20

Analog-Digital Self-Learning Fuzzy Spiking Neural Network in Image Processing Problems

Artem Dolotov and Yevgeniy Bodyanskiy
Kharkiv National University of Radio Electronics
Ukraine

1. Introduction

Computational intelligence provides a variety of means that can perform complex image processing in a rather effective way. Among them, self-learning systems, especially self-learning artificial neural networks (self-organizing maps, ART neural networks, 'Brain-State-in-a-Box' neuromodels, etc.) (Haykin, 1999) and fuzzy clustering systems (fuzzy c-means, algorithms of Gustafson-Kessel, Yager-Filev, Klawonn-Hoeppner, etc) (Bezdek et al., 2005; Sato-Ilic & Jain, 2006), occupy a significant place as they make it possible to solve a data processing problem in the absence of a priori knowledge of it.

While there are many artificial neural networks that can be successfully used in image processing tasks, the most prominent of them are networks of a new, the third generation, commonly known as spiking neural networks (Maass & Bishop, 1998; Gerstner & Kistler, 2002). On the one hand, spiking neural networks are biologically more plausible than neural networks of the previous generations that is of fundamental importance for computational intelligence from theoretical point of view. On the other hand, networks of spiking neurons appeared to be computationally more powerful than conventional neural networks (Maass, 1997b). In addition, complex data processing via artificial neural networks of the second generation is time consuming due to multi-epoch learning; instead, spiking neural networks can perform the same processing tasks much faster as they require a few learning epochs only (Bohte et al., 2002; Berredo, 2005; Meftah et al., 2008; Lindblad & Kinser, 2005). All these facts are causing considerable interest in networks of spiking neurons as a powerful computational intelligence tool for image processing

Although spiking neural networks are becoming a popular computational intelligence tool for various technical problems solving, their architecture and functioning are treated in terms of neurophysiology rather than in terms of any technical sciences apparatus in the most research works on engineering subjects. Yet none technically plausible description of spiking neurons functioning has been provided.

In contrast to artificial neural networks, fuzzy logic systems are capable of performing accurate and efficient data processing under a priori and current uncertainty, particularly if classes to be separated overlap one another. Integrating artificial neural networks and fuzzy systems together allows of combining capabilities of both in a synergetic way (Jang et al., 1997), thus producing hybrid intelligent systems that achieve high performance and reliability in real life problems solving, particularly in image processing. Obviously,

Source: Image Processing, Book edited by: Yung-Sheng Chen,
ISBN 978-953-307-026-1, pp. 572, December 2009, INTECH, Croatia, downloaded from SCIYO.COM

designing hybrid intelligent systems based on the new generation of artificial neural networks attracts both practical and theoretical interest.
In the present chapter of the book, analog-digital self-learning fuzzy spiking neural network that belongs to a new type of computational intelligence hybrid systems combining spiking neurons computational capabilities and fuzzy systems tolerance for uncertainty is proposed. It is demonstrated that both fuzzy probabilistic and fuzzy possibilistic approaches can be implemented on spiking neural network basis. A spiking neural network is treated in terms of well-known and widely used apparatus of classical automatic control theory based on the Laplace transform. It is shown that a spiking neuron synapse is nothing other than a second-order critically damped response unit, and a spiking neuron soma is a kind of threshold detection system. An improved unsupervised learning algorithm for the proposed neural network based on 'Winner-Takes-More' rule is introduced. Capabilities of the neural network in solving image processing problems are investigated. From theoretical point of view, the proposed neural network is another step toward evolution of artificial neural networks theory as a part of computational intelligence paradigm.

2. Formal models of spiking neurons

Biological neuron constructive features that are significant for the discussion that follows are sketched on Fig. 1 (Dayan & Abbott, 2001; Scott, 2002). As illustrated, neuron includes synapses, dendritic tree, soma and axon and its terminals. Synapse connects axonal terminals of a neuron with dendrites of another neuron. Soma processes incoming information and transmits it through axon and axonal terminals to synapses of the subsequent neurons. Neurons communicate one another by nerve pulses (action potentials, spikes).

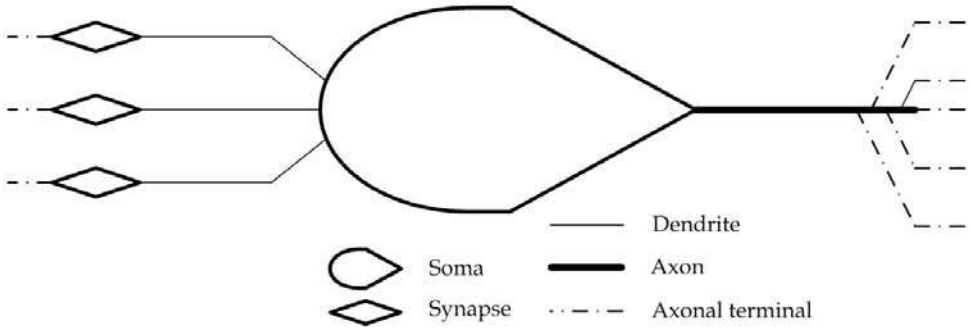

Fig. 1. Biological neuron

Neuron behaviour can be briefly described in the following way (Fig. 2). Spike arrived to synapse from presynaptic neuron generates postsynaptic potential (either excitatory or inhibitory – depending on synapse type). Postsynaptic potential reaches neuron soma through a dendrite and either increases membrane potential, or decreases it. Neuron soma accumulates all postsynaptic potentials incoming from different synapses. When membrane potential exceeds firing threshold, neuron fires and emits outgoing spike that moves through axon to postsynaptic neurons. Once neuron has fired, its soma produces spike after-potential, namely, the membrane potential drops steeply below the rest potential and then it ascends gradually to the rest potential back. Period when membrane potential is below the

rest potential is called refractory period. Within this period, appearance of another spike is unlikely. If the firing threshold is not reached after arrival of a postsynaptic potential, membrane potential gradually descends to rest potential until another postsynaptic potential incomes.

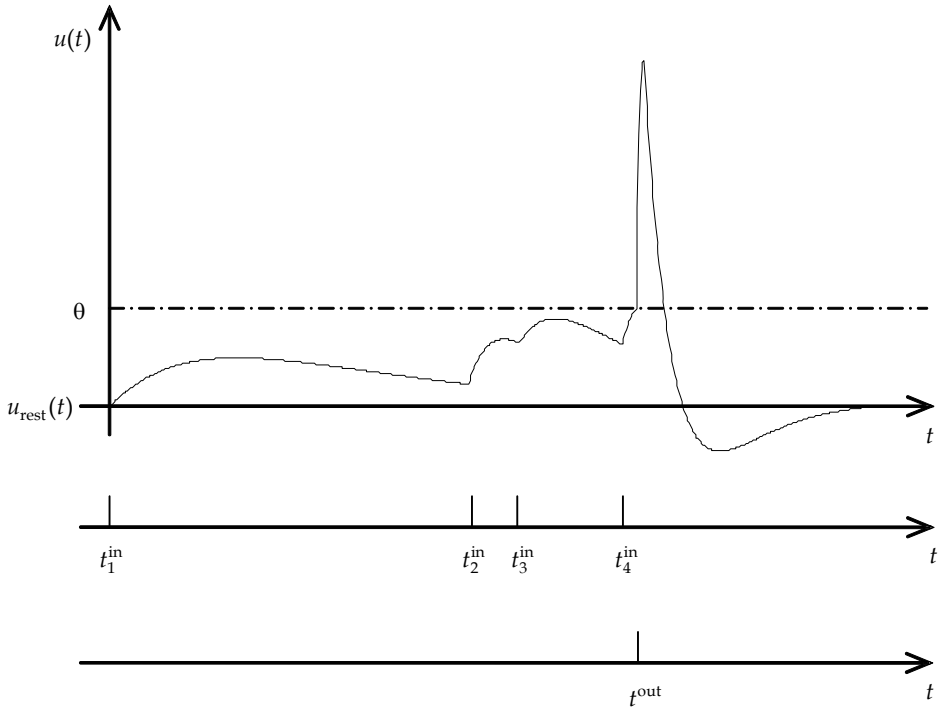

Fig. 2. Biological neuron behaviour: a) Dynamics of membrane potential $u(t)$ (θ is the firing threshold, $u_{\text{rest}}(t)$ is the rest potential); b) Incoming spikes; c) Outgoing spike

Traits of any artificial neural networks generation depend upon the formal model of biological neuron that is considered within scope of that generation. Any formal model treats biological neuron on a certain level of abstraction. It takes into account some details of biological neuron behaviour and features, but disregards other ones. On the one hand, prescribing complexity level of formal model sets computational and performance properties of artificial neural networks originated by that model. On the other hand, chosen level of abstraction of formal model defines how realistic artificial neural networks are.

Both the first and the second generations of artificial neural networks (Maass, 1997b) rest on the rate model that neglects temporal properties of biological neuron (Maass & Bishop, 1998). One of the essential elements for both generations is a neuron activation function. The rate model based on the threshold activation function (McCulloch & Pitts, 1943) gave birth to the first generation of artificial neural networks. Though such networks were capable of performing some elementary logic functions, their computational capabilities were very limited (Minsky & Papert, 1969). Replacing the threshold activation function with

continuous one resulted in appearance of the second generation, that turned out to be significantly powerful than networks of the previous generation (Cybenko, 1989; Hornik et al., 1989). Nevertheless, neurons of the second generation are even far from real biological neurons than the first generation neurons since they ignore soma firing mechanism totally (Gerstner & Kistler, 2002). This gap is avoided in threshold-fire models (Maass & Bishop, 1998). One of such models, namely, the leaky integrate-and-fire model (Maass & Bishop, 1998; Gerstner & Kistler, 2002), is the basis for the third generation of artificial neural networks.

The leaky integrate-and-fire model is the one of the simplest and well-known formal models of a biological neuron that are used in different areas of neuroscience. It captures neuron behaviour described above except the neuron refractoriness. The model considers that on firing, membrane potential drops to the rest potential, not below it.

The spike-response model (Maass & Bishop, 1998; Gerstner & Kistler, 2002), another threshold-fire model, captures biological neuron behaviour more accurately. Besides postsynaptic potentials accumulating and spike firing, it models the neuron refractoriness also. This model will be used in the subsequent discussion.

It makes sense to mention here that computational neuroscience and computational intelligence sometimes understand spiking neurons in a different way. Spiking neuron in computational neuroscience is any model of biological neuron that transmits and processes information by spike trains. Within scope of computational intelligence, spiking neuron is a leaky integrate-and-fire model usually. This results from the fact that self-learning properties of spiking neurons are caused by capability of any threshold-fire model to detect coincidences in input signal. Since the leaky integrate-and-fire model is the simplest one among the threshold-fire models, there is no sense to use any complicated ones. Obviously, if any more complicated model reveals some particular properties that are useful for solving technical problems, the concept of spiking neurons will be extended in computational intelligence.

3. Self-learning spiking neural network

3.1 Introduction

Ability of spiking neurons to respond to incoming signal selectively was originally discovered by J. Hopfield in 1995 (Hopfield, 1995). He found that spiking neuron soma behaviour was similar to a radial basis function: the neuron fired as earlier as higher degree of coincidence of incoming spikes was; if the degree was sufficiently low, the neuron did not fire at all. And spiking neuron synapses appeared to be acting as a spike pattern storing unit: one was able to get a spiking neuron to fire to a certain spike pattern by adjusting synaptic time delays the way that they evened out (in temporal sense) incoming signal and made it to reach the neuron soma simultaneously. Spike pattern encoded in synaptic time delays of a neuron was called a center of spiking neuron in the following. Here it is worth to note that synchronization phenomena is of primary importance in nature (Pikovsky et al., 2001), particularly in the brain functioning (Malsburg, 1994).

The discovered capabilities of spiking neurons provided the basis for constructing self-learning networks of spiking neurons. Original architecture of self-learning spiking neural network and its learning algorithm, namely, a temporal Hebbian rule were introduced in (Natschlaeger & Ruf, 1998). The proposed self-learning network was able to separate

clusters as long as their number was not greater than dimensionality of input signal. If clusters number exceeded number of input signal dimensions, spiking neural network performance decreased. This drawback was overcome by using population coding of incoming signal based on pools of receptive neurons in the first hidden layer of the network (Bohte et al., 2002). Such spiking neural network was shown to be considerably powerful and significantly fast in solving real life problems. Henceforward we will use this network as a basis for its further improvements and hybrid architectures designing.

3.2 Architecture

Self-learning spiking neural network architecture is shown on Fig. 3. As illustrated, it is heterogeneous two-layered feed-forward neural network with lateral connections in the second hidden layer.

The first hidden layer consists of pools of receptive neurons and performs transformation of input signal. It encodes an $(n\times 1)$-dimensional input sampled pattern $x(k)$ (here n is the dimensionality of input space, $k=\overline{1,N}$ is a patter number, N is number of patterns in incoming set) into $(hn\times 1)$-dimensional vector of input spikes $t^{[0]}(x(k))$ (here h is the number of receptive neurons in a pool) where each spike is defined by its firing time.

Spiking neurons form the second hidden layer of the network. They are connected with receptive neurons with multiple synapses where incoming vector of spikes transforms into postsynaptic potentials. Number of spiking neurons in the second hidden layer is set to be equal to the number of clusters to be detected. Each spiking neuron corresponds to a certain cluster. The neuron that has fired to the input pattern defines cluster that the pattern belongs to. Thus, the second hidden layer takes $(hn\times 1)$-dimensional vector of input spikes $t^{[0]}(x(k))$ and outputs $(m\times 1)$-dimensional vector of outgoing spikes $t^{[1]}(x(k))$ that defines the membership of input pattern $x(k)$.

This is the basic architecture and behaviour of self-learning spiking neural network. The detailed architecture is stated below.

3.3 Population coding and receptive neurons

The first hidden layer is constructed to perform population coding of input signal. It acts in such a manner that each dimensional component $x_i(k)$, $i=\overline{1,n}$, of input signal $x(k)$ is processed by a pool of h receptive neurons RN_{li}, $l=\overline{1,h}$. Obviously, there can be different number of receptive neurons h_i in a pool for each dimensional component in the general case. For the sake of simplicity, we will consider here that the number of neurons is equal for all pools.

As a rule, activation functions of receptive neurons within a pool are bell-shaped (Gaussians usually), shifted, overlapped, of different width, and have dead zone. Generally firing time of a spike emitted by a receptive neuron RN_{li} upon incoming signal $x_i(k)$ lies in a certain interval $\{-1\}\cup[0,t^{[0]}_{\max}]$ referred to as coding interval and is described by the following expression:

$$t^{[0]}_{li}(x_i(k))=\begin{cases}\left\lfloor t^{[0]}_{\max}\left(1-\psi\left(\left|x_i(k)-c^{[0]}_{li}\right|,\sigma_{li}\right)\right)\right\rfloor, & \psi\left(\left|x_i(k)-c^{[0]}_{li}\right|,\sigma_{li}\right)\geq\theta_{\text{r.n.}},\\ -1, & \psi\left(\left|x_i(k)-c^{[0]}_{li}\right|,\sigma_{li}\right)<\theta_{\text{r.n.}},\end{cases} \tag{1}$$

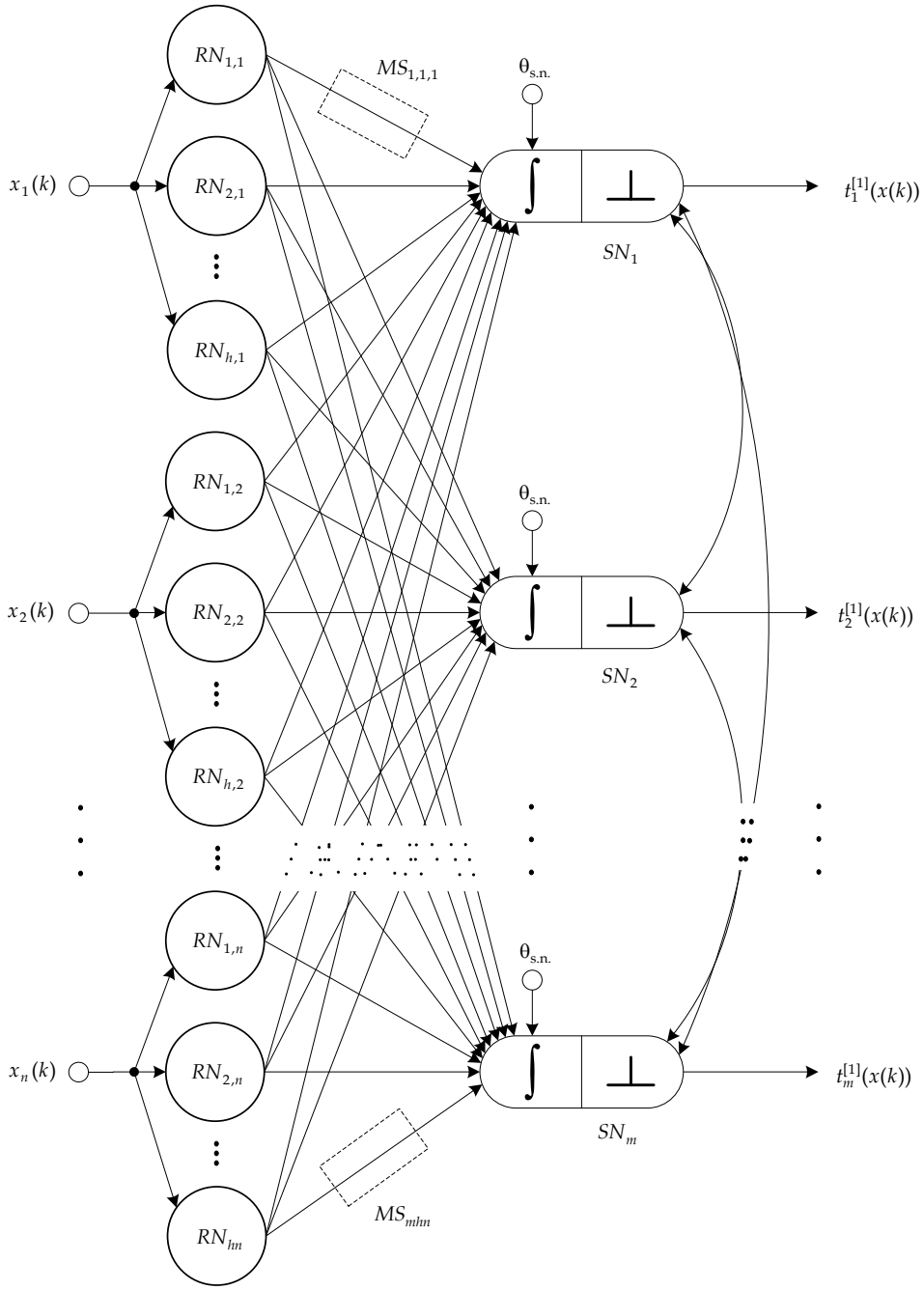

Fig. 3. Self-learning spiking neural network (spiking neurons are depicted the way to stress they act in integrate-and-fire manner)

where $\lfloor \bullet \rfloor$ is the floor function, $\psi(\bullet,\bullet)$, $c_{li}^{[0]}$, σ_{li}, and $\theta_{\text{r.n.}}$ are the receptive neuron's activation function, center, width and dead zone, respectively (r.n. in the last parameter means 'receptive neuron'), -1 indicates that the neuron does not fire. An example of population coding is depicted on Fig. 4. It is easily seen that the closer $x_i(k)$ is to the center $c_{li}^{[0]}$ of receptive neuron RN_{li}, the earlier the neuron emits spike $t_{li}^{[0]}(x_i(k))$.

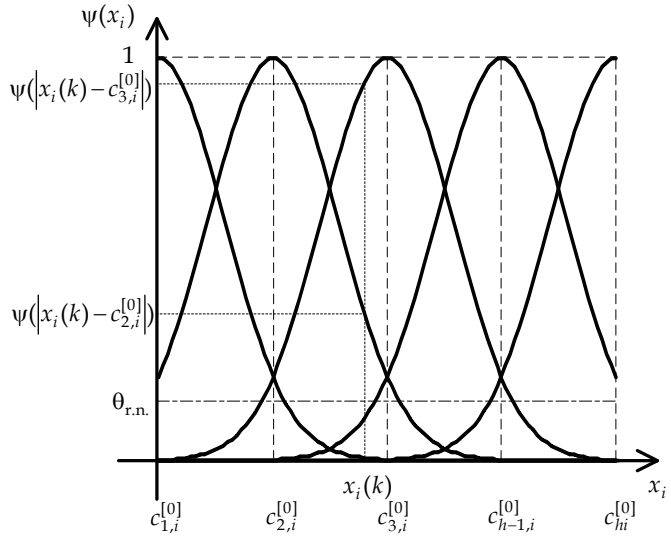

Fig. 4. An example of population coding. Incoming signal $x_i(k)$ fires receptive neurons $RN_{2,i}$ and $RN_{3,i}$. It is considered here that width of activation function of all receptive neurons within the pool is the same

In this work, we used Gaussian as activation function of receptive neurons:

$$\psi\left(\left|x_i(k)-c_{li}^{[0]}\right|,\sigma_{li}\right)=\exp\left(-\frac{\left(x_i(k)-c_{li}^{[0]}\right)^2}{2\sigma_{li}^2}\right). \tag{2}$$

There can be several ways to set widths and centers of receptive neurons within a pool. As a rule, activation functions can be of two types – either 'narrow' or 'wide'. Centers of each width type of activation function are calculated in different ways but in either case they cover date range uniformly. More details can be found in (Bohte et al., 2002).

3.4 Spiking neurons layer

Spiking neuron is considered to be formed of two constituents, they are: synapse and soma. As it was mentioned above, synapses between receptive neurons and spiking neurons are multiple structures. As shown on Fig. 5, a multiple synapse MS_{jli} consists of a set of q subsynapses with different time delays d^p, $d^p - d^{p-1} > 0$, $d^q - d^1 > t_{\max}^{[0]}$, and adjustable weights w_{jli}^p (here $p = \overline{1,q}$). It should be noted that number of subsynapses within a

multiple synapse are fixed for the whole network. Having a spike $t_{li}^{[0]}(x_i(k))$ from the li-th receptive neuron, the p-th subsynapse of the j-th spiking neuron produces delayed weighted postsynaptic potential

$$u_{jli}^{p}(t) = w_{jli}^{p}\varepsilon_{jli}^{p}(t) = w_{jli}^{p}\varepsilon\left(t - \left(t_{li}^{[0]}(x_i(k)) + d^{p}\right)\right), \tag{3}$$

where $\varepsilon(\bullet)$ is a spike-response function usually described by the expression (Natschlaeger & Ruf, 1998)

$$\varepsilon\left(t - \left(t_{li}^{[0]}(x_i(k)) + d^{p}\right)\right) = \frac{t - \left(t_{li}^{[0]}(x_i(k)) + d^{p}\right)}{\tau_{\mathrm{PSP}}} \exp\left(1 - \frac{t - \left(t_{li}^{[0]}(x_i(k)) + d^{p}\right)}{\tau_{\mathrm{PSP}}}\right) \times \\ \times H\left(t - \left(t_{li}^{[0]}(x_i(k)) + d^{p}\right)\right), \tag{4}$$

τ_{PSP} – membrane potential decay time constant whose value can be obtained empirically (PSP means 'postsynaptic potential'), $H(\bullet)$ – the Heaviside step function. Output of the multiple synapse MS_{jli} forms total postsynaptic potential

$$u_{jli}(t) = \sum_{p=1}^{q} u_{jli}^{p}(t) . \tag{5}$$

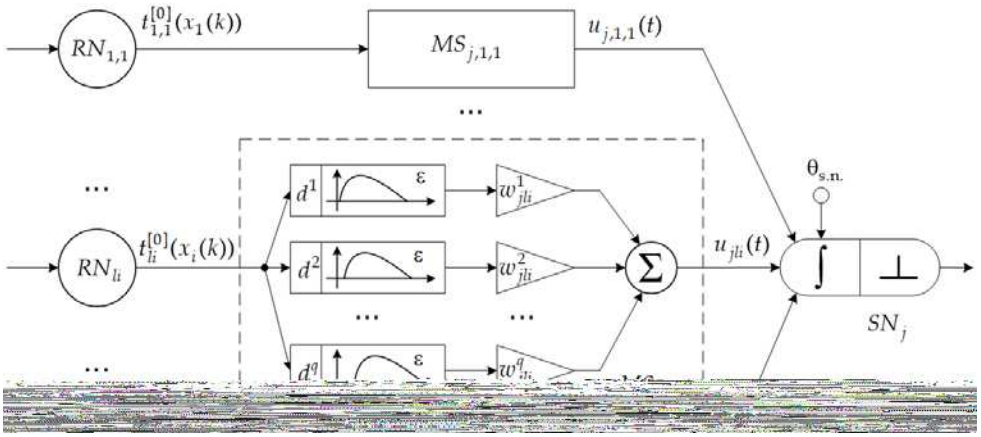

Fig. 5. Multiple synapse

Each incoming total postsynaptic potential contributes to membrane potential of spiking neuron SN_j as follows:

$$u_j(t) = \sum_{i=1}^{n}\sum_{l=1}^{h} u_{jli}(t) . \tag{6}$$

Spiking neuron SN_j generates at most one outgoing spike $t_j^{[1]}(x(k))$ during a simulation interval (the presentation of an input pattern $x(k)$), and fires at the instant the membrane potential reaches firing threshold $\theta_{\text{s.n.}}$ (s.n. means here 'spiking neuron'). After neuron has fired, the membrane potential is reset to the rest value u_{rest} (0 usually) until the next input pattern is presented.
It is easily seen that the descried behaviour of spiking neuron corresponds to the one of the leaky integrate-and-fire model.
Spiking neurons are linked with lateral inhibitory connections that disable all other neurons to fire after the first one has fired. Thus, any input pattern makes only one spiking neuron to fire that is only one component of the vector of outgoing spikes has non-negative value. There can be cases when a few spiking neurons fire simultaneously for an input pattern. Such cases are rare enough, and their appearance depends directly on initial synaptic weights distribution.

3.5 Crisp data clustering

As it was mentioned above, a spiking neuron acts similarly to a radial basis function, and its response depends on degree of coincidence of the input. There was considered a spiking neuron center to describe such neuron behaviour in a convenient way (Natschlaeger & Ruf, 1998). In the general case, it is considered to possess the following property: the closer input pattern is to the neuron's center, the earlier output spike fires. Hence, a spiking neuron firing time reflects the similarity (Natschlaeger & Ruf, 1998) (or distance (Bohte et al., 2002)) between the input pattern and the neuron center. Degree of coincidence is utilized here as a similarity (distance) measure.
Center of spiking neuron is encoded in the synaptic time delays. They produce coincidence output (and in its terns it makes the soma to fire at the earliest possible time) if incoming pattern is similar to the encoded one. Thus, the learned spiking neuron can respond selectively to the input set of patterns. Data clustering in the described neural network rests on this property of spiking neuron. Input pattern fires the neuron whose center is the most similar (the closest) to it, and the fired spike prevents the rest neurons to fire through the lateral inhibitory connections. This way self-learning spiking neural network performs clusters separation if classes to be detected do not overlap one another.
One can readily see that the unsupervised pattern classification procedure of the spiking neurons layer is identical with the one of self-organizing maps (Kohonen, 1995).

4. Spiking neural network learning algorithms

4.1 Winner-takes-all

The purpose of an unsupervised learning algorithm of spiking neural network is to adjust centers of spiking neurons so as to make each of them to correspond to centroid of a certain data cluster. Such learning algorithm was introduced on the basis of two learning rules, namely, 'Winner-Takes-All' rule and temporal Hebbain rule (Natschlaeger & Ruf, 1998; Gerstner et al., 1996). The first one defines which neuron should be updated, and the second one defines how it should be updated. The algorithm updates neuron centers through synaptic weights adjusting, whereas synaptic time delays always remain constant. The concept here is that significance of the given time delay can be changed by varying corresponding synaptic weight.

Each learning epoch consists of two phases. Competition, the first phase, defines a neuron-winner. Being laterally linked with inhibitory connections, spiking neurons compete to respond to the pattern. The one wins (and fires) whose center is the closest to the pattern. Following competition phase, weights adjusting takes place. The learning algorithm adjusts synaptic weights of the neuron-winner to move it closer to the input pattern. It strengthens weights of those subsynapses which contributed to the neuron-winner's firing (i.e. the subsynapses produced delayed spikes right before the neuron firing) and weakens ones which did not contribute (i.e. the delayed spikes appeared right after the neurons firing or long before it). Generally, the learning algorithm can be expressed as

$$w_{jli}^{p}(K+1)=\begin{cases} w_{jli}^{p}(K)+\eta_{w}(K)L(\Delta t_{jli}^{p}), & j=\tilde{j}, \\ w_{jli}^{p}(K), & j\neq\tilde{j}, \end{cases} \tag{7}$$

where K is the current epoch number, $\eta_{w}(\bullet)>0$ is the learning rate (while it is constant in (Natschlaeger & Ruf, 1998), it can depend on epoch number in the general case; w means 'weights'), $L(\bullet)$ is the learning function (Gerstner et al, 1996), $\tilde{j}$ is the number of neuron that has won on the current epoch, Δt_{jli}^{p} is the time delay between delayed spike $t_{li}^{[0]}(x_i(k))+d^{p}$ produced by the p-th subsynapse of the li-th synapse and spiking neuron firing time $t_{j}^{[1]}(x(k))$:

$$\Delta t_{jli}^{p}=t_{li}^{[0]}(x_i(k))+d^{p}-t_{j}^{[1]}(x(k)). \tag{8}$$

As a rule, the learning function has the following form (Berredo, 2005):

$$L(\Delta t_{jli}^{p})=(1+\beta)\exp\left(-\frac{\left(\Delta t_{jli}^{p}-\alpha\right)^{2}}{2(\kappa-1)}\right)-\beta, \tag{9}$$

$$\kappa=1-\frac{\nu^{2}}{2\ln\left(\frac{\beta}{1+\beta}\right)}, \tag{10}$$

where $\alpha<0$, $\beta>0$, ν are the shape parameters of the learning function $L(\bullet)$ that can be obtained empirically (Berredo, 2005; Natschlaeger & Ruf, 1998). The learning function and its shape parameters are depicted on Fig. 6. The effect of the shape parameters on results of information processing performed by spiking neural network can be found in (Meftah et al., 2008).

Upon the learning stage, center of a spiking neuron represents centroid of a certain data cluster, and spiking neural network can successfully perform unsupervised classification of the input set.

4.2 Winner-takes-more

The learning algorithm (7) updates only neuron-winner on each epoch and disregards other neurons. It seems more natural to update not only spiking neuron-winner, but also its neighbours (Bodyanskiy & Dolotov, 2009). This approach is known as 'Winner-Takes-More'

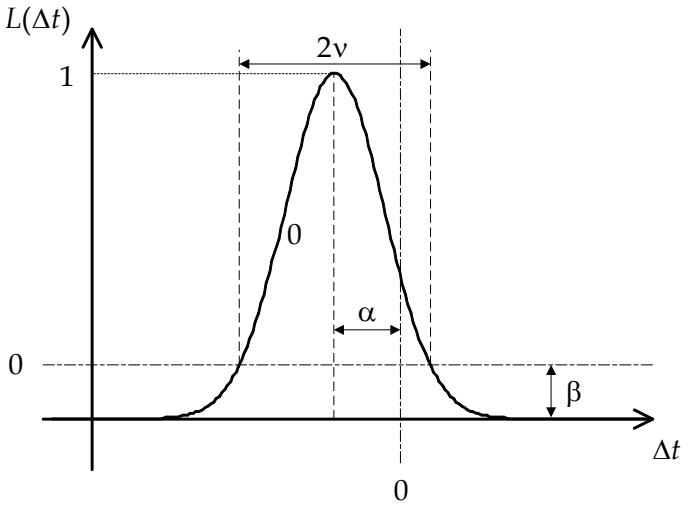

Fig. 6. Learning function $L(\bullet)$

rule. It implies that there is a cooperation phase before weights adjustment. Neuron-winner determines a local region of topological neighbourhood on each learning epoch. Within this region, the neuron-winner fires along with its neighbours, and the closer a neighbour is to the winner, the more significantly its weights are changed. The topological region is represented by the neighbourhood function $\varphi(\left|\Delta t_{\tilde{j}j}\right|)$ that depends on difference $\left|\Delta t_{\tilde{j}j}\right|$ between the neuron-winner firing time $t_{\tilde{j}}^{[1]}(x(k))$ and the neighbour firing time $t_j^{[1]}(x(k))$ (distance between the neurons in temporal sense) and a parameter that defines effective width of the region. As a rule, $\varphi(\bullet)$ is a kernel function that is symmetric about its maximum at the point where $\Delta t_{\tilde{j}j} = 0$. It reaches unity at that point and monotonically decreases as $\Delta t_{\tilde{j}j}$ tends to infinity. The functions that are the most frequently used as neighbourhood function are Gaussian, paraboloid, Mexican Hat, and many others (Bodyanskiy & Rudenko, 2004).

For self-learning spiking neural network, the learning algorithm based on 'Winner-Takes-More' rule can be expressed in the following form (Bodyanskiy & Dolotov, 2009):

$$w_{jli}^{p}(K+1) = w_{jli}^{p}(K) + \eta_{w}(K)\varphi(\left|\Delta t_{\tilde{j}j}\right|)L(\Delta t_{jli}^{p}), \tag{11}$$

where temporal distance $\Delta t_{\tilde{j}j}$ is

$$\Delta t_{\tilde{j}j} = t_{\tilde{j}}^{[1]}(x(k)) - t_j^{[1]}(x(k)). \tag{12}$$

Obviously, expression (11) is a generalization of (7).

Analysis of competitive unsupervised learning convergence showed that width parameter of the neighbourhood function should decrease during synaptic weights adjustment (Cottrell & Fort, 1986). For Gaussian neighbourhood function

$$\varphi\left(\left|\Delta t_{\tilde{j}\tilde{j}}\right|, K\right) = \exp\left(-\frac{\left(\Delta t_{\tilde{j}\tilde{j}}\right)^2}{2\rho^2(K)}\right) \tag{13}$$

width parameter ρ can be adjusted as follows (Ritter & Schulten, 1986):

$$\rho(K) = \rho(0)\exp\left(\frac{K}{\gamma}\right), \tag{14}$$

where $\gamma > 0$ is a scalar that determines rate of neuron-winner effect on its neighbours. Noteworthily that exponential decreasing of width parameter can be achieved by applying the simpler expression instead of (14) (Bodyanskiy & Rudenko, 2004):

$$\rho(K) = \gamma\rho(K-1), 0 < \gamma < 1. \tag{15}$$

Learning algorithm (11) requires modification of self-learning spiking neural architecture. Lateral inhibitory connections in the second hidden layer should be replaced with excitatory ones during the network learning stage in order to implement 'Winner-Takes-More' rule.
In the following sections, it will be shown that the learning algorithm based on 'Winner-Takes-More' rule is more natural than the one based on 'Winner-Takes-All' rule to learn fuzzy spiking neural network.

5. Spiking neural network as an analog-digital system

5.1 Introduction

Hardware implementations of spiking neural network demonstrated fast processing ability that made it possible to apply such systems in real-life applications where processing speed was a rather critical parameter (Maass, 1997a; Maass & Bishop, 1998; Schoenauer et al., 2000; Kraft et al., 2006). From theoretical point of view, the current research works on spiking neurons hardware implementation subject are very particular, they lack for a technically plausible description on a general ground. In this section, we consider a spiking neuron as a processing system of classical automatic control theory (Feldbaum & Butkovskiy, 1971; Dorf & Bishop, 1995; Phillips & Harbor, 2000; Goodwin et al., 2001). Spiking neuron functioning is described in terms of the Laplace transform. Having such a general description of a spiking neuron, one can derive various hardware implementations of self-learning spiking neural network for solving specific technical problems, among them realistic complex image processing.
Within a scope of automatic control theory, a spike $t(x(k))$ can be represented by the Dirac delta function $\delta(t - t(x(k)))$. Its Laplace transform is

$$L\{\delta(t - t(x(k)))\} = e^{-t(x(k))s}, \tag{16}$$

where s is the Laplace operator. Spiking neuron takes spikes on its input, performs spike–membrane potential–spike transformation, and produces spikes on its output. Obviously, it is a kind of analog-digital system that processes information in continuous-time form and transmits it in pulse-position form. This is the basic concept for designing analog-digital architecture of self-learning spiking neural network. Overall network architecture is depicted on Fig. 7 and is explained in details in the following subsections.

$x_1(k)$
$x_i(k)$
$x_n(k)$
$RN_{1,1}$
$RN_{l,1}$
$RN_{h,1}$
$RN_{1,i}$
RN_{li}
RN_{hi}
$RN_{1,n}$
RN_{ln}
RN_{hn}
$\delta\left(t-t_{1,1}^{[0]}(x_1(k))\right)$
$\delta\left(t-t_{li}^{[0]}(x_i(k))\right)$
$\delta\left(t-t_{hn}^{[0]}(x_n(k))\right)$
$e^{-t_{1,1}^{[0]}(x_1(k))s}$
$e^{-t_{li}^{[0]}(x_i(k))s}$
$e^{-t_{hn}^{[0]}(x_n(k))s}$
t
$t_{\max}^{[0]}$
$MS_{j,1,1}$
MS_{jhn}
$u_{j,1,1}(t)$
$u_{j,1,1}(s)$
$u_{jhn}(t)$
$u_{jhn}(s)$
$e^{-d^1 s}$
$e^{-d^p s}$
$e^{-d^q s}$
$\frac{e\tau_{\mathrm{PSP}}}{(\tau_{\mathrm{PSP}}s+1)^2}$
w_{jli}^1
w_{jli}^p
w_{jli}^q
$u_{jli}^1(t)$
$u_{jli}^1(s)$
$u_{jli}^p(t)$
$u_{jli}^p(s)$
$u_{jli}^q(t)$
$u_{jli}^q(s)$
Σ
MS_{jli}
$u_{jli}(t)$
$u_{jli}(s)$
$\theta_{\mathrm{s.n.}}$
$u_j(t)$
$u_j(s)$
s
w_{SAP}
$\frac{1}{(\tau_{\mathrm{SAP}}s+1)}$
SN_j

5.2 Synapse as a second-order critically damped response unit

Multiple synapse MS_{jli} of a spiking neuron SN_j transforms incoming pulse-position signal $\delta\left(t-t_{li}^{[0]}(x_i(k))\right)$ to continuous-time signal $u_{jli}(t)$. Spike-response function (4), the basis of such transformation, has form that is similar to the one of impulse response of second-order damped response unit. Transfer function of a second-order damped response unit with unit gain factor is

$$\widetilde{G}(s)=\frac{1}{(\tau_1 s+1)(\tau_2 s+1)}=\frac{1}{\tau_4^2 s^2+\tau_3 s+1}, \tag{17}$$

where $\tau_{1,2}=\frac{\tau_3}{2}\pm\sqrt{\frac{\tau_3^2}{4}-\tau_4^2}$, $\tau_1\geq\tau_2$, $\tau_3\geq 2\tau_4$, and its impulse response is

$$\widetilde{\varepsilon}(t)=\frac{1}{\tau_1-\tau_2}\left(e^{-\frac{t}{\tau_1}}-e^{-\frac{t}{\tau_2}}\right). \tag{18}$$

Putting $\tau_1=\tau_2=\tau_{\mathrm{PSP}}$ (that corresponds to a second-order critically damped response unit) and applying l'Hôpital's rule, one can obtain

$$\widetilde{\varepsilon}(t)=\frac{t}{\tau_{\mathrm{PSP}}^2}e^{-\frac{t}{\tau_{\mathrm{PSP}}}}. \tag{19}$$

Comparing spike-response function (4) with the impulse response (19) leads us to the following relationship:

$$\varepsilon(t)=e\tau_{\mathrm{PSP}}\widetilde{\varepsilon}(t). \tag{20}$$

Thus, transfer function of the second-order critically damped response unit whose impulse response corresponds to a spike-response function is

$$G_{\mathrm{SRF}}(s)=\frac{e\tau_{\mathrm{PSP}}}{(\tau_{\mathrm{PSP}}s+1)^2}, \tag{21}$$

where SRF means 'spike-response function'.

Now, we can design multiple synapse in terms of the Laplace transform (Bodyanskiy et al., 2009). As illustrated on Fig. 7, multiple synapse MS_{jli} is a dynamic system that consists of a set of subsynapses that are connected in parallel. Each subsynapse is formed by a group of time delay, second-order critically damped response unit, and gain. As a response to incoming spike $\delta\left(t-t_{li}^{[0]}(x_i(k))\right)$, the subsynapse produces delayed weighted postsynaptic potential $u_{jli}^p(s)$, and the multiple synapse produces total postsynaptic potential $u_{jli}(s)$ that arrives to spiking neuron soma.

Taking into account (21), transfer function of the p-th subsynapse of MS_{jli} takes the following form:

$$U_{jli}^p(s)=w_{jli}^p e^{-d^p s}G_{\mathrm{SRF}}(s)=\frac{\tau_{\mathrm{PSP}}w_{jli}^p e^{1-d^p s}}{(\tau_{\mathrm{PSP}}s+1)^2}, \tag{22}$$

and its response to a spike $\delta\left(t-t_{li}^{[0]}(x_i(k))\right)$ is

$$u_{jli}^{p}(s)=e^{-t_{li}^{[0]}(x_i(k))s}U_{jli}^{p}(s)=\frac{\tau_{\text{PSP}}w_{jli}^{p}e^{1-\left(t_{li}^{[0]}(x_i(k))+d^{p}\right)s}}{\left(\tau_{\text{PSP}}s+1\right)^{2}}. \tag{23}$$

So finally, considering transfer function of multiple synapse MS_{jli}

$$U_{jli}(s)=\sum_{p=1}^{q}U_{jli}^{p}(s)=\sum_{p=1}^{q}\frac{\tau_{\text{PSP}}w_{jli}^{p}e^{1-d^{p}s}}{\left(\tau_{\text{PSP}}s+1\right)^{2}}, \tag{24}$$

the Laplace transform of the multiple synapse output can be expressed in the following form:

$$u_{jli}(s)=e^{-t_{li}^{[0]}(x_i(k))s}U_{jli}(s)=\sum_{p=1}^{q}\frac{\tau_{\text{PSP}}w_{jli}^{p}e^{1-\left(t_{li}^{[0]}(x_i(k))+d^{p}\right)s}}{\left(\tau_{\text{PSP}}s+1\right)^{2}}. \tag{25}$$

Here it is worth to note that since it is impossible to use δ-function in practice (Phillips & Harbor, 2000), it is convenient to model it with impulse of a triangular form (Feldbaum & Butkovskiy, 1971) as shown on Fig. 8. Such impulse is similar to δ-function under the following condition

$$\lim_{\Delta\to 0}a(t,\Delta)=\delta(t). \tag{26}$$

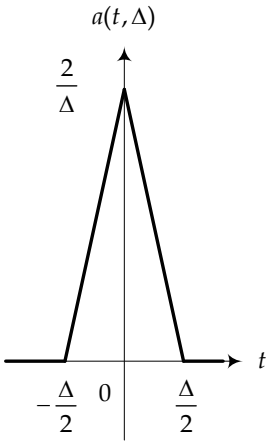

Fig. 8. Triangular impulse $a(t,\Delta)$

In this case, numerator of (21) should be revised the way to take into account finite peak of $a(t,\Delta)$ (in contrast to the one of the Dirac delta function).

5.3 Soma as a threshold detection unit

Spiking neuron soma performs transformation that is opposite to one of the synapse. It takes continuous-time signals $u_{jli}(t)$ and produces pulse-position signal $\delta\left(t-t_{j}^{[1]}(x(k))\right)$. In doing

so, soma responds each time membrane potential reaches a certain threshold value. In other words, spiking neuron soma acts as a threshold detection system and consequently it can be designed on the base of relay control systems concept (Tsypkin, 1984). In (Bodyanskiy et al., 2009), mechanisms of threshold detection behaviour and firing process were described. Here we extend this approach to capture refractoriness of spiking neuron.
Threshold detection behaviour of a neuron soma can be modelled by an element relay with dead zone $\theta_{s.n.}$ that is defined by the nonlinear function

$$\Phi_{\text{relay}}\left(u_j(t), \theta_{s.n.}\right) = \frac{\text{sign}\left(u_j(t) - \theta_{s.n.}\right) + 1}{2}, \tag{27}$$

where sign (•)is the signum function. Soma firing can be described by a derivative unit that is connected with the element relay in series and produces a spike each time the relay switches. In order to avoid a negative spike that appears as a response to the relay resetting, a conventional diode is added next to the derivative unit. The diode is defined by the following function:

$$\Phi_{\text{diode}}\left(\delta\left(t - t_{\text{relay}}^{[1]}\right)\right) = \delta\left(t - t_{\text{relay}}^{[1]}\right) H\left(\delta\left(t - t_{\text{relay}}^{[1]}\right)\right), \tag{28}$$

where $t_{\text{relay}}^{[1]}$ is a spike produced by the derivative unit upon the relay switching.
Now we can define the Laplace transform of an outgoing spike $t_j^{[1]}(x(k))$, namely,

$$L\left\{\delta\left(t - t_j^{[1]}(x(k))\right)\right\} = e^{-t_j^{[1]}(x(k))s} = L\left\{\Phi_{\text{diode}}\left(sL\left\{\Phi_{\text{relay}}\left(u_j(t), \theta_{s.n.}\right)\right\}\right)\right\}. \tag{29}$$

As it was mentioned above, the leaky integrate-and-fire model disregards the neuron refractoriness. Anyway, the refractory period is implemented in the layer of spiking neurons indirectly. The point is that a spiking neuron cannot produce another spike after firing and until the end of the simulation interval since the input pattern is provided only once within the interval. In the analog-digital architecture of spiking neuron, the refractoriness can be modelled by a feedback circuit. As shown on Fig. 7, it is a group of a time delay, a second-order critically damped response unit, and a gain that are connected in series. The time delay defines duration of a spike generation period d_{spike} (usually, $d_{\text{spike}} \to 0$). The second-order critically damped response unit defines a spike after-potential. Generally, spike after-potential can be represented by a second-order damped response unit, but for the sake of simplicity, we use critically damped response unit as it can be defined by one parameter only, namely, τ_{SAP} (SAP means here 'spike after-potential'). This parameter controls duration of the refractory period. Finally, the gain unit sets amplitude of the spike after-potential w_{SAP}. Obviously, w_{SAP} should be much greater than any synaptic weight.
Thus, transfer function of the feedback circuit is

$$G_{\text{F.B.}}(s) = \frac{w_{\text{SAP}} e^{-d_{\text{spike}} s}}{\left(\tau_{\text{SAP}} s + 1\right)^2}, \tag{30}$$

where F.B. means 'feedback circuit', and transfer function of the soma is

$$G_{\text{soma}}(s) = \frac{G_{\text{F.F.}}}{1 + G_{\text{F.F.}} G_{\text{F.B.}}}, \tag{31}$$

where $G_{F.F.}$ is defined by (29) (F.F. means 'feed-forward circuit').
It is easily seen that the functioning of spiking neuron analog-digital architecture introduced above is similar to the spike-response model.

6. Self-learning hybrid systems based on spiking neural network

6.1 Fuzzy receptive neurons

A common peculiarity of artificial neural networks is that they store dependence of system model outputs on its inputs in the form of 'black box'. Instead, data processing methods based on fuzzy logic allow of system model designing and storing in analytical form that can be substantially interpreted in a relatively simple way. This fact arises interest in designing of hybrid systems that can combine spiking neural networks computational capabilities with capability of fuzzy logic methods to conveniently describe input-output relationships of the system being modelled. The present section shows how receptive neuron layers, a part of spiking neural network, can be 'fuzzified'.
One can readily see that the layer of receptive neuron pools is identical to a fuzzification layer of neuro-fuzzy systems like Takagi-Sugeno-Kang networks, ANFIS, etc. (Jang et al., 1997). Considering activation function $\psi_{li}(x_i(k))$ as a membership function, the receptive neurons layer can be treated as the one that transforms input data set to a fuzzy set that is defined by values of activation-membership function $\psi_{li}(x_i(k))$ and is expressed over time domain in form of firing times $t_{li}^{[0]}(x_i(k))$ (Bodyanskiy et al., 2008a). In fact, each pool of receptive neurons performs zero order Takagi-Sugeno fuzzy inference (Jang et al., 1997)

$$\text{IF } x_i(k) \text{ IS } X_{li} \text{ THEN OUTPUT IS } t_{li}^{[0]}, \tag{32}$$

where X_{li} is the fuzzy set with membership function $\psi_{li}(x_i(k))$. Thus, one can interpret a receptive neurons pool as a certain linguistic variable and each receptive neuron (more precisely, fuzzy receptive neuron) within the pool – as a linguistic term with membership function $\psi_{li}(x_i(k))$ (Fig. 9). This way, having any a priori knowledge of data structure, it is possible beforehand to adjust activation functions of the first layer neurons to fit them and thus, to get better clustering results.

6.2 Fuzzy clustering

Conventional approach of data clustering implies that each pattern $x(k)$ can belong to one cluster only. It is more natural to consider that a pattern can belong to a several clusters with different membership levels. This case is the subject matter of fuzzy cluster analysis that is heading in several directions. Among them, algorithms based on objective function are the most mathematically rigorous (Bezdek, 1981). Such algorithms solve data processing tasks by optimizing a certain preset cluster quality criterion.
One of the commonly used cluster quality criteria can be stated as follows:

$$E(\mu_j(x(k)), v_j) = \sum_{k=1}^{N}\sum_{j=1}^{m} \mu_j^{\zeta}(x(k)) \left\| x(k) - v_j \right\|_A^2, \tag{33}$$

where $\mu_j(x(k)) \in [0,1]$ is the membership level of the input pattern $x(k)$ to the j-th cluster, v_j is the center of the j-th cluster, $\zeta \geq 0$ is the fuzzifier that determines boundary between

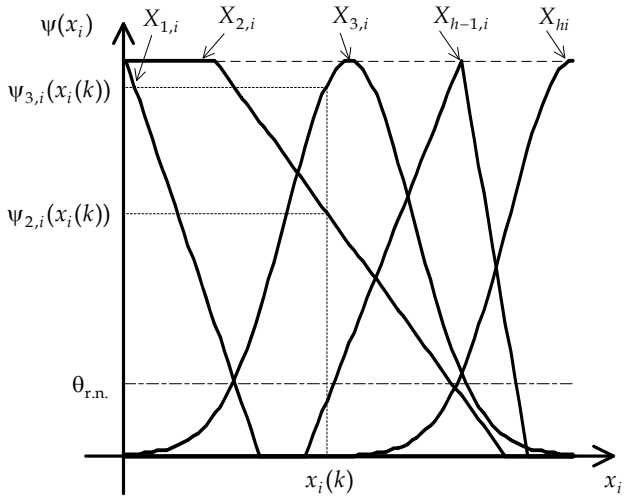

Fig. 9. Terms of linguistic variable for the *i*-th input. Membership functions are adjusted to represent a priori knowledge of input data structure. Incoming signal $x_i(k)$ fires fuzzy receptive neurons $FRN_{2,i}$ and $FRN_{3,i}$

clusters and controls the amount of fuzziness in the final partition, $\|x(k)-v_j\|_A$ is the distance between $x(k)$ and v_j in a certain metric, A is a norm matrix that defines distance metric. By applying the method of indefinite Lagrange multipliers under restrictions

$$\sum_{j=1}^{m}\mu_j(x(k))=1,\ k=\overline{1,N}\,, \tag{34}$$

$$0<\sum_{k=1}^{N}\mu_j(x(k))<N,\ j=\overline{1,m}\,, \tag{35}$$

minimization of (33) leads us to the following solution:

$$\mu_j(x(k))=\frac{\left(\left\|x(k)-v_j\right\|_A^2\right)^{\frac{1}{1-\zeta}}}{\sum_{\iota=1}^{m}\left(\left\|x(k)-v_\iota\right\|_A^2\right)^{\frac{1}{1-\zeta}}}, \tag{36}$$

$$v_j=\frac{\sum_{k=1}^{N}\mu_j^{\zeta}(x(k))x(k)}{\sum_{k=1}^{N}\mu_j^{\zeta}(x(k))} \tag{37}$$

that originates the methods of so-called fuzzy probabilistic clustering (Bezdek et al., 2005). In the case when norm matrix A is the identity matrix and $\zeta=1$, equations (36), (37) present hard c-means algorithm, and for $\zeta=2$, they are conventional fuzzy c-means algorithm.

Efficiency of fuzzy probabilistic clustering decreases in the presence of noise. Algorithm (36), (37) produces unnaturally high degree of membership for outliers that are equidistant from clusters centers. This drawback is avoided by applying fuzzy possibilistic approach that is based on the following objective function:

$$E(\mu_j(x(k)), v_j) = \sum_{k=1}^{N}\sum_{j=1}^{m}\mu_j^{\zeta}(x(k))\left\|x(k)-v_j\right\|_A^2 + \sum_{j=1}^{m}\lambda_j\sum_{k=1}^{N}\left(1-\mu_j(x(k))\right)^{\zeta}, \quad (38)$$

where $\lambda_j > 0$ is the scalar parameter that defines the distance at which membership level takes the value 0.5, i.e. if $\left\|x(k)-v_j\right\|_A^2 = \lambda_j$, then $\mu_j(x(k)) = 0.5$. Minimization of (38) with respect to $\mu_j(x(k))$, v_j, and λ_j yields the following solution:

$$\mu_j(k) = \left(1+\left(\frac{\left\|x(k)-v_j\right\|_A^2}{\lambda_j}\right)^{\frac{1}{\zeta-1}}\right)^{-1}, \quad (39)$$

$$v_j = \frac{\sum_{k=1}^{N}\mu_j^{\zeta}(k)x(k)}{\sum_{k=1}^{N}\mu_j^{\zeta}(k)}, \quad (40)$$

$$\lambda_j = \frac{\sum_{k=1}^{N}\mu_j^{\zeta}(k)\left\|x(k)-v_j\right\|_A^2}{\sum_{k=1}^{N}\mu_j^{\zeta}(k)} \quad (41)$$

that gives conventional possibilistic c-means algorithm if $\zeta = 2$ and A is the identity matrix (Bezdek et al., 2005).

After spiking neural network learning has been done, center $c_j^{[1]}$ of a spiking neuron SN_j represents center v_j of a certain data cluster, and its firing time $t_j^{[1]}(x(k))$ reflects distance $\left\|x(k)-v_j\right\|_A$ in temporal sense (Natschlaeger & Ruf, 1998; Bohte et al., 2002). This notion allows us of using self-learning spiking neural network output in fuzzy clustering algorithms described above. In order to implement fuzzy clustering on the base of the spiking neural network, its architecture is modified in the following way: lateral connections in the second hidden layer are disabled, and output fuzzy clustering layer is added next to spiking neuron layer. Such modification is applied to the spiking neural network on data clustering stage only. Output fuzzy clustering layer receives information on the distances of input patter to centers of all spiking neurons and produces fuzzy partition using either probabilistic approach (36), (37) as follows (Bodyanskiy & Dolotov, 2008a-b):

$$\mu_j(x(k)) = \frac{\left(t_j^{[1]}(x(k))\right)^{\frac{2}{1-\zeta}}}{\sum_{\iota=1}^{m}\left(t_{\iota}^{[1]}(x(k))\right)^{\frac{2}{1-\zeta}}}, \quad (42)$$

or possibilistic approach (39)-(41) as follows (Bodyanskiy et al., 2008b):

$$\mu_j(x(k)) = \left(1 + \left(\frac{\left(t_j^{[1]}(x(k))\right)^2}{\lambda_j} \right)^{\frac{1}{\zeta-1}} \right)^{-1}, \tag{43}$$

$$\lambda_j = \frac{\sum_{k=1}^{N} \mu_j^{\zeta}\left(t_j^{[1]}(x(k))\right)^2}{\sum_{k=1}^{N} \mu_j^{\zeta}(x(k))}. \tag{44}$$

Obviously, the learning algorithm (11) is more natural here then (7) since response of each spiking neuron within the second hidden layer matters for producing fuzzy partition by the output layer.

The advantage of fuzzy clustering based on self-learning spiking neural network is that it is not required to calculate centers of data clusters according to (37) or (40) as the network finds them itself during learning.

7. Simulation experiment

The proposed self-learning fuzzy spiking neural network was tested on the coloured Lenna image shown on Fig. 10a (USC-SIPI Image Database). The image is a standard benchmark that is widely used in image processing. The image has 3 layers (RGB) with spatial dimensions 512×512 so the set to process is formed of 262144 three-dimensional data points (n=3). The purpose was to separate classes by colour of pixels avoiding their spatial location. Obviously, some classes overlap one another as three RGB-components define a plenty of colours, and the boundary between colours is indistinct. There were considered 8 classes to be separated on the image (m=8). A certain grade of grey was assigned to each of the eight classes to visualize the obtained results. 30% of the image pixels were randomly selected to generate a training set (Fig. 10b).

Self-leaning fuzzy spiking neural network settings were set as follows (the most settings were taken from (Berredo, 2005)): time step is 0.1 sec, h=6, receptive neuron type – crisp, $\theta_{\text{r.n.}} = 0.1$, $t_{\max}^{[0]} = 20$ sec, $\tau_{PSP} = 3$ sec, $q = 16$, $d^1 = 0$, $d^{16} = 15$, minimum value of a synaptic weight is 0, maximum value is 1, simulation interval length is 30 sec, $\eta_w = 0.35$, $\alpha = -2.3$ sec, $\beta = 0.2$, $\nu = 5$ sec, $\theta_{\text{s.n.}}(0) = 9$, $\theta_{\text{s.n.}}(K+1) = \theta_{\text{s.n.}}(K) + \frac{0.3 \cdot \theta_{\text{s.n.}}(K)}{K_{\max}}$, $K_{\max} = 3$, neighbourhood function – Gaussian, $\rho(0) = 6$, $\gamma = 0.5$, calculating $\rho(K)$ – expression (15), fuzzy clustering – probabilistic, $\zeta = 2$, defuzzification method – the largest value. Results of image processing produced by the spiking neural network on the 1st and the 3rd epochs are shown on Fig. 10c and Fig. 10d, respectively.

Fuzzy c-means algorithm was also trained over the same testing set ($\zeta = 2$, defuzzification method – the largest value). Results of image processing produced by the algorithm on the 3rd and the 30th epochs are shown on Fig. 10e and Fig. 10f, respectively.

Thus, self-learning fuzzy spiking neural network requires a number of epochs that is in an order less then conventional fuzzy c-means algorithm requires.

Fig. 10. The Lenna image processing: a) Original image; b) Training set (30% of the original image); c) The 1st epoch of self-learning fuzzy spiking neural network learning; d) The 3rd epoch of self-learning fuzzy spiking neural network learning; e) The 3rd epoch of fuzzy c-means learning; f) The 30th epoch of fuzzy c-means learning

8. Conclusion

Spiking neural networks are more realistic models of real neuronal systems than artificial neural networks of the previous generations. Nevertheless, they can be described, as it was shown in earlier sections, in a strict technically plausible way based on the Laplace transform. Spiking neural network designed in terms of transfer functions is an analog-digital nonlinear dynamic system that conveys and processes information both in pulse-position and continuous-time forms. Such precise formal description of spiking neural network architecture and functioning provides researchers and engineers with a framework to construct hardware implementations of various spiking neural networks for image processing of different levels of complexity.

Networks of spiking neurons introduced new, biologically more plausible essence of information processing and gave rise to a new, computationally more powerful generation of computational intelligence hybrid systems. In the present chapter, self-learning fuzzy spiking neural network that combined spiking neural network and fuzzy probabilistic and fuzzy possibilistic clustering algorithms was described as an example of such hybrid systems. It was shown that using of hybrid systems constructed on a spiking neural network basis made it possible to reduce number of learning epochs as compared to conventional fuzzy clustering algorithms. In addition, the way to 'fuzzify' spiking neural network architecture was demonstrated with consideration of a pool of receptive neurons to be a linguistic variable.

Although the temporal Hebian learning algorithm of spiking neural network is biologically plausible, even more realistic learning algorithm based on 'Winner-Takes-More' rule was proposed as its improvement.

Both theoretical innovations and simulation experiment presented in this chapter confirmed that self-learning spiking neural network and hybrid systems developed on its basis are powerful and efficient advanced tool of computational intelligence for data clustering and, particularly, for image processing.

9. References

Berredo, R.C.de (2005). *A review of spiking neuron models and applications,* M. Sc. Dissertation, Pontifical Catholic University of Minas Gerais, Belo Horizonte, http://www.loria.fr/~falex/pub/Teaching/disserta.pdf.

Bezdek, J.C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms,* Plenum Press, ISBN 0-306-40671-3, New York.

Bezdek, J.C.; Keller, J.; Krishnapuram, R. & Pal, N.R. (2005). *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Springer, ISBN 0-387-24515-4, New York.

Bodyanskiy, Ye. & Rudenko, O. (2004). *Artificial Neural Networks: Architectures, Learning, Applications*, Teletekh, ISBN 966-95416-2-2, Kharkiv (in Russian).

Bodyanskiy, Ye. & Dolotov, A. (2008a). Image processing using self-learning fuzzy spiking neural network in the presence of overlapping classes, *Proceedings of the 11th International Biennial Baltic Electronics Conference "BEC 2008",* pp. 213-216, ISBN 978-1-4244-2060-5, Tallinn/Laulasmaa, Estonia, October 2008, Tallinn University of Technology, Tallinn.

Bodyanskiy, Ye. & Dolotov, A. (2008b). A self-learning spiking neural network for fuzzy clustering task, *Scientific Proceedings of Riga Technical University, Computer Science*

Series: Information Technology and Management Science, Vol. 36, pp. 27-33, ISSN 1407-7493.

Bodyanskiy, Ye. & Dolotov, A. (2009). Hebbian learning of fuzzy spiking neural network based on 'Winner-Takes-More' rule, *Proceedings of the 11th International Conference on Science and Technology "System Analysis and Information Technologies",* p. 271, ISBN 978-966-2153-27-9, Kyiv, May 2009, ESC "IASA" NTUU "KPI", Kyiv.

Bodyanskiy, Ye., Dolotov, A. & Pliss, I. (2008a). Fuzzy receptive neurons using in self-learning spiking neural network, *Proceedings of International Scientific and Technical Conference "Automation: Problems, Ideas, Solutions",* pp. 12-14, ISBN 978-966-2960-32-7, Sevastopil, September 2008, Publishing House of SevNTU, Sevastopil (in Russian).

Bodyanskiy, Ye.; Dolotov, A.; Pliss, I. & Viktorov, Ye. (2008b). Fuzzy possibilistic clustering using self-learning spiking neural network, *Wissenschaftliche Berichte der Hochschule Zittau/Goerlitz,* Vol. 100, pp. 53-60, ISBN 3-9808089-9-9.

Bodyanskiy, Ye.; Dolotov, A. & Pliss, I. (2009). Self-learning fuzzy spiking neural network as a nonlinear pulse-position threshold detection dynamic system based on second-order critically damped response units, In: *International Book Series "Information Science and Computing": No. 9, Intelligent Processing,* K. Markov, P. Stanchev, K. Ivanova, I. Mitov, (Eds.), pp.63-70, Institute of Information Theories and Applications FOI ITHEA, ISSN 1313-0455, Sofia.

Bohte, S.M.; Kok, J.N. & La Poutre, H. (2002). Unsupervised clustering with spiking neurons by sparse temporal coding and multi-layer RBF networks, *IEEE Transactions on Neural Networks*, Vol. 13, pp.426-435, ISSN 1045-9227.

Cottrell, M. & Fort, J.C. (1986). A stochastic model of retinotopy: a self-organizing process, *Biological Cybernetics,* Vol. 53, No. 6, pp. 405-411, ISSN 0340-1200.

Cybenko, G. (1989). Approximation by superposition of a sigmoidal function, *Mathematics of Control, Signals, and Systems,* Vol. 2, pp.303-314, ISSN 0932-4194.

Dayan, P. & Abbott, L.F. (2001). *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems,* MIT Press, ISBN 0-262-04199-5, Cambridge.

Dorf, R.C. & Bishop, R.H. (1995). *Modern Control Systems,* Addison-Wesley, ISBN 0-201-84559-8, Reading.

Feldbaum, A.A. & Butkovskiy, A.G. (1971). *Methods of Automatic Control Theory,* Nauka, Moscow (in Russian).

Gerstner, W.; Kempter, R.; van Hemmen, J.L. & Wagner, H. (1996). A neuronal learning rule for sub-millisecond temporal coding, *Nature,* Vol. 383, pp.76-78, ISSN 0028-0836.

Gerstner, W. & Kistler, W.M. (2002). *Spiking Neuron Models: Single Neurons, Populations, Plasticity,* Cambridge University Press, ISBN 0-521-81384-0, Cambridge.

Goodwin, G.C.; Graebe, S.F. & Salgado, M.E. (2001). *Control System Design,* Prentice Hall, ISBN 0-13-958653-9, Upper Saddle Reiver.

Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation,* Prentice Hall, ISBN 0-13-273350-1, Upper Saddle River.

Hopfield, J.J. (1995). Pattern recognition computation using action potential timing for stimulus representation, *Nature,* Vol. 376, pp. 33-36, ISSN 0028-0836.

Hornik, K.; Stinchombe, M. & White, H. (1989). Multilayer feedforward networks are universal approximators, *Neural Networks,* Vol. 2, No. 5, pp. 359-366, ISSN 0893-6080.

Jang, J.-Sh.R.; Sun, Ch.-T. & Mizutani, E. (1997). *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence,* Prentice Hall, ISBN 0-13-261066-3, Upper Saddle River.

Kohonen, T. (1995). *Self-Organizing Maps,* Springer, ISBN 3-540-58600-8, Berlin.

Kraft, M; Kasinski, A. & Ponulak, F. (2006). Design of the spiking neuron having learning capabilities based on FPGA circuits, *Proceedings of the 3rd IFAC Workshop on Discrete-Event System Design "DESDes'06",* pp. 301-306, ISBN 83-7481-035-1, Rydzyna, September 2006, University of Zielona Gora Press, Zielona Gora.

Lindblad, T. & Kinser, J.M. (2005). *Image Processing Using Pulse-Coupled Neural Networks,* Springer, ISBN 3-540-28293-9, Berlin.

Maass, W. (1997a). Fast sigmoidal networks via spiking neurons, *Neural Computation,* Vol. 9, No. 2, pp. 279-304, ISSN 0899-7667.

Maass, W. (1997b). Networks of spiking neurons: the third generation of neural network models, *Neural Networks,* Vol. 10, No. 9, pp. 1659-1671, ISSN 0893-6080.

Maass, W. & Bishop, C.M. (1998). *Pulsed Neural Networks,* MIT Press, ISBN 0-262-13350-4, Cambridge.

Malsburg, C. von der (1994). The correlation theory of brain function, In: *Models of Neural Networks II: Temporal Aspects of Coding and Information Processing in Biological Systems,* E. Domany, J.L. van Hemmen, K. Schulten, (Eds.), pp. 95-119, Springer-Verlag, ISBN: 978-0-387-94362-6, New York.

McCulloch, W.S. & Pitts, W.A. (1943). A logical calculus of ideas immanent in nervous activity, *Bulletin of Mathematical Biophysics,* Vol. 5, pp. 115-133, ISSN 0092-8240.

Meftah, B.; Benyettou, A.; Lezoray, O. & QingXiang, W. (2008). Image clustering with spiking neuron network, *Proceedings of the International Joint Conference on Neural Networks "IJCNN 2008", part of the IEEE World Congress on Computational Intelligence "WCCI 2008",* pp. 681-685, ISBN 978-1-4244-1821-3, Hong Kong, June 2008, IEEE Press, Piscataway.

Minsky, M.L. & Papert, S.A. (1969). *Perceptrons: An Introduction to Computational Geometry,* MIT Press, ISBN 0-262-63022-2, Cambridge.

Natschlaeger, T. & Ruf, B. (1998). Spatial and temporal patterns analysis via spiking neurons, *Network: Computation in Neural Systems,* Vol. 9, No. 3, pp. 319-332, ISSN 1361-6536.

Phillips, C.L. & Harbor, R.D. (2000). *Feedback Control Systems,* Prentice Hall, ISBN 0-13-949090-6, Upper Saddle River.

Pikovsky, A.; Rosenblum, M. & Kurths, J. (2001). *Synchronization: A Universal Concept in Nonlinear Sciences,* Cambridge University Press, ISBN 0-521-59285-2, Cambridge.

Ritter, H. & Schulten, K. (1986). On the stationary state of Kohonen's self-organizing sensory mapping, *Biological Cybernetics,* Vol. 54, No. 2, pp. 99-106, ISSN 0340-1200.

Sato-Ilic, M. & Jain, L.C. (2006). *Innovations in Fuzzy Clustering: Theory and Applications,* Springer, ISBN 978-3-540-34356-1, Berlin.

Scott, A. (2002). *Neuroscience: A Mathematical Primer,* Springer, ISBN 0-387-95403-1, New York.

Schoenauer, T.; Atasoy, S.; Mehrtash, N. & Klar, H. (2000). Simulation of a digital neuro-chip for spiking neural networks, *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Network "IJCNN 2000", Vol. 4,* pp. 490-495, ISBN 0-7695-0619-4, Como, July 2000, IEEE Computer Society, Los Alamitos.

Tsypkin, Ya.Z. (1984). *Relay Control Systems,* Cambridge University Press, ISBN 0-521-24390-4, Cambridge.

USC-SIPI Image Database, the. Lenna, 4.2.04, *University of Southern California,* Signal & Image Processing Institute, Electrical Engineering Department, http://sipi.usc.edu/database/database.cgi?volume=misc&image=12.

Image Processing

Edited by Yung-Sheng Chen

ISBN 978-953-307-026-1
Hard cover, 516 pages
Publisher InTech
Published online 01, December, 2009
Published in print edition December, 2009

There are six sections in this book. The first section presents basic image processing techniques, such as image acquisition, storage, retrieval, transformation, filtering, and parallel computing. Then, some applications, such as road sign recognition, air quality monitoring, remote sensed image analysis, and diagnosis of industrial parts are considered. Subsequently, the application of image processing for the special eye examination and a newly three-dimensional digital camera are introduced. On the other hand, the section of medical imaging will show the applications of nuclear imaging, ultrasound imaging, and biology. The section of neural fuzzy presents the topics of image recognition, self-learning, image restoration, as well as evolutionary. The final section will show how to implement the hardware design based on the SoC or FPGA to accelerate image processing.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821