) neuron can be applied to solve non-linear pattern recognition problems. A LIF neuron is stimulated during T ms with an input signal and fires when its membrane potential reaches a specific value generating an action potential (spike) or a train of spikes. Given a set of input patterns belonging to K classes, each input pattern is transformed into an input signal, then the spiking neuron is stimulated during T ms and finally the firing rate is computed. After adjusting the synaptic weights of the neuron model, we expect that input patterns belonging to the same class generate almost the same firing rate; on the other hand, we also expect that input patterns belonging to different classes generate firing rates different enough to discriminate among the different classes. At last, a comparison between a feed-forward neural network and a spiking neuron is presented when are applied to solve non-linear problems. The main contribution of this paper is the description of a methodology to apply a LIF neuron in pattern recognition tasks.

II. Leaky Integrate and Fire Model

A typical spiking neuron can be divided into three functionally distinct parts, called dendrites, soma, and axon. The dendrites play the role of the *input device* that collects signals from other neurons and transmits them to the soma. The soma is the *central processing unit* that performs an important non-linear processing step: if the total input exceeds a certain threshold, then an output signal is generated. The output signal is taken over by the *output device*, the axon, which delivers the signal to other neurons. The neuronal signals consist of short electrical pulses. The pulses, so-called action potentials or spikes, have an amplitude of about 100 mV and typically a duration of 1-2 ms. The form of the pulse does not change as the action potential propagates along the axon. A chain of action potentials emitted by a single neuron is called a spike train (a sequence of stereotyped events which occur at regular or irregular intervals). Since all spikes of a given neuron look alike, the form of the action potential does not carry any information. Rather, it is the number and the timing of spikes which matter. The action potential is the elementary unit of signal transmission. Action potentials in a spike train are usually well separated. Even with a very strong input, it is impossible to excite a second spike during or immediately after a first one. The minimal distance between two spikes defines the absolute refractory period of the neuron. The absolute refractory period is followed by a phase of relative refractoriness where it is difficult, but not impossible to excite an action potential [16].

IEEE Catalog Number: CFP10827-ART
ISBN: 978-1-4244-7314-4
978-1-4244-7314-4/10/$26.00 ©2010 IEEE

The LIF model is one of the most widely used in computational neuroscience. One of the reasons for this is that it is the easiest to implement. Its model is so simple that the computational cost is minimum compared against other spiking neuron models.

The neuron is considered leaky if there is decay with a characteristic time constant in the summed contributions to the membrane potential; when this "leak" is forfeit the model is considered a perfect integrator.

It can be concluded then that the model should be mathematically represented by the dynamics of the membrane potential. Such representation is as follows:

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

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

In this paper, we will use another representation of the LIF model [14] defined as:

$$\begin{aligned} v' &= I + a - bv \\ \text{if } v \geq v_{thresh}&, \text{ then } v \leftarrow c \end{aligned} \qquad (2)$$

where I is the input current of the neuron (directly injected into the neuron), v_{thresh} is the threshold for firing the spike, and c is the reset or rest state voltage.

III. Proposed Method

Before describing the proposed method applied to solve pattern recognition problems, it is important to notice that when the input current signal changes, the response of the LIF neuron also changes, generating different firing rates, see Fig. 1.

The firing rate is computed as the number of spikes generated in an interval of duration T divided by T. The neuron is stimulated during T ms with an input signal and fires when its membrane potential reaches a specific value generating an action potential (spike) or a train of spikes.

Let $\left\{\mathbf{x}^i, k\right\}_{i=1}^{p}$ be a set of p input patterns where $k = 1,\ldots,K$ is the class to which $\mathbf{x}^i \in \mathbb{R}^n$ belongs. First of all, each input pattern is transformed into an input signal I, after that the spiking neuron is stimulated using I during T ms and then the firing rate of the neuron is computed. With this information, the average firing rate $\mathbf{AFR} \in \mathbb{R}^K$ and standard deviation $\mathbf{SDFR} \in \mathbb{R}^K$ of each class can be computed.

Finally, we expect that input patterns, which belong to the same class, generate almost the same firing rates (low standard deviation); and input patterns, which belong to different classes, generate firing rates different enough (average spiking rate of each class widely separated) to discriminate among the different classes. In order to achieve that, a training phase is required in the proposed method.

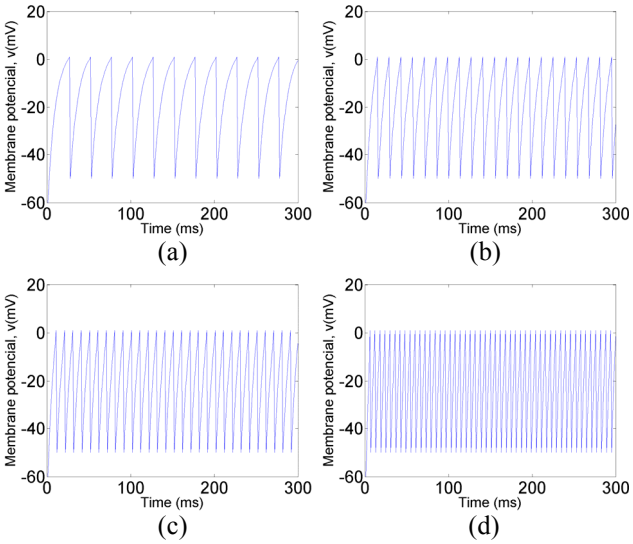


Fig. 1. Different firing rates of a LIF neuron with different input currents: (a) 0.4 mV. (b) 1.6044 mV. (c) 3 mV. (d) 7 mV.

During training phase, the synaptic weights of the model, which are directly connected to the input pattern, are adjusted by means of a differential evolution algorithm.

At last, the class of an input pattern $\tilde{\mathbf{x}}$ is determined by means of the firing rates as

$$cl = \arg\min_{k=1}^{K}\left(\left|AFR_k - fr\right|\right) \qquad (3)$$

where fr is the firing rate generated by the neuron model stimulated with the input pattern $\tilde{\mathbf{x}}$.

LIF neurons cannot be directly stimulated with the input pattern $\mathbf{x} \in \mathbb{R}^n$, but with an injection current I.

Since synaptic weights are directly connected to the input pattern $\mathbf{x} \in \mathbb{R}^n$, the injection current of this input pattern can be computed as

$$I = \mathbf{x} \cdot \mathbf{w} \cdot \theta \qquad (4)$$

where $\mathbf{w} \in \mathbb{R}^n$ is the set of synaptic weights of the neuron model and θ a gain factor.

Finally, synapses of the neuron model are adjusted by means of a differential evolution algorithm. Evolutionary algorithms not only have been used to design artificial neural networks [3], but also to evolve structure-function

IEEE Catalog Number: CFP10827-ART
ISBN: 978-1-4244-7314-4
978-1-4244-7314-4/10/$26.00 ©2010 IEEE

mapping in cognitive neuroscience [17] and compartmental neuron models [18].

Differential evolution is a powerful and efficient technique for optimizing non-linear and non-differentiable continuous space functions [15]. Differential evolution is regarded as a perfected version of genetic algorithms for rapid numerical optimization. Differential evolution has a lower tendency to converge to local maxima with respect to the conventional genetic algorithm, because it simulates a simultaneous search in different areas of solution space. Moreover, it evolves populations with smaller number of individuals, and have a lower computation cost.

Differential evolution begins by generating a random population of candidate solutions in the form of numerical vectors. The first of these vectors is selected as the target vector. Next, differential evolution builds a trial vector by executing the following sequence of steps:

1. Randomly select two vectors from the current generation.
2. Use these two vectors to compute a difference vector.
3. Multiply the difference vector by weighting factor F.
4. Form the new trial vector by adding the weighted difference vector to a third vector randomly selected from the current population.

The trial vector replaces the target vector in the next generation if and only if the trial vector represents a better solution, as indicated by its measured cost value computed with a fitness function. Differential evolution repeats this process for each of the remaining vectors in the current generation. Differential evolution then replaces the current generation with the next generation, and continues the evolutionary process over many generations.

In order to achieve that input patterns, which belong to the same class, generate almost the same firing rates (low standard deviation), and input patterns, which belong to different classes, generate firing rates different enough (average spiking rate of each class widely separated) to discriminate among the different classes, the next fitness function was proposed

$$f = \frac{1}{dist(\mathbf{AFR})} + \sum_{k=1}^{K} SDFR_k \qquad (5)$$

where $dist(\mathbf{AFR})$ is the summation of the Euclidian distances among the average firing rate of each class.

III. Results

In order to evaluate and compare the accuracy of the proposed method, against feed-forward neural networks, several experiments were performed using 2 datasets taken from UCI machine learning benchmark repository [13]: iris plant and wine datasets.

The iris plant dataset is composed of 3 classes and each input pattern is composed of 4 features. The wine dataset is composed of 3 classes and each input pattern is composed of 13 features.

Both datasets were partitioned into two sets: training set (50% of the samples of each dataset) and testing set (50% of the samples of each dataset).

First of all, the parameters for the neuron model as for the training algorithms were defined. These parameters were used through all sets of experiments. The parameters for the LIF neuron were defined as $a = 0.5$, $b = -0.001$, $c = -50$, $v_r = -60$ and $v_{peak} = 50$. The Euler method was used to solve the differential equation of the model with $dt = 1$. The parameter to compute input current I from the input pattern was set to $\theta = 0.1$ with a duration of $T = 1000$. For the case of the differential evolution algorithm, $NP = 40$, $MAXGEN = 1000$, $F = 0.8$, $XMAX = 1$, $XMIN = 0$ and $CR = 0.9$.

The classic back-propagation and Levenberg-Marquardt algorithms were used to train the feed-forward neural network. The number of generations was set to 10000 and the learning rate $\alpha = 0.1$. Concerning the architecture of the feed-forward neural network, one hidden layer composed of five hyperbolic tangent neuron and an output layer composed of K hyperbolic tangent neurons were used in all experiments.

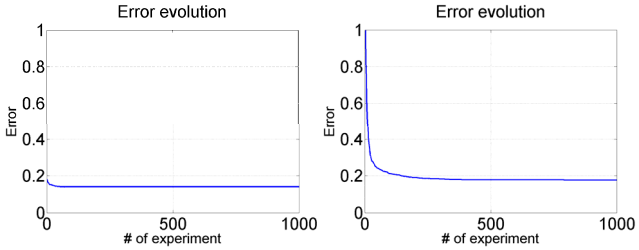
The stop criterion for the three algorithms was the number of generations or the minimum error which was set to $e = 0$.

The accuracy (classification rate), achieved with the proposed method, was computed as the number of input patterns correctly classified divided by the total number of tested input patterns. To validate the accuracy of the proposed method 10 experiments over each dataset were performed. The same metric and number of experiments were used to measure the accuracy of the feed-forward neural network trained with the two different algorithms.

Through the learning evolution process we observed that independently of the initial value, the error converges quickly, see Fig. 2. This is more evident in the wine dataset where a maximum error was quickly reduced into an acceptable level. As the error approached its stable state, the improvement rate reduced greatly.

IEEE Catalog Number: CFP10827-ART
ISBN: 978-1-4244-7314-4
978-1-4244-7314-4/10/$26.00 ©2010 IEEE

The experimental results, obtained with the iris plant dataset, are shown in Fig 3. The reader can appreciate in Table I that the LIF neuron trained with the proposed method provided better results than the feed-forward neural network trained with the two classical algorithms. Furthermore, standard deviation from all experiments was really small compared against that obtained using feed-forward neural networks. Something that should be remarked is that while the feed-forward neural network was composed of 8 neurons, the proposed method used only one LIF neuron.



(a) (b)

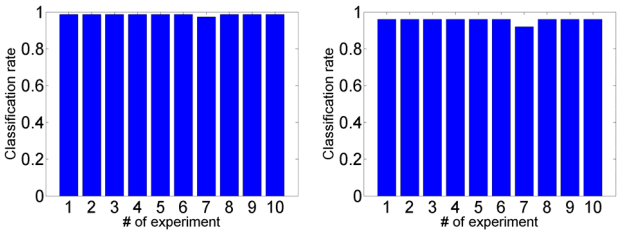
Fig. 2. Average error evolution for the iris plant dataset (a) and the wine dataset (b)

The experimental results, obtained with the wine dataset, are shown in Fig 4. The reader can appreciate in Table I that the LIF neuron trained with the proposed method also provided better results than the feed-forward neural network trained with the two classical algorithms. Nevertheless, the average classification rate obtained during testing phase was not as good as that obtained during training phase. Furthermore, even if the feed-forward neural network was composed with 8 neurons, the proposed method provided much better results.

Although we are aware of the low accuracy achieved with the feed-forward neural network, it is important to remark that we are not concerning ourselves with the topology of the feed-forward neural network. This is with the purpose of showing that we do not need to do so with the proposed method and to highlight its flexibility in solving different problems.

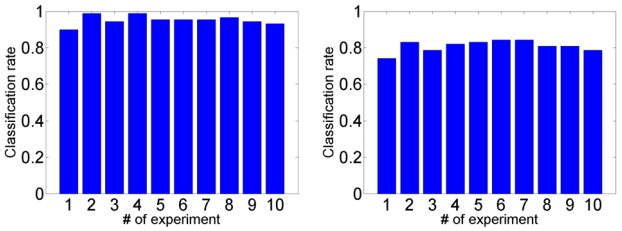
IV. Discussion

In Tables I and II, the average and standard deviation classification rate computed from all experimental results are shown. The results obtained with the spiking neuron model, trained with the proposed method, significantly improve the results obtained with feed-forward neural networks when applied to non-linear problems. Furthermore, the standard deviations computed from all experiments support the stability of the proposed method. While feed-forward neural networks provided different results through each experiment, the proposed method provided similar results.



(a) (b)

Fig. 3. Experimental results obtained with the iris plant dataset.
(a) Accuracy of the proposed method during training phase.
(b) Accuracy of the proposed method during testing phase.



(a) (b)

Fig. 4. Experimental results obtained with the wine dataset.
(a) Accuracy of the proposed method during training phase.
(b) Accuracy of the proposed method during testing phase.

TABLE I
AVERAGE ACCURACY

Data base	Back-propagation algorithm		Levenberg-Marquartd algorithm		Proposed method	
	Tr. cr.	Te. cr.	Tr. cr.	Te. cr.	Tr. cr.	Te. cr.
Iris plant	0.8707	0.8500	0.8467	0.8020	0.9853	0.9560
Wine	0.3933	0.4045	0.6433	0.6433	0.9528	0.8101

Tr. cr. = Training classification rate, Te. cr. = Testing classification rate.

TABLE II
STANDARD DEVIATION

Data base	Back-propagation algorithm		Levenberg-Marquartd algorithm		Proposed method	
	Tr. sd.	Te. sd.	Tr. sd.	Te. sd.	Tr. sd.	Te. sd.
Iris plant	0.1661	0.1688	0.2520	0.2339	0.0042	0.0126
Wine	0.0406	0.0471	0.3274	0.3274	0.0264	0.0315

Tr. sd. = Training standard deviation, Te. sd. = Testing standard deviation.

These preliminary results suggest that spiking neurons can be considered as an alternative way to perform different pattern recognition tasks.

We can also conjecture that if only one neuron is capable to solve pattern recognition problems, perhaps several spiking neurons working together can improve the

IEEE Catalog Number: CFP10827-ART
ISBN: 978-1-4244-7314-4
978-1-4244-7314-4/10/$26.00 ©2010 IEEE

experimental results obtained in this research. However, that is something that should be proved.

V. Conclusions

In this paper a new method to apply a spiking neuron in a pattern recognition task was proposed. This method is based on the firing rates produced with a LIF neuron when stimulated. The firing rate is computed as the number of spikes generated in an interval of duration T divided by T. A spiking neuron is stimulated during T ms with an input signal and fires when its membrane potential reaches a specific value generating an action potential (spike) or a train of spikes.

The training phase of the neuron model was done by means of a differential evolution algorithm. After training, we observed that input patterns, which belong to the same class, generate almost the same firing rates (low standard deviation) and input patterns, which belong to different classes, generate firing rates different enough (average spiking rate of each class widely separated) to discriminate among the different classes.

Through several experiments we observed that the proposed method, improves the results obtained with feed-forward neural networks, when it is applied to non-linear problems. On the other hand, the standard deviations computed from all experiments, support the stability of the proposed method. While feed-forward neural networks provided different results through each experiment, the proposed method provided similar results.

We conclude that spiking neurons can be considered as an alternative tool to solve pattern recognition problems.

Nowadays, we are analyzing different behaviors of the LIF model in order to determine which configuration is the most suitable for a pattern recognition task. In addition, we are testing different fitness functions to improve the results obtained in this research. Furthermore, we are researching different alternatives of combining several LIF neurons in one network to improve the results obtained in this research and then apply it in more complex pattern recognition problems such as face, voice and 3D object recognition. In addition, we are planning to compare this kind of LIF neural network against related approaches such as oscillatory neural networks when they are applied to complex non-linear problems.

Acknowledgments

The authors thank Universidad La Salle for the economical support under grant number ULSA I-113/10.

References

[1] Anderson, J. “Introduction to Neural Networks,” (Cambridge, MA: MIT Press), 1995.
[2] Werbos, P. J. “Backpropagation through time: What it does and how to do it,” Proc. IEEE, vol. 78, pp. 1550–1560, 1990.
[3] Garro, B. A., Sossa, H. and Vazquez, R. A. “Design of Artificial Neural Networks using a Modified Particle Swarm Optimization Algorithm,” In: Proc. IEEE Int. Joint Conf. Neural Networks, pp. 938–945, 2009.
[4] Maass, W. "Networks of spiking neurons: the third generation of neural network models." Neural Networks **10**(9): 1659-1671, 1997.
[5] Rieke, F., R. Steveninck, et al. “Spikes:: Exploring the Neural Code”, Bradford Book, 1997.
[6] Hasselmo, M. E., C. Bodelon, et al. “A Proposed Function for Hippocampal Theta Rhythm: Separate Phases of Encoding and Retrieval Enhance Reversal of Prior Learning”, Neural Computation **14**: 793-817, 2002.
[7] Hopfield, J. J. and C. D. Brody "What is a moment? "Cortical" sensory integration over a brief interval." Proceedings of the National Academy of Sciences **97**(25): 13919, 2000.
[8] Loiselle, S., J. Rouat, et al. “Exploration of rank order coding with spiking neural networks for speech recognition. Neural Networks”, IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on, 2005.
[9] Azhar, H., K. Iftekharuddin, et al. “A chaos synchronization-based dynamic vision model for image segmentation”, Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on, 2005.
[10] Thorpe, S. J., R. Guyonneau, et al. “SpikeNet: Real-time visual processing with one spike per neuron” Neurocomputing **58**(60): 857-864, 2004.
[11] Di Paolo, E. A. “Spike-timing dependent plasticity for evolved robots” Adaptive Behavior **10**(3): 243-263, 2002.
[12] Floreano, D., J. Zufferey, et al. (2005). "From wheels to wings with evolutionary spiking neurons." Artificial Life **11**(1-2): 121–138
[13] Murphy, P. M. and Aha, D. W. “UCI repository of machine learning databases,” Dept. Inf. Comput. Sci., Univ. California, Irvine, CA, 1994.
[14] Izhikevich, E. M. “Which model to use for cortical spiking neurons?”, Neural Networks, IEEE Transactions on **15**(5): 1063-1070, 2004.
[15] Price, K., Storn, R. M., Lampinen, J. A. “Diffentential evolution: a practical approach to global optimization”, Springer-verlag, 2005.
[16] Gerstner, W., Kistler, W. “Spiking Neuron Models”, Cambridge University Press, 2002.
[17] Frias-Martinez, E. Gobet, F. “Automatic generation of cognitive theories using genetic programming”, Minds and Machines 17(3), 287-309, 2007.
[18] Hendrickson, E., et al. “Converting a globus pallidus neuron model from 585 to 6 compartments using an evolutionary algorithm”, J. BMC Neurosci. 8(suppl 2), P122, 2007

IEEE Catalog Number: CFP10827-ART
ISBN: 978-1-4244-7314-4
978-1-4244-7314-4/10/$26.00 ©2010 IEEE