Spiking neuron network for image segmentation
Boudjelal Meftah, M. Debakla, M. Zaagane, A. Benyettou, Olivier Lezoray

To cite this version:

HAL Id: hal-00329522
https://hal.archives-ouvertes.fr/hal-00329522
Submitted on 11 Oct 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Spiking Neuron Network for Image Segmentation

B. Meftah, M. Debakla, M. Zaagane
Laboratoire de Recherche sur les Systèmes Biologiques et la Géomatique (LRSBG) équipe EDTEC, Centre Universitaire Mustapha Stambouli, Mascara, Algérie

A. Benyettou
Laboratoire SIMPA, BP 1505, Département d'Informatique, Faculté des Sciences, USTO-Mohamed Boudiaf, Oran, Algérie

O. Lezoray
Université de Caen, GREYC UMR CNRS 6072, 6 Bd. Marchal Juin, F-14050, Caen, France

Abstract

The process of segmenting images is one of the most critical ones in automatic image analysis whose goal can be regarded as to find what objects are presented in images. Artificial neural networks have been well developed. First two generations of neural networks have a lot of successful applications. Spiking Neuron Networks (SNNs) are often referred to as the 3rd generation of neural networks which have potential to solve problems related to biological stimuli. They derive their strength and interest from an accurate modeling of synaptic interactions between neurons, taking into account the time of spike emission. SNNs overcome the computational power of neural networks made of threshold or sigmoidal units. Based on dynamic event-driven processing, they open new horizons for developing models with an exponential capacity of memorizing and a strong ability to fast adaptation. Moreover, SNNs add a new dimension, the temporal axis, to the representation capacity and the processing abilities of neural networks. In this paper, we present how SNN can be applied with efficacy in image segmentation.

Key words: Classification, Segmentation, Learning, Microscopic cellular images, Segmentation, Spiking Neuron Network.

1. Introduction

Image segmentation consists of subdividing an image into its constituent parts and extracting these parts of interest. A large number of segmentation algorithms have been developed since the middle of 1960's [1], and this number continually increases at a fast rate. Simple and popular methods are threshold-based and process histogram characteristics of the pixel intensities of the image. Of course, thresholding has many limitations: the transition between objects and background has to be distinct and the result does not guarantee closed object contours, often requiring substantial post-processing.

Region-based methods have also been developed; they exploit similarity in intensity, gradient, or variance of neighboring pixels. Watersheds methods can be included in this category. The problem with these methods is that they do not employ any shape information of the image, which can be useful in the presence of noise.

Meanwhile, artificial neural networks are already becoming a fairly renowned technique within computer science. Since 1997, Maass [2,3] has quoted that computation and learning has to proceed quite differently in SNNs. He proposes to classify neural networks as follows:

- 1st generation: Networks based on McCulloch and Pitts' neurons as computational units, i.e. threshold gates, with only digital outputs (e.g. perceptrons, Hopfield network, Boltzmann machine, multilayer perceptrons with threshold units).

- 2nd generation: Networks based on computational units that apply an activation function with a continuous set of possible output values, such as sigmoid or polynomial or exponential functions (e.g. MLP, RBF networks). The real-valued outputs of such networks can be interpreted as firing rates of natural neurons.

- 3rd generation of neural network models: Networks which employ spiking neurons as computational units, taking into account the precise firing times of neurons for information coding.

The use of spiking neurons promises high relevance for biological systems and, furthermore, might be more flexible for computer vision applications [4]. Spiking neurons with leaky integrator synapses have been used to model image segmentation and binding by synchronization and desynchronization of neuronal group activity. The model, which is called RFSLISSOM, integrates the spiking leaky integrator model with the RF-LISSOM structure, modeling self-organization and
2.2 Models of spiking neurons

Since the works of Santiago Ramon y Cajal and Camillo Golgi, a vast number of theoretical neuron models have been created, with a modern phase beginning with the work of Hodgkin and Huxley [8]. We divide the spiking neuron models into three main classes, namely threshold-fire, conductance based and compartmental models. Because of the nature of this paper we will only cover the class of threshold-fire and specially spike response model (SRM).

The SRM as defined by Gerstner [9] is simple to understand and to implement. The model expresses the membrane potential \( u \) at time \( t \) as an integral over the past, including a model of refractoriness. Let \( F_i = \{ t_f; 1 \leq f \leq n \} \) denote the set of all firing times of neuron \( N_i \) and \( \Gamma_j = \{ t; N_i \text{ is presynaptic to } N_j \} \) denote the set of all presynaptic neuron to \( N_j \).

The state \( U_j(t) \) of neuron \( N_j \) at time \( t \) is given by:

\[
u_j^{(s)} = \sum_{t^{(f)}_{ij} \in \Gamma_j} \eta_j \left( t - t^{(f)}_{ij} \right) + \sum_{i \in I} \sum_{t^{(f)}_{ij} \in F_i} w_{ij} e_{ij} \left( t - t^{(f)}_{ij} - \tau \right)
\]

\[\eta_j \text{ models the potential reset after a spike emission, } w_{ij} \text{ describes the response to presynaptic spikes. For the kernel functions, a choice of usual expressions is given by:}
\]

\[\eta_j(s) = -\vartheta e^{-s/\tau} H(s)
\]

where \( H \) is the Heaviside function, \( \vartheta \) is the threshold and \( \tau \) a time constant defining the decay of the PSP. The function \( e(t) \) is an \( \alpha \)-function as:

\[e(t) = \begin{cases} \frac{t}{\tau} e^{(1-t/\tau)} & \text{for } t > 0 \\ 0 & \text{else} \end{cases}
\]
3. Segmentation Using Spiking Neural Network

However, before building a SNN, we have to explore three important issues: network architecture, information encoding and learning method. After that we will use the SNN to segment images.

3.1 Network architecture

The network architecture consists of a fully connected feedforward network of spiking neurons with connections implemented as multiple delayed synaptic terminals (cf. Figure 2).

The network consists of an input layer, a hidden layer, and an output layer. The first layer is composed of three inputs neurons (RGB values) of pixel. Each node in the hidden layer has a localized activation \( \Phi(x) = \Phi(\|x - C_n\|, \sigma_n) \) where \( \Phi(.) \) is a radial basis function (RBF) localized around \( C_n \) with the degree of localization parameterized by \( \sigma_n \).

Choosing \( \Phi(Z, \sigma) = e^{-\frac{Z^2}{2\sigma^2}} \) gives the Gaussian RBF. This layer transforms real values to temporal values. The activations of all hidden nodes are weighted and sent to the output layer. Instead of a single synapse, with its specific delay and weight, this synapse model consists of many sub-synapses, each one with its own weight and delay \( d^k \).

![Image of network architecture](image)

Figure 2. Network architecture

3.2 Information encoding

The first question that arises when dealing with spiking neurons is how neurons encode information in their spike trains, since we are especially interested in a method to translate an analog value into spikes. We distinguish essentially three different approaches [9] in a very rough categorization:

- Rate coding: the information is encoded in the firing rate of the neurons.
- Temporal coding: the information is encoded by the timing of the spikes.
- Population coding: information is encoded by the activity of different pools (populations) of neurons, where a neuron may participate of several pools.

We have used the temporal encoding proposed by Bohte et al. in [10]. By this method, the input variables are encoded with graded and overlapping activation functions, modeled as local receptive fields. Each neuron of entry is modeled by a local receiving field. A receiving field is a Gaussian function. Each receiving field \( i \) have a center \( C_i \) given by the equation (4) and a width \( \sigma_i \) given by the equation (5) such as: \( m \) is number of receptive fields in each population and \( \gamma = 1.5 \).

\[
\begin{align*}
C_i &= \frac{i - 1.5}{m - 2} \quad (4) \\
\sigma_i &= \frac{1}{\gamma(m - 2)} \quad (5)
\end{align*}
\]

3.3 Learning method

The approach presented here implements the Hebbian reinforcement learning method through a winner-takes-all algorithm [11]. For unsupervised learning, a Winner-Take-All learning rule modifies the weights between the input neurons and the neuron first to fire in the output layer using a time-variant of Hebbian learning: if the start of a PSP at a synapse slightly precedes a spike in the output neuron, the weight of this synapse is increased, as it had significant influence on the spike-time via a relatively large contribution to the membrane potential. Earlier and later synapses are decreased in weight, reflecting their lesser impact on the output neuron’s spike time. For a weight with delay \( d^k \) from neuron \( i \) to neuron \( j \) we use:

\[
\Delta w_{ij} = \eta L(\Delta t_{ij}) \\
\text{And} \\
L(\Delta t) = (1 + \beta)e^{\frac{(\Delta t - \sigma)^2}{2(\kappa - 1)}} - \beta \\
\text{with } \kappa = 1 - \frac{\rho^2}{2\ln(\frac{\rho}{1+\rho})}
\]
The learning window is defined by the following parameters:

- $\alpha$: because of the time constant $\tau$ of the EPSP, a neuron $i$ firing exactly with $j$ does not contribute to the firing of $j$, so the learning window must be shifted slightly to consider this time interval and to avoid reinforcing synapses that do not stimulate $j$.

Inside the neighborhood the weights are increased, otherwise they are decreased.

- $\beta$: this parameter determines the amount by which the weights will be reduced and corresponds to the part of the curve laying outside the neighborhood and below the zero line.

The images obtained are shown in Figure 5:

To show the influence of the number of sub-synapses on the number of areas of the segmented image we had fixed the number of area at exit at 10, the step of training to 0.35, the choice of the base of training is random starting from the image source of 5% and numbers of receiving fields with 18 (6 for each value of intensity) and we varied the number of subsynapses. The images obtained are shown in Figure 6:

To show the influence of the number of receptive fields on the number of classes of the segmented image we had fixed the number of area at exit at 10, the step of training
to 0.35, the choice of the base of training is random starting from the image source of 5%, the number of sub-synapses at 14 and we varied numbers of receiving fields. The images obtained are shown in Figure 7:

![Fig. 7. Segmented image with 4 and 6 receptive fields for each value of intensity.](image)

To show the influence of the percentage of simple training on the number of classes of the segmented image we had fixed the number of area at exit at 10, the step of training to 0.35, the number of sub-synapses at 14 and numbers of receiving fields with 18 (6 for each value of intensity) and we varied the number of percentage of simple training. The images obtained are shown in Figure 8:

![Fig. 8. Segmented image with 5 and 20% of simple training.](image)

5. Conclusion
In this paper we applied spiking neural networks to image segmentation. At first, the network is build, a subset of the image pixel is taken to be learned by the network and finally the SNN processes the rest of the image to have as a result an important number of classes quantized the image.

References


