Image Processing with Spiking Neuron Networks

Boudjelal Meftah, Olivier Lézoray, Soni Chaturvedi,
Aleefia A. Khurshid, and Abdelkader Benyettou

Abstract. Artificial neural networks have been well developed so far. First two generations of neural networks have had a lot of successful applications. Spiking Neuron Networks (SNNs) are often referred to as the third 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 up 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 chapter, we present how SNN can be applied with efficacy in image clustering, segmentation and edge detection. Results obtained confirm the validity of the approach.

1 Introduction

There are many artificial neural networks that can be successfully used in image processing tasks, the most prominent of them are networks, commonly known by

Boudjelal Meftah
Equipe EDTEC, Université de Mascara, Mascara, Algérie

Olivier Lézoray
Université de Caen Basse-Normandie, GREYC UMR CNRS 6072, 6 Bd. Maréchal Juin, F-14050, Caen, France

Soni Chaturvedi · Aleefia A. Khurshid
Priyadarshini Institute of Engineering and Technology, Nagpur Maharashtra, India

Abdelkader Benyettou
Laboratoire Signal Image et Parole, Université Mohamed Boudiaf, Oran, Algérie

X.-S. Yang (Ed.): Artif. Intell., Evol. Comput. and Metaheuristics, SCI 427, pp. 525–544.
springerlink.com

now as Spiking Neural Networks (SNN) [1]. Highly inspired from natural computing in the brain and recent advances in neuroscience, they derive their strength and interest from an accurate modeling of synaptic interactions between neurons, taking into account the time of spike firing. SNNs overcome the computational power of neural networks made of threshold or sigmoidal units [2].

The use of spiking neurons promises high relevance for biological systems and, furthermore, might be more flexible for computer vision applications [3]. Wu et al. [4] proposed hierarchical spiking neural networks to process visual stimuli, in which multiple overlapped objects are represented by different orientation bars. Their model segments images and binds their pixels to form shapes of objects using local excitatory lateral connections. Girau et al. [5] had implemented integrate-and-fire neurons to the standard LEGION (Local Excitatory Global Inhibitory Oscillator Network) architecture to segment grey-level images. In order to segment images, the LEGION model groups oscillators that receive their input from similar features in an image. Oscillators group together by synchronization of their phase thanks to excitatory connections, and they get desynchronized from other groups of oscillators by means of global inhibition. Buhmann et al. [6] proposed a network of leaky integrate-and-fire neurons to segment gray-scale images. The network architecture with local competition between neurons that encode segment assignments of image blocks is motivated by an histogram clustering approach to image segmentation. Rowcliffe et al. [7] had developed an algorithm to produce self-organisation of a purely excitatory network of Integrate-and-Fire neurons. Pixels from an image are used as scalar inputs for the network, and segmented as the oscillating neurons are clustered into synchronised groups.

In this chapter, a spiking neural network is used to cluster images, segment images and detect edges with Hebbian based winner-take-all learning. We seek, through a series of experiments carried out, the best parameters of the SNN network to have a good segmentation and a fine detection of contours.

The chapter is organized as follows: in first Section, related works are presented within the literature of spiking neural network (SNNs). Second Section is the central part of the chapter and is devoted to the description of the architecture of a spiking neural network with multiple delay connections, the encoding mechanism for converting the real valued inputs into time vectors and the learning rule. The results and discussions of the experimental activity are reported in the third Section. Last Section concludes.

2 Overview of Spiking Neuron Networks

Spiking neural networks (SNNs) are a class of ANNs that are increasingly receiving the attention as both a computationally powerful and biologically plausible mode of computation [8], [9]. SNNs model the precise time of the spikes fired by a neuron, as opposed to the conventional neural networks which model only the average firing rate of the neurons. It is proved that the neurons that convey information by

individual spike times are computationally more powerful than the neurons with sigmoidal activation functions [10].

2.1 Artificial Neuron Generations

Wolfgang Maass [11] delineates past and current artificial neural network research into three generations and makes the following observations.

The first generation is based on the McCulloch-Pitts neuron (also known as a perceptron or a threshold-gate) as the basic computation unit. Models of the first generation, such as the multi-layer perceptron, use digital input and output, usually binary or bipolar. Any Boolean function can be computed by some multi-layer perceptron with a single hidden layer.

The second generation is based on computation units (neurons) that use an activation function of a continuous set of possible output values. Commonly, these activation functions are the sigmoid, or the hyperbolic tangent. Second generation neural networks, like first generation networks, can compute arbitrary boolean functions (after using a threshold). Second generation networks can compute certain boolean functions with fewer neurons than first generation neurons. Also, second generation networks with one hidden layer can approximate any continuous, analog function arbitrarily well. Important to many implementations is the fact that second generation networks support learning algorithms based on gradient descent, such as error back-propagation.

The third generation of artificial neural networks is based on spiking neurons, or integrate and fire neurons. These neurons use recent insights from neurophysiology, specifically the use of temporal coding to pass information between neurons. These networks, like those of the second generation, can approximate continuous functions arbitrarily well, but with temporally encoded inputs and outputs [11], [12]. Further, there are function that require fewer neurons in a pulsed neural net to approximate than would be needed in a second generation network [11].

All three of these generations are simplifications of what is known about the physiology of biological neurons but the third generation is the model with the highest fidelity.

2.2 Spiking Neuron Networks Architecture

The network architecture consists in a feedforward network of spiking neurons with multiple delayed synaptic terminals (Fig. 1.a). The neurons in the network generate action potentials, or spikes, when the internal neuron state variable, called ”membrane potential”, crosses a threshold ϑ. The relationship between input spikes and the internal state variable is described by the Spike Response Model (SRM), as introduced by Gerstner [9]. Depending on the choice of suitable spike-response functions, one can adapt this model to reflect the dynamics of a large variety of different spiking neurons.

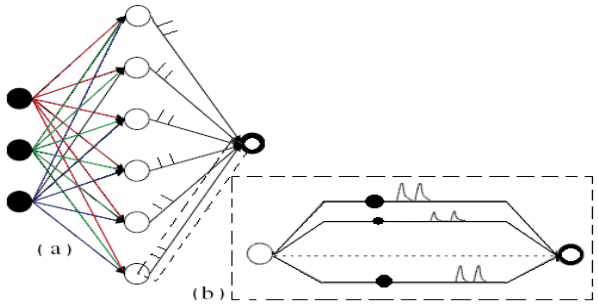


Fig. 1 (a) Spiking neural network architecture; (b) Multiple synapses transmitting multiple spikes.

Formally, a neuron j, having a set Γ_j of immediate predecessors (pre-synaptic neurons), receives a set of spikes with firing times t_i, $i \in \Gamma_j$. Any neuron generates at most one spike during the simulation interval, and fires when the internal state variable reaches a threshold ϑ. The dynamics of the internal state variable $x_j(t)$ are determined by the impinging spikes, whose impact is described by the spike-response function $\varepsilon(t)$ modeling a simple α-function weighted by the synaptic efficacy w_{ij}:

$$x_j(t) = \sum_{i \in \Gamma_j} \sum_{k=1}^{m} w_{ij}^k \varepsilon(t - t_i - d^k) \tag{1}$$

The height of the post-synaptic potential (PSP) is modulated by the synaptic weight w_{ij} to obtain the effective post-synaptic potential (PSP).

$\varepsilon(t)$ a spike-response function shaping a PSP and τ models the membrane potential decay time constant that determines the rise and decay time of the PSP. Figure 2 illustrates and equation (2) represents one of the most popular mathematical spike response models.

$$x_j(t) = \sum_{i \in \Gamma_j} \sum_{k=1}^{m} w_{ij}^k \varepsilon(t - t_i - d^k) \tag{2}$$

In the network as introduced in [13], an individual connection consists in a fixed number of m synaptic terminals, where each terminal serves as a sub-connection that is associated with a different delay and weight (Fig. 1.b). The delay d^k of a synaptic terminal k is defined by the difference between the firing time of the pre-synaptic neuron, and the time the post-synaptic potential starts rising.

We describe a presynaptic spike at a synaptic terminal k as a PSP of standard height with delay d^k. The unweighted contribution of a single synaptic terminal to the state variable is then given by:

$$y_i^k(t) = \varepsilon(t - t_i - d^k) \tag{3}$$

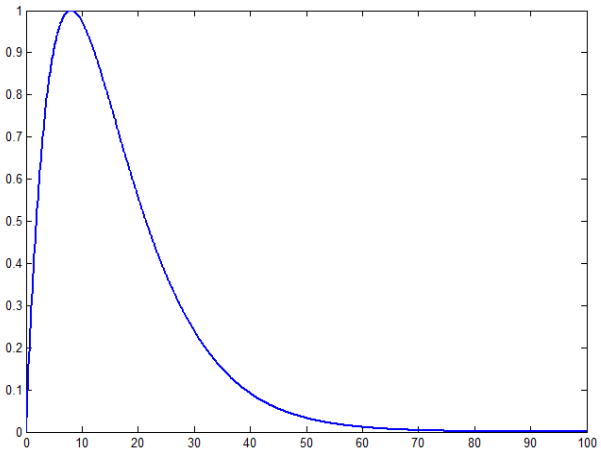


Fig. 2 Spike response Function, Postsynaptic Potential is Excitatory (EPSP), $\tau = 8$.

The time t_i is the firing time of pre-synaptic neuron i, and d^k the delay associated with the synaptic terminal k.

Extending equation Eq.1 to include multiple synapses per connection and inserting equation (Eq.3), the state variable x_j of neuron j receiving input from all neurons i can then be described as the weighted sum of the pre-synaptic contributions:

$$x_j(t) = \sum_{i \in \Gamma_j} \sum_{k=1}^{m} w_{ij}^k y_i^k(t) \tag{4}$$

Where w_{ij}^k denotes the weight associated with synaptic terminal k. The firing time t_j of neuron j is determined as the first time when the state variable crosses the threshold ϑ: $x_j(t) \geq \vartheta$. Thus, the firing time t_j is a non-linear function of the state variable $x_j : t_j = t_j(x_j)$.

2.3 Neural Coding Schemes

The spiking model is fundamentally different than previous generations of artificial neurons. Most importantly, the information passed by spikes can only be that of the relative timing between them. Thus the passing of useful information across a spiking net requires conversion from other forms (typically analog) to temporal data. The first question that arises when dealing with spiking neurons is how neurons encode information in their spike trains. Basically, there are three different coding methods: rate coding, temporal coding and population coding (see [14] for reviews).

2.3.1 Rate Coding

Rate coding is a traditional coding scheme, assuming that most, if not all, information about the stimulus is contained in the firing rate of the neuron. Because the sequence of action potentials generated by a given stimulus varies from trial to trial, neuronal responses are typically treated statistically or probabilistically. They may be characterized by firing rates, rather than as specific spike sequences. Consequently, rate coding is inefficient but highly robust with respect to the Inter-Spike Interval Noise (ISI Noise) [15].

2.3.2 Temporal Coding

When precise spike timing or high-frequency firing-rate fluctuations are found to carry information, the neural code is often identified as a temporal code [16]. A number of studies have found that the temporal resolution of the neural code is on a millisecond time scale, indicating that precise spike timing is a significant element in neural coding [17], [18]. Temporal codes employ those features of the spiking activity that cannot be described by the firing rate. The temporal structure of a spike train or firing rate evoked by a stimulus is determined both by the dynamics of the stimulus and by the nature of the neural encoding process. Stimuli that change rapidly tend to generate precisely timed spikes and rapidly changing firing rates no matter what neural coding strategy is being used. Temporal coding refers to temporal precision in the response that does not arise solely from the dynamics of the stimulus, but that nevertheless relates to properties of the stimulus. The interplay between stimulus and encoding dynamics makes the identification of a temporal code difficult.

2.3.3 Population Coding

Population coding is a method to represent stimuli by using the joint activities of a number of neurons. In population coding, each neuron has a distribution of responses over some set of inputs, and the responses of many neurons may be combined to determine some value about the inputs. From the theoretical point of view, population coding is one of a few mathematically well-formulated problems in neuroscience. It grasps the essential features of neural coding and yet, is simple enough for theoretic analysis [17]. Experimental studies have revealed that this coding paradigm is widely used in the sensor and motor areas of the brain.

3 Spiking Neuron Networks for Clustering, Segmentation and Edge Detection

However, before building a SNN, we have to explore three important issues: information coding, learning method and network architecture for each operation of image processing. After that we will use the SNN to cluster images, segment images and detect edges.

In order to simplify the model, we assume that before a neuron generates a spike, it has been at its resting state for a sufficiently long time such that the back propagation action potential is negligible. Also, in one learning cycle, each neuron fires only once.

In this section, we will review how to encode real input data temporally, the architecture and learning of spiking neural networks.

3.1 Information Coding

Spike timing encoding is the process of transforming measurements of sensory inputs into a spike train representation, which is the form of input a spiking neuron can handle. Thus the multidimensional raw data, which consists of real values, needs to be mapped into a temporal space before being fed to the network.

Bohte et al. [19], proposed the population coding method that encodes an input variable using multiple overlapping Gaussian Receptive Fields (RF). Gaussian RF are used to generate firing times from real values. The range of the data is first calculated, and then each input feature is encoded with a population of neurons that cover the whole data range. For a range $[I_{Max}..I_{Min}]$ of a variable, which is also called the coding interval, a set of m Gaussian RF neurons are used. The center C_i and the width σ_i of each RF neuron i are determined by the following equations:

$$C_i = I_{min} + \left(\frac{2i-3}{2}\right)\left(\frac{I_{max}-I_{min}}{m-2}\right) \tag{5}$$

$$\sigma_i = \frac{1}{\gamma}\frac{I_{max}-I_{min}}{m-2} \tag{6}$$

Where m is number of receptive fields in each population and a value of 1.5 is used for the variable γ.

While converting the activation values of RF into firing times, a threshold ϑ has been imposed on the activation value. A receptive field that gives an activation value less than this threshold will be marked as not-firing and the corresponding input neuron will not contribute to the post-synaptic potential.

An illustration of this encoding scheme is shown in Figure 3, which shows the firing times resulting from the encoding of the real value 0.3 using six RF. In this example, assuming that the time unit is millisecond, the value 0.3 was encoded with six neurons by delaying the firing of neurons 1 (5.564ms), 2 (1.287ms), 3 (0.250ms), 4 (3.783ms) and 5 (7.741ms). Neuron 6 does not fire at all, since the delay is above threshold 9ms and stand in the no firing zone.

3.2 Spiking Neuron Networks for Unsupervised Learning Method

Our goal is that after learning, the spiking neural network can do clustering, segmentation and edge detection by using the firing time of postsynaptic neurons associated

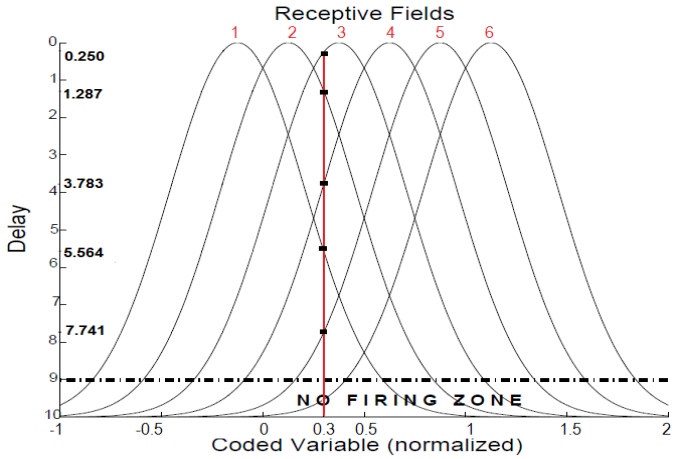


Fig. 3 Coding of a real value, and its corresponding firing time.

with each input pattern. The approach presented here implements the Hebbian reinforcement learning method through a winner-take-all algorithm [20], [21].

For unsupervised learning, a Winner-Takes-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. The synaptic weights should be randomly initialized. When an input pattern is presented to the network, neurons are expected to fire. The first neuron to fire is called the winner of the competitive process. Only the weights of the winner neuron are updated using a Hebbian learning rule $L(\Delta t)$.

In a clustering task, the learning process consists mainly of adapting the time delays, so that each output neuron represents an RBF center. This goal is achieved using a learning function (Fig.4), which is defined as a function of the time interval Δt_{ij} between the firing times t_i and t_j. This function controls the learning process by updating the weights based on this time difference, as shown in equation (7), where Δw_{ij} is the amount by which the weights w_{ij} are increased or decreased and η is the learning rate.

$$\Delta w_{ij}^{k} = \eta L(\Delta t_{ij}) \tag{7}$$

The learning function is a Gaussian curve defined by the equation (8). It reinforces the synapse between neurons i and j if $\Delta t_{ij} < 0$, and depresses the synapse if $\Delta t_{ij} > 0$ (Gerstner, 2002, Leibold,2001).

$$L(\Delta t) = (1+b)e^{\frac{(\Delta t-c)^2}{2(k-1)}} - b \tag{8}$$

with
$k = 1 - \frac{\nu^2}{2ln\frac{b}{1+b}}$

where: $L(.)$ is the learning function; η is the learning rate; ν determines the width of the learning window; Δt is the difference between the arriving of the spike and the fire of neuron j; b determines the negative update given to a neuron; c fixes the peak of the learning function; w_{ij}^k is the increase of the k^{th} connection between neurons i and j. The weights are limited to the range 0 to w_{max}, the maximum value that a weight can take.

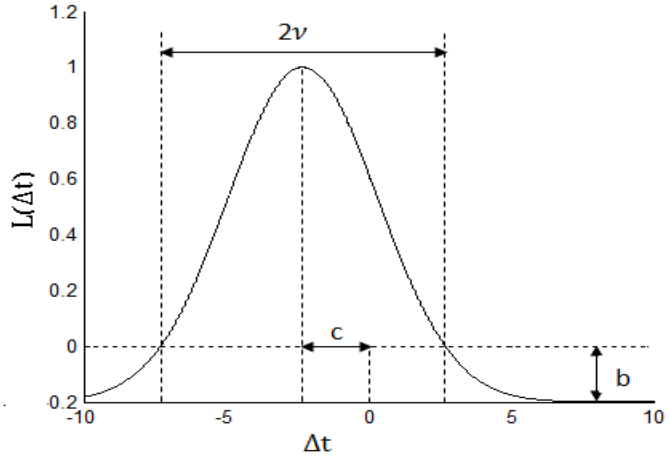


Fig. 4 Gaussian learning function with b =0.2 c=-2.3 and $\vartheta = 5$.

It is important to remark that this system is extremely sensible to the b parameter, since a range from 0 to -0.3 leads to completely different dynamics in the learning process. When the synaptic weight sum is stable (b=-0.007), the firing time tends to evolute only according to the competitive learning process [23].

3.3 SNN Architecture for Clustering, Segmentation and Edge Detection

3.3.1 SNN Architecture for Clustering Images

The model for a spiking neuron which we use in the following is the spike response model with short term memory. Here we consider a network of such spiking architecture in a fully connected feedforward with connections implemented as multiple delayed synaptic terminals (Fig. 5).

The network consists in an input layer, a hidden layer, and an output layer. The first layer is composed of three inputs neurons (RGB values) of pixels. Each node in the hidden layer has a localized activation $\phi^n = \phi(\|X - C_n\|, \sigma_n)$ where $\phi^n(.)$ is

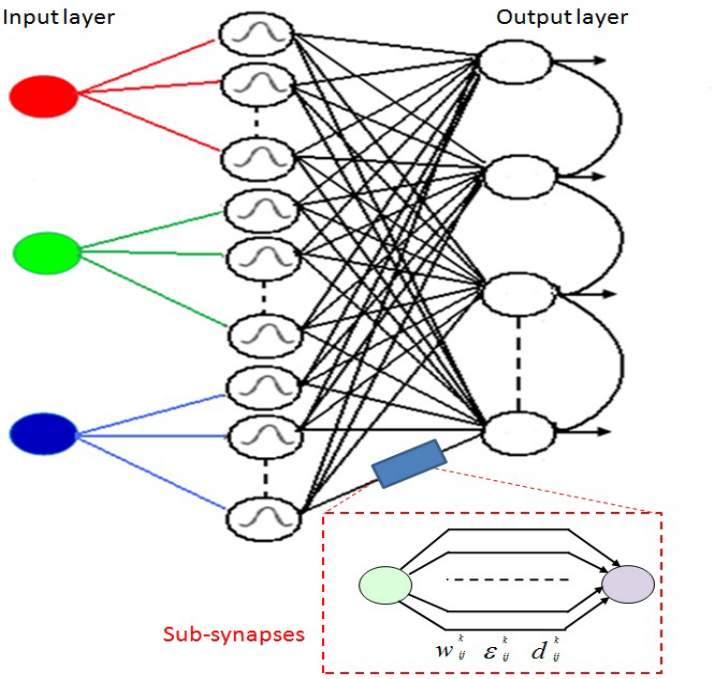


Fig. 5 Network topology for clustering and Segmentation images.

a radial basis function (RBF) localized around C_n with the degree of localization parameterized by σ_n. Choosing $\phi(Z,\sigma) = exp\frac{Z^2}{2\sigma^2}$ gives the Gaussian RBF. This layer transforms real values to temporal values.

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, as shown in Fig.1.b. The use of multiple synapses enables an adequate delay selection using the learning. For each multiple synapse connecting neuron i to neuron j, with s subsynapses, the resulting PSP is given by equation (1). The total contribution of all presynaptic neurons is then given by equation (4). The neuron model implemented is the SRM_0 (Gerstner, 2002), with a strictly excitatory PSP. The delays d^k are fixed for all sub-synapse k, varying from zero in 1ms fixed intervals. $\varepsilon(t)$ modeling a simple α-function.

3.3.2 SNN Architecture for Cell Segmentation

In this section, before introducing the architecture used, we give a quick review of cellular segmentation methods.

Image analysis in the field of cancer screening is a significant tool for cytopathology [24]. Two principal reasons can be highlighted. First, the quantitative analysis

of shape and structure of nuclei coming from microscopic color images brings to the pathologist valuable information for diagnosis assistance. Second, the quantity of information that the pathologist must deal with is large, in particular when the number of cancer screening increases. That is why; a segmentation scheme for microscopic cellular imaging must be efficient for reliable analysis.

Many cellular segmentation methods have been presented. They include watershed [25], region-based [26] and threshold-based methods [27]. Application of active contour has been widely investigated for cell segmentation (Karlosson, 2003). Cells stained with Papanicolaou international staining make it possible to classify the color pixels among three classes [29]: background, cytoplasm or nucleus. However, this classification cannot be perfect. Indeed, a fraction on nuclei pixels have the same color then cytoplasm pixels because of the variability of the nuclei according to the type of the cells and to the chromatin distribution. Moreover, for some cytopathologies, the mucus present in the background has the same color as some cells (cytoplasm and nucleus).

For cell segmentation, The network architecture consists in a fully connected feedforward network of spiking neurons with connections implemented as multiple delayed synaptic terminals. We consider two different topologies for unsupervised and supervised learning. For unsupervised learning, the SNN performs its learning directly on the pixels of the image to classify. For unsupervised learning, a reference data set of pixels from different images is used for learning. In both topologies depicted in Fig. 6(a) and Fig. 6(b), the network consists in an input layer, a hidden layer, and an output layer. The first layer is composed of RGB values of pixels. Each node in the hidden layer has a localized activation $\phi^n = \phi(\|X - C_n\|, \sigma_n)$ where $\phi^n(.)$ is a radial basis function (RBF) localized around C_n with the degree of localization parameterized by σ_n. Choosing $\phi(Z, \sigma) = exp\frac{Z^2}{2\sigma^2}$ gives the Gaussian RBF. This layer transforms the RGB values of pixels in first layer to temporal values. Third layer consist in class outputs (cell background, cytoplasm and nuclei).

The network architecture consists in a fully connected feedforward network of spiking neurons with connections implemented as multiple delayed synaptic terminals. We consider two different topologies for unsupervised and supervised learning. For unsupervised learning, the SNN performs its learning directly on the pixels of the image to classify. For unsupervised learning, a reference data set of pixels from different images is used for learning. In both topologies depicted in Figure 6(a) and Figure 6(b), the network consists of an input layer, a hidden layer, and an output layer. The first layer is composed of RGB values of pixels. Each node in the hidden layer has a localized activation n where n (.) is a radial basis function (RBF) localized around cn with the degree of localization parameterized by n. Choosing z gives the Gaussian RBF. This layer transforms the RGB values of pixels in first layer to temporal values. Third layer consist in class outputs (cell background, cytoplasm and nuclei).

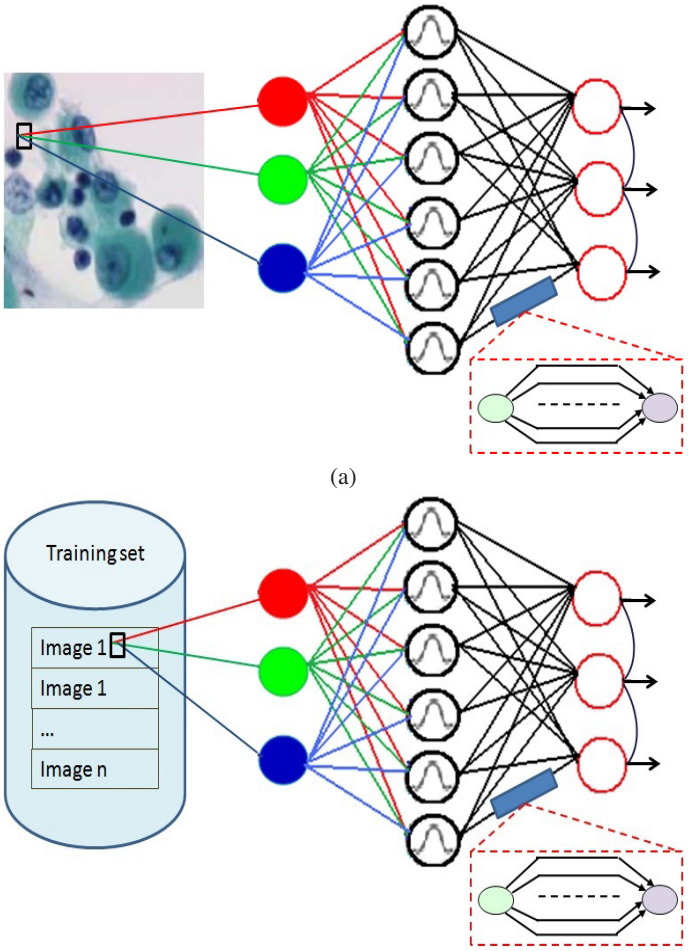


Fig. 6 (a) Network topology for unsupervised training; (b) Network topology for supervised training.

3.3.3 SNN Architecture for Edge Detection

First image of a microscopic cell is segmented with spiking neural network. Once the segmentation done, we will record the activity of each output neuron which gives for each input pixel an output binary 1 if the neuron is active or 0 if the neuron is inactive. The result of binary matrices activation of output neurons can be represented by binary images containing the edges detected by these neurons for each class. Fusion is then made to have the final edges by superimposing the resulting images (Figure 7).

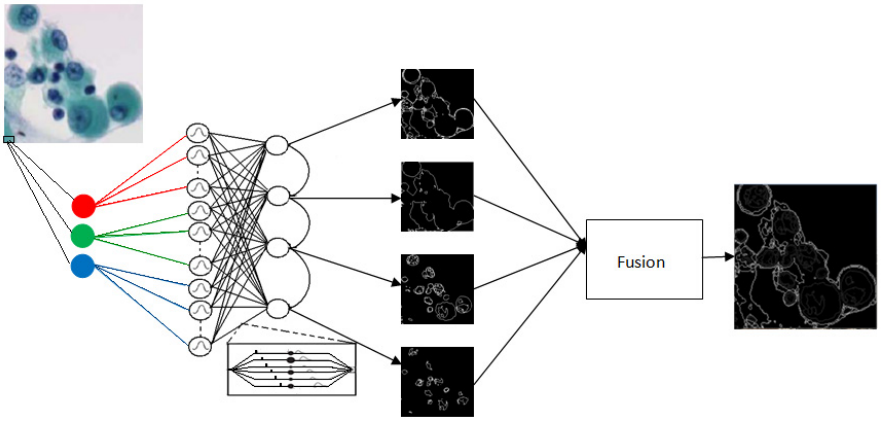


Fig. 7 SNN edge network topology.

4 Experimental Results and Discussion

4.1 Clustering Images

Based on the work in [30], several experiments are carried out by changing the number of synapses, the number of receptive fields and the size of training corpus to select the best network parameters on a set of 50 images taken from the Berkeley database (Martin, 2001). The best architecture for a mean quadratic error of $87.352^{+}_{-}[28.747, 39.319]$ has the following parameters:

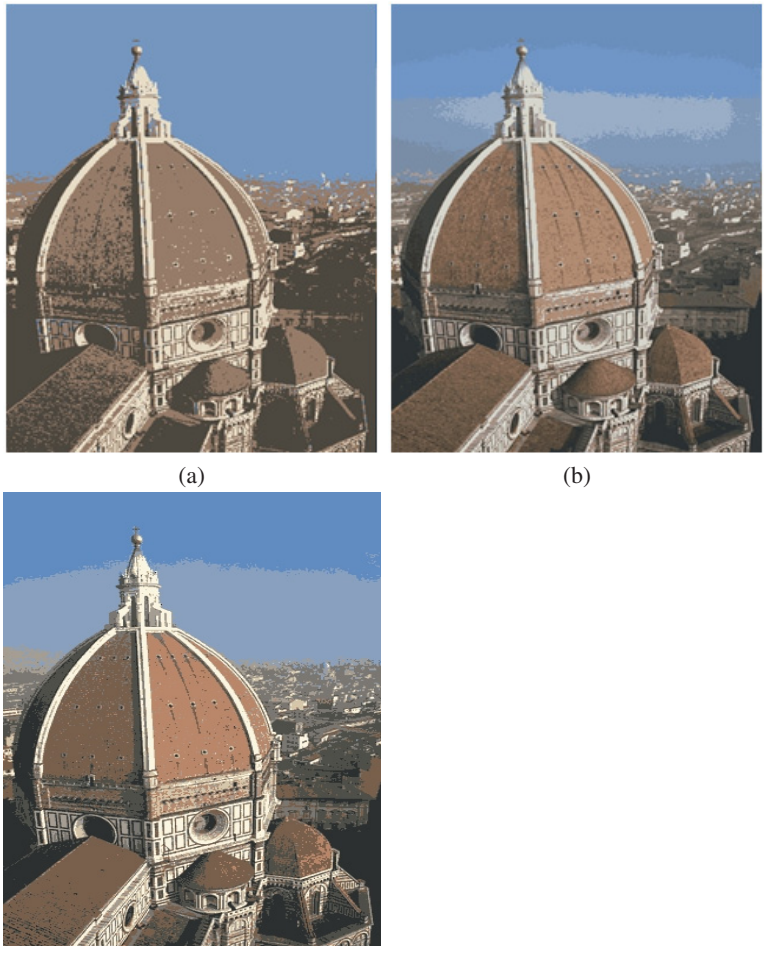
Table 1. Best parameter of the SNN.

Receptive field	Subsynapse	Threshold ϑ	Training set	η	τ	υ	b	c
8	12	9	20%	0.0025	3	5	-0.007	-2.3

To compare the result of clustering with others models, we had used the neural network SOM and Kmeans. The clustering image with Kmeans is shown in Figure 8.a, with SOM neural network is shown below in Figure 8.b and with spiking neural networks in Figure 8.c.

Evaluation Methods

To see if clustering is close to the original image, an error metric is needed. The error between the original image and the quantized image is generally used. For this evaluation we had used the Peak Signal Noise Ratio (PSNR), the Mean Square Error (MSE), the Mean Absolute Error (MAE) and Normalized Color Difference (NCD) are therefore considered to evaluate the clustering. Table 2 summarizes the evaluation obtained for each resulting image in Figure 8.



(a) (b)

(c)

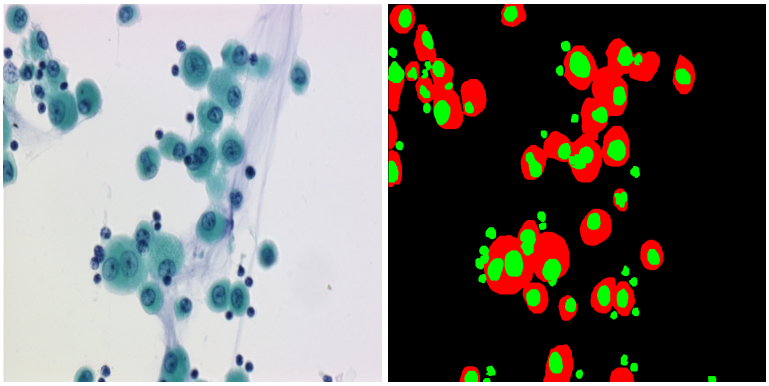
Fig. 8 (a) Clustering image with Kmeans (b) Clustering image with SOM (c) Clustering image with SNN.

Table 2. Clustering evaluation (best rates bold faced).

	clustering with Kmeans	clustering with SOM	clustering with SNN
PSNR	51.283	62.574	**65.404**
MSE	385.37	124.596	**93.845**
MAE	16.643	7.960	**7.841**
NCD	0.152	0.110	**0.083**

4.2 Cell Segmentation

For the considered class of microscopic images, a microscopy expert has to choose judicious images that well describe the whole segmentation problem (a ground truth). This ground truth database can be used for the learning step and also as a reference segmentation to evaluate the relevance of an automatic segmentation. In the sequel, we will consider a publicly available database [32] of 8 microscopic images of bronchial tumors (752 x 574 pixels). The pixels of these images have to be classified into one of the three following classes background, cell cytoplasm and cell nucleus. Figures 9.a and 9.b shows a microscopic color image and its ground truth. Pixel dataset has been split to produce training, validation and test sets.



(a) (b)

Fig. 9 (a) Original image; (b) Ground truth.

Images in Figure 10 show segmentation results with our segmentation scheme for the parameter of Table 1 in comparison with the expert segmentation. It is worth to note that the mucus present in all images is correctly identified as background [33].

Evaluation Methods

To evaluate our approach, we use several classification rates. These classifications rates are expressed as follows:

$$R_0 = \frac{Number\ of\ pixels\ well\ classified}{Number\ of\ pixels\ of\ the\ image} \tag{9}$$

$$R_1 = \frac{Number\ of\ nuclei\ pixels\ well\ classified}{Number\ of\ nuclei\ pixels\ of\ the\ image} \tag{10}$$

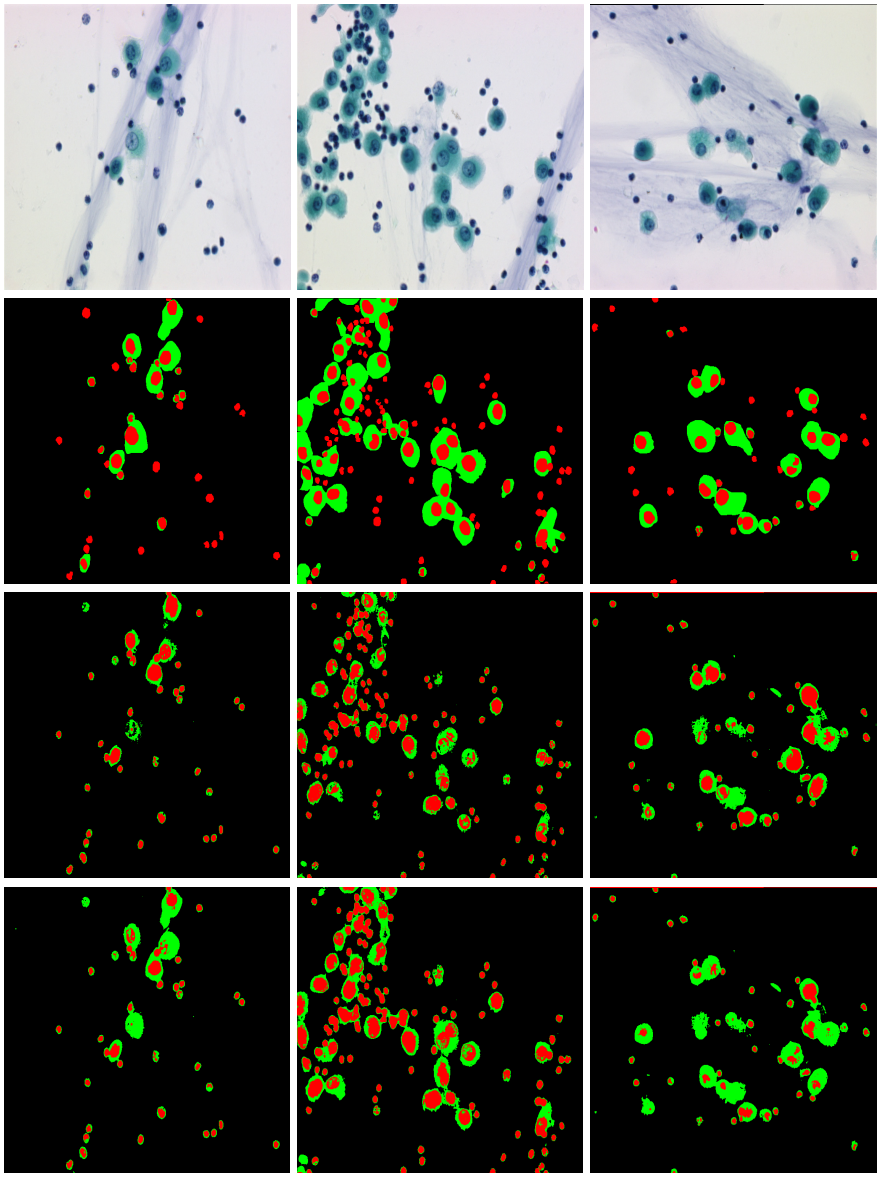


Fig. 10 Cell microscopic images (First row); expert segmentation (Second row); segmentation produced by unsupervised training (Third row) and segmentation produced by supervised training (Fourth row).

$$R_2 = \frac{Number\ of\ background\ pixels\ well\ classified}{Number\ of\ background\ pixels\ of\ the\ image} \tag{11}$$

$$R_3 = \frac{R_1 + R_2}{2} \tag{12}$$

Results in Table 3 show that SNN with supervised training has the best classification accuracies as compared to SNN with unsupervised training.

Table 3. Classification rates (best rates bold faced).

	SNN with unsupervised training	SNN with supervised training
R_0	89.07%	**94.27**%
R_1	69.57%	**80.37**%
R_2	94.55%	**99.06**%
R_3	82.06%	**89.71**%

Table 4 presents a comparison of the classification accuracies obtained by Meurie et al. [32] for different classifiers as well as with our SNN supervised training. Our approach clearly outperforms all these state-of-the-art methods.

Table 4. Segmentation rates and comparison with Meurie et al. approaches [32], with best rates bold faced.

Classifier	R_1
SVM	**72.2**%
Bayes	**74.6**%
K-means	**74.4**%
MLP	**73**%
Fisher 1	**72.3**%
KNN	**70**%
Supervised SNN	**80.37**%

4.3 Edge Detection

The result of edge detection and a comparison with other methods of edge detection is obtained in Figure 11.

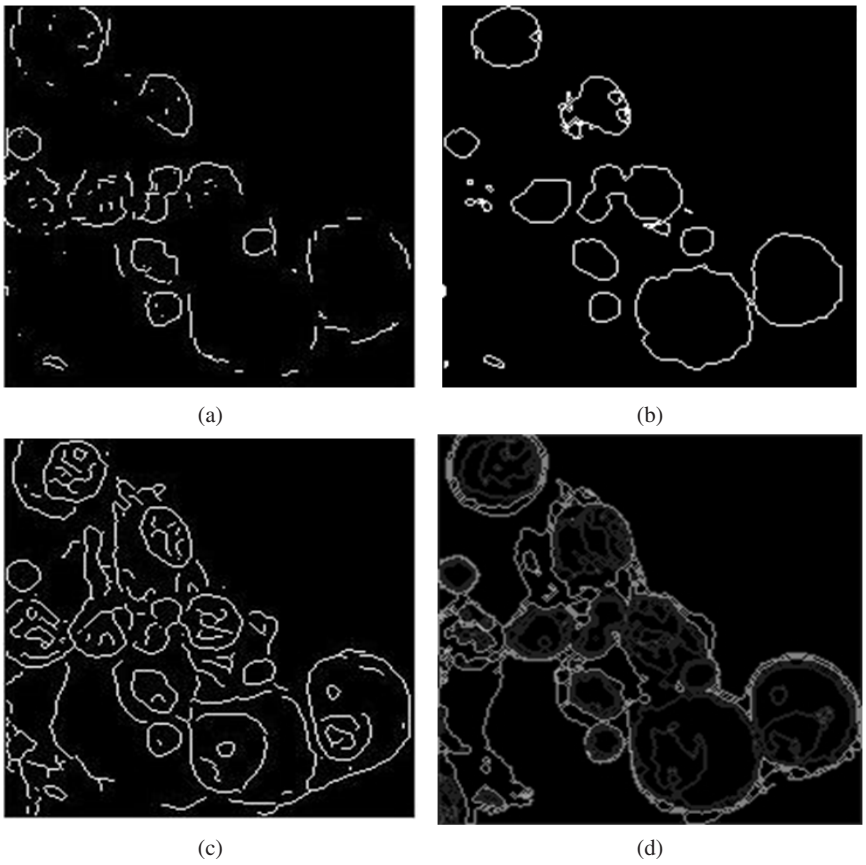


Fig. 11 (a) Edge detection with Prewitt (b) Edge detection with morphology black top hat (c) Edge detection with Canny (d) Edge detection with SNN.

5 Conclusion

In this chapter, we have applied a Spiking Neural Network (SNN) Model for image clustering, segmentation and edge detection. To use a SNN for these problems, we have addressed the issue of parameter selection. We have focused our study on the keys parameters: network architecture (number of subsynapses, receptive fields, output neurons) and learning parameters (training step, size of the training data base, peak of the learning function). These parameters are set up for each specific image problem problems. Results have shown that a careful setting of parameters is required to obtain efficient results. Future works will concern the application of this works to video processing.

References

1. Ghosh-Dastidar, S., Adeli, H.: Third generation neural networks: Spiking neural networks. In: Yu, W., Sanchez, E.N. (eds.) Advances in Computational Intelligence. AISC, vol. 61, pp. 167–178. Springer, Heidelberg (2009)
2. Paugam-Moisy, H., Bohte, S.M.: Computing with Spiking Neuron Networks. In: Kok, J., Heskes, T. (eds.) Handbook of Natural Computing. Springer, Heidelberg (2009)
3. Thorpe, S. J., Delorme, A., VanRullen, R. : Spike-based strategies for rapid processing. Neural Networkss 14(6-7), 715–726 (2001)
4. Wu, Q.X., McGinnity, M., Maguire, L.P., Belatreche, A., Glackin, B.: Processing visual stimuli using hierarchical spiking neural networks. Neurocomputing 71(10-12), 2055–2068 (2008)
5. Girau, B., Torres-Huitzil, C.: FPGA implementation of an integrate-and-fire LEGION model for image segmentation. In: European Symposium on Artificial Neural Networks, ESANN 2006, pp. 173–178 (2006)
6. Buhmann, J., Lange, T., Ramacher, U.: Image Segmentation by Networks of Spiking Neurons. Neural Computation 17(5), 1010–1031 (2005)
7. Rowcliffe, P., Feng, J., Buxton, H.: Clustering within Integrate-and-Fire Neurons for Image Segmentation. In: Dorronsoro, J.R. (ed.) ICANN 2002. LNCS, vol. 2415, pp. 69–74. Springer, Heidelberg (2002)
8. Maass, W.: On the relevance neural networks. MIT Press, London (2001)
9. Gerstner, W., Kistler, W.M.: Spiking neuron models. Cambridge University Press (2002)
10. Gerstner, W., Kistler, W.: Mathematical formulations of Hebbian learning. Biological Cybernetics 87, 404–415 (2002)
11. Maass, W.: Networks of Spiking Neurons: The Third Generation of Neural Network Models. Neural Networks 10(9), 1659–1671 (1997)
12. Maass, W.: Computing with spiking neurons. In: Maass, W., Bishop, C.M. (eds.) Pulsed Neural Networks, MIT Press, Cambridge (1999)
13. NatschlNager, T., Ruf, B.: Spatial and temporal pattern analysis via spiking neurons. Network: Comp. Neural Systems 9(3), 319–332 (1998)
14. Averbeck, B., Latham, P., Pouget, A.: Neural correlations, population coding and computation. Nature Reviews Neuroscience 7, 358–366 (2006)
15. Stein, R., Gossen, E., Jones, K.: Neuronal variability: noise or part of the signal? Nature Reviews Neuroscience 6, 389–397 (2005)
16. Dayan, P., Abbott, L.F.: Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems. The MIT Press, Cambridge (2001)
17. Butts, D.A., Weng, C., Jin, J., Yeh, C., Lesica, N.A., Alonso, J.M., Stanley, G.B.: Temporal precision in the neural code and the timescales of natural vision. Nature 449, 92–95 (2007)
18. Bohte, S.M.: The Evidence for Neural Information Processing with Precise Spike-times: A Survey. Natural Computing 3(2), 195–206 (2004)
19. Bohte, S.M., La Poutre, H., Kok, J.N.: Unsupervised clustering with spiking neurons by sparse temporal coding and Multi-Layer RBF Networks. IEEE Transactions on Neural Networks 13(2), 426–435 (2002)
20. Oster, M., Liu, S.C.: A winner-take-all spiking network with spiking inputs. In: Proceedings of the 11th IEEE International Conference on Electronics, Circuits and Systems (ICECS 2004), vol. 11, pp. 203–206 (2004)
21. Gupta, A., Long, L.N.: Hebbian learning with winner take all for spiking neural networks. In: IEEE International Joint Conference on Neural Networks (IJCNN), pp. 1189–1195 (2009)

22. Leibold, C., Hemmen, J.L.: Temporal receptive fields, spikes, and Hebbian delay selection. Neural Networks 14(6-7), 805–813 (2001)
23. da Silva Simões, A., Costa, A.H.R.: A Learning Function for Parameter Reduction in Spiking Neural Networks with Radial Basis Function. In: Zaverucha, G., da Costa, A.L. (eds.) SBIA 2008. LNCS (LNAI), vol. 5249, pp. 227–236. Springer, Heidelberg (2008)
24. Knesek, E.A.: Roche image analysis system. Acta Cytologica 40(1), 60–66 (1996)
25. Lezoray, O., Cardot, H.: Cooperation of pixel classification schemes and color watershed: a Study for Microscopical Images. IEEE Transactions on Images Processing 11(7), 738–789 (2002)
26. Mouroutis, T., Roberts, S.J., Bharath, A.A.: Robust cell nuclei segmentation using statistical modeling. BioImaging 6, 79–91 (1998)
27. Wu, H.S., Barba, J., Gil, J.: Iterative thresholding for segmentation of cells from noisy images. J. Microsc. 197, 296–304 (2000)
28. Karlsson, A., Stråhlén, K., Heyden, A.: Segmentation of Histopathological Sections Using Snakes. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 595–602. Springer, Heidelberg (2003)
29. Papanicolaou, G.N.: A new procedure for staining vaginal smears. Science 95, 432 (1942)
30. Meftah, B., Benyettou, A., Lezoray, O., Wu, Q.X.: Image clustering with spiking neuron network. In: IEEE World Congress on Computational Intelligence, International Joint Conference on Neural Networks (IJCNN 2008), pp. 682–686 (2008)
31. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int Conf. Computer Vision, vol. 2, pp. 416–423 (2001)
32. Meurie, C., Lezoray, O., Charrier, C., Elmoataz, A.: Combination of multiple pixel classifiers for microscopic image segmentation. IASTED International Journal of Robotics and Automation 20(2), 63–69 (2005)
33. Meftah, B., Lezoray, O., Lecluse, M., Benyettou, A.: Cell Microscopic Segmentation with Spiking Neuron Networks. In: Diamantaras, K., Duch, W., Iliadis, L.S. (eds.) ICANN 2010, Part I. LNCS, vol. 6352, pp. 117–126. Springer, Heidelberg (2010)