Development of a Library for Sound Classification Using Spiking Neural Network

José Tomás Molas[1,2], Iván Peralta[1], Cesar Martinez[1,2], and Hugo Leonardo Rufiner[1,2,3]

[1] Laboratorio de Cibernética, Facultad de Ingeniería, Universidad Nacional de Entre Ríos, CC 47 Suc. 3, E3100, Ruta 11, km. 10, Oro Verde, Entre Ríos, Argentina

[2] Centro de I+D en Señales, Sistemas e Inteligencia Computacional (SINC(i)), Facultad de Ingeniería y Cs. Hídricas, Universidad Nacional del Litoral CC 217, Ciudad Universitaria, Paraje El Pozo, S3000 Santa Fe, Argentina

[3] CONICET, Argentina

{cmartinez,lrufiner}@bioingenieria.edu.ar, {bioivanperalta,josetomasmolas}@gmail.com

Abstract— **This article describes the development of a library for Matlab (The MathWorks, Inc.) for sounds recognition based on Spiking Neural Networks. The mathematical model of the integrate and fire type is presented. The main modules of the library are as follows: encoding in "spikes", spiking neural network and finally the training module. To illustrate its use, the library application to the task of phoneme recognition is presented.**

Keywords— **automatic speech recognition, artificial neural networks, spiking neural networks.**

I. INTRODUCTION

Artificial neural networks (ANN, Artificial Neural Networks English) are composed of basic units called neurons, interconnected by different weights that determine how strongly interacting these neurons. Unlike the ANN, in the live cells, the temporal dimension, in the excitation signal and the response, are as important as the intensity. This way of working of the ANN has been produced by an oversimplification of the first biological models that apparently omitted this dimension and now appears as crucial. As usual, simplification is only valid in some contexts.

The search for a closer analogy to the biological reality has recently led to the development of pulsed neural networks (SNN of Spiking Neural Networks) [1]. Neurons communicate with one another in a language in which the given meaning to the receiving neuron is encoded in the time of action potentials (AP). These sets of action potentials can be treated as a signal. A neuron can fire (generate AP) if its input AP occurring in a given sequence and not fire if the same increase in distance (time difference between AP), whether the interval between them is more long,whether the interval between them is longer or shorter.

Thus, communication and computation in biological neurons are completely different from the way that work until now computers and classical artificial neural networks. If these information can be encoded in the phase of action potentials (or pulses for artificial neurons), could also be incorporated into models of neurons the ability to change their behavior depending on the timing of the input signals. This would lead to a learning rule for changing the temporal properties of the network connections, ie the idea of dynamic synapses [2, 3].

In this work the implementation of a Matlab [4] library is presented in and are tested the principal modules of spiking neural network to classify audio signals. The rest of the paper is organized as detailed below. Section 2 presents the mathematical model of pulsating networks. Section 3 presents the main modules of the software. Section 4 presents some numerical results and simulations. Finally, Section 5 presents the conclusions and proposals for future works.

II. SPIKING NEURON NETWORKS

A. Integrate and fire model

The neuron model used is the type pulsating integrates and fire, the model used in [5]. The neuron is modeled by an RC circuit (Fig. 1). The membrane potential is $\mu(t)$, while this potential is less than a threshold value $\mu_{threshold}$, the dynamics is modeled by:

$$\frac{d\mu(t)}{dt} = \frac{1}{C}\left(-\frac{\mu(t)-\mu_{rest}}{R}+I(t)\right) \tag{1}$$

donde:

$\mu(t) =$	Membrane potential versus time [Volts].
$u_{rest} =$	Initial membrane potential [Volts].
$I(t) =$	Total current that reaches the neuron [Amperes].
$R =$	Membrane resistance [Ohm].
$C =$	Membrane capacitance [Farad].

If the potential $\mu(t)$ exceeds the value $\mu_{thershold}$, the neuron fires (generates a *"spike'*) and the following events occur:

- The potential $u(t)$ is reset to a value u_{reset}.
- The neuron enters an absolute refractory period.
- The neuron generates a *spike* to exit.

A. Braidot and A. Hadad (eds.), *VI Latin American Congress on Biomedical Engineering CLAIB 2014, Paraná, Argentina 29, 30 & 31 October 2014*,
IFMBE Proceedings 49, DOI: 10.1007/978-3-319-13117-7_166

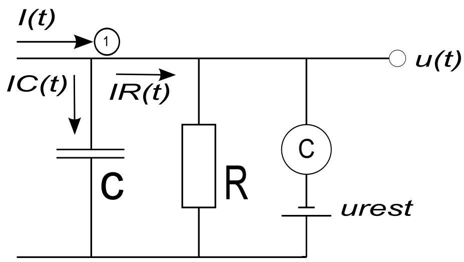

Figura 1: Electrical circuit of the *integrates and fire* model.

B. *Integrate-and-fire equations*

Applying Kirchhoff node rule to currents at Node 1 and Ohm's law in the circuit of Figure 1, we can derive the equations for the model:

$$I(t) \quad = \tfrac{\mu(t)}{R} + I_C(t) \tag{2}$$

Applying the definition of capacitance:

$$C\frac{d\mu(t)}{dt} \quad = I_C(t). \tag{3}$$

From (2) and (3), we obtain:

$$I(t) = \frac{\mu(t)}{R} + C\frac{d\mu(t)}{dt}$$

which is a linear ordinary differential equation (LODE) first order. If we now consider that the capacitor has an initial potential (μ_{rest}) we obtain:

$$\frac{d\mu(t)}{dt} \quad = \quad \frac{1}{C}\left(-\frac{\mu(t)-\mu_{rest}}{R} + I(t)\right) \tag{4}$$

Returning to equation (1) and write it differently:

$$\frac{d\mu(t)}{dt} + \frac{1}{C}\mu(t) = \frac{\mu_{rest}}{RC} + \frac{I(t)}{C}.$$

If we now assume a constant stimulus, given the following definitions and we rewrite the last equation:

$$I_0 \stackrel{\text{def}}{=} I(t)$$
$$\mu \stackrel{\text{def}}{=} \mu(t)$$
$$\mu' \stackrel{\text{def}}{=} \frac{d\mu(t)}{dt}$$

$$\mu' + \frac{1}{RC}\mu = \frac{I_0}{C} + \frac{\mu_{rest}}{RC} \tag{5}$$

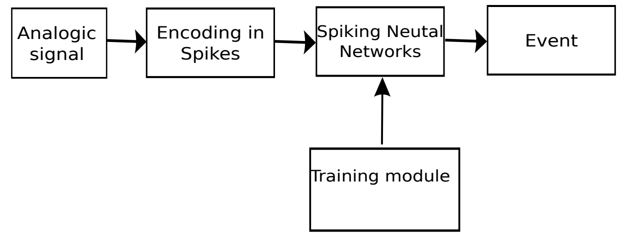

Figura 2: Principals software modules

The particular solution is [6]:

$$\mu(t) \quad = (\mu_{rest} + RI_0)\left(1 - e^{-\frac{t}{RC}}\right) + C_i e^{-\frac{t}{RC}}, \tag{6}$$

the latter being the form as used in the work.

III. SOFTWARE MODULES

In the Fig. 2 the principals software modules are presented.

A. *Signal encoding with spikes*

A very important process when working with spiking neural networks is the encoding of analog data pulse into trains or "spikes". The neural networks processed as input a set of times of firing and generate output to another set of firing times. There are several ways to encode analog values in sets of times of fires.

The simplest method is to encode the time proportional to the analog value, with the disadvantage that only a time value is obtained. A generalization of this method is to use a time window that moves over time, and different portions of a one-dimensional time signal is converted into pulses. Another simple way is to apply a coding threshold signal and generating a firing time set each time the value of the signal exceeds the predefined threshold. A method utilizing more than one neuron is encoding by populations of neurons. One way to implement this method is using *receptive fields* [7].

The method implemented in this library detects events on the spectrogram of an audio signal. The process is straightforward because the x-axis of the spectrogram is the time variable and it is only necessary to define the characteristic events or searched in the energy axis (Z axis) of the spectrogram, then place them in time. Another important feature of this method is that if we repeat the process for several frequencies in the same spectrogram, we obtain a set of firing times of trains or "spikes". The main functions of this module are: `GenerarMapaEventos`, `EspectrogramaNBandas` and `EtapaPreprocesado`.

B. Spiking Neural Network

The basic units of the network are type neurons integrate-and-fire as previously described. These neurons can be of two types: excitatory (α) or inhibitory (β). Excitatory neuron generates stimulus which increases the potential of the membrane of the postsynaptic neuron and inhibitory neuron decreases the potential. Neurons are organized in a feed-forward type network in three layers: input layer, hidden layer and output layer. The trigger response function is exponential and can be modified very easily. All parameters of each neuron can be configured. The main functions implemented in this module are: `MostrarPotencialesVsCorrientes`, `SimulacionAnalitica` and `SimulacionNumericaEuler`.

C. Training module

Training based on the temporal coding of spikes is an important area of research in the SNN because of its great similarity with biological learning and methods of supervised training for the perceptron and the sigmoid neurons can not be applied directly to the SNN. Exist other methods based on the error gradient methods [8] assuming linear approximation around of the fire instant of the neuron. Another important aspect of training is the number of spikes that are used for training the network. These methods can be classified into two types, simple spikes and multiple spikes training algorithms. Another important aspect of training is the error function to upgrade the connection weights in the case of neurons spikes can be used multiple measures of distances between spikes [9]. In the first step, in this training module will be implemented by means of genetic algorithms. The implemented function of this module is `ArmarGruposCapaW`, which is a method which groups the neurons are firing at a frequency close to sync for a short time and generate a spike on the post-synaptic neuron to recognize a pattern.

IV. SIMULATIONS AND RESULTS

A. Generating events (*"spikes"*)

The implemented algorithm analyzes a number of frequency bands of a word (in this case, the pronunciation of a digit) detects when and where the event of interest occurred. This event may be the maximum value, when it starts to grow or when it starts to decrease [10] [5]. A simple way to implement this algorithm is by applying the spectrogram to the audio signal, then the width of the time windows or the frequency resolution is calculated, to achieve the number of frequency bands that want to use, then a simple algorithm with the instants of events of interest to us are detected. The Fig. 3 shows the spectrogram of the word "one", the events generated over 40 frequency bands for two types of events, *offset* and *onset*.

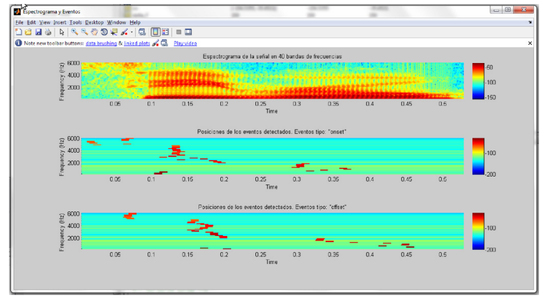

Figura 3: Above: 40-band spectrogram of the word *one*. Center: Representation of the positions *"spikes"* calculated for each frequency and the *offset* event. Bottom: Representation of the positions calculated for the *onset* event.

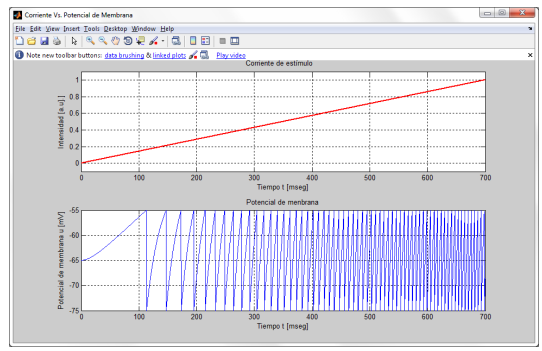

Figura 4: Top: stimulus current. Bottom: membrane potential.

B. Application of different stimuli (current)

To understand the functioning of the neuron *integrates and fire*, it is useful to apply different types of stimuli (current) and plot the behavior of the membrane potential. A current ramp is a simple stimulus to generate and gives us enough information about the relationship between stimulus current and firing frequency. In the Fig. 4 the relationship between the current and membrane potential are shown.

C. Two Alpha-coupled neurons

Here will be described an essential feature to understand how the Alpha-coupled neurons can recognize a pattern. The basic idea is very simple, it is part of the property that two neurons with the same parameters physical (τ, μ_{rest}, μ_{reset} and $\mu_{threshold}$) when stimulated by the same current will fire at the same frequency. This is shown in Fig. 5,

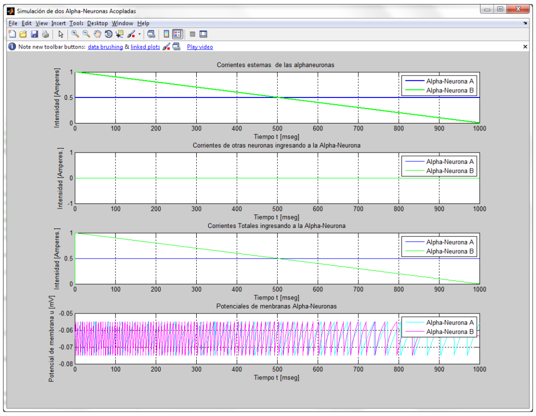

Figura 5: Membrane potentials in terms of two types of current into two Alpha-coupled neurons.

where the estimulam a neuron with a constant current and the other with a decreasing ramp. It can be seen as the currents are similar, the periods too. This feature can be used to generate groups of neurons, through the connection weights during the training stage. These neurons could recognize a phoneme, when firing at approximately the same time, for a set of events from the spectrogram coded. This same group of neurons would be connected to a single neuron of the output layer act as *synchronism detector* and generate a *spike* indicating this phoneme.

V. CONCLUSIONS

In this work we present some modules to a library for MATLAB using a spiking neural network to recognition sounds by encoding events from a time-frequency transformation. In subsequent work other learning strategies will be implemented in the training module and use the parallel computing capabilities of MATLAB. For availability contact the authors.

ACKNOWLEDGEMENTS

The authors acknowledge the support of: the *Universidad Nacional de Entre Ríos* (con PID NOVEL 6121) and the *Universidad Nacional de Litoral* (con CAI+D 2011 #58-511 and #58-525).

REFERENCIAS

1. Maass Wolfgang, Bishop Christopher M. *Pulsed neural networks*. MIT Press 2001.
2. Maass Wolfgang, Zador Anthony. Computing and learning with dynamic synapses *Pulsed neural networks.* 1999;6:321–336.
3. Mattia Maurizio, Giudice P. Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses *Neural Computation.* 2000;12:2305–2329.
4. MathWorks . MATLAB *http://www.mathworks.com/.* .
5. Wills Sebastian A. *Computation with spiking neurons*. PhD thesis Cambridge University 2004.
6. Kaplan Wilfred, Jimeno Amaury Llamas, Mateos Manuel López. *Matemáticas avanzadas: para estudiantes de ingeniería*. Addison Wesley Iberoamericana 1986.
7. Bohte Sander M, Kok Joost N, La PoutrÃ© Han. Unsupervised classification of complex clusters in networks of spiking neurons in *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*;3:279–284IEEE 2000.
8. Bohte Sander M, Kok Joost N, La Poutre Han. Error-backpropagation in temporally encoded networks of spiking neurons *Neurocomputing.* 2002;48:17–37.
9. Rossum M. A novel spike distance *Neural Computation.* 2001;13:751–763.
10. Hopfield John J, Brody Carlos D. What is a moment? Cortical sensory integration over a brief interval *Proceedings of the National Academy of Sciences.* 2000;97:13919–13924.