37. Computational Modeling with Spiking Neural Networks

Stefan Schliebs, Nikola Kasabov

This chapter reviews recent developments in the area of spiking neural networks (SNN) and summarizes the main contributions to this research field. We give background information about the functioning of biological neurons, discuss the most important mathematical neural models along with neural encoding techniques, learning algorithms, and applications of spiking neurons. As a specific application, the functioning of the evolving spiking neural network (eSNN) classification method is presented in detail and the principles of numerous eSNN based applications are highlighted and discussed.

37.1 Neurons and Brain

The brain is arguably the most complex organ of the human body. It contains approximately 10^{11} neurons, which are the elementary processing units of the brain. These neurons are interconnected and form a complex and very dense neural network. On average 1 cm^3 of brain matter contains 10^4 cell bodies and several kilometers of *wire*, i. e., connections between neurons in the form of branching cell extensions.

Like most cells in the human body, neurons maintain a certain ion concentration across their cell membrane. Therefore, the membrane contains ion pumps which actively transport sodium ions from the intracellular to the extra-cellular liquid. Potassium ions are pumped in the opposite direction from the outside to the inside of the cell. In addition to the ion pumps, a number of specialized proteins, so-called ion channels, are embedded in the membrane. They allow a slow inward flow of sodium ions into the cell, while potassium ions leak outwards into the extra-cellular liquid. Thus, the ion streams at the channels have opposite directions to the ion pumps. Furthermore, since both ion streams differ in their strengths, an electrical potential exists across the cell membrane. The inside of the cell is negatively charged in relation the extra-cellular liquid. The membrane is polarized which is the resting condition of the neuron CE0.

A large variety of neural shapes and sizes exist in the brain. A typical neuron is illustrated in Fig. 37.1. The central part of the neuron is called the soma, in which the nucleus is located. It contains the genetic in-

CE0 Please check for correctness of meaning
CE1 Please check that this is correct; "neuron" has been inserted

Editor's or typesetter's annotations (will be removed before the final TeX run)

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

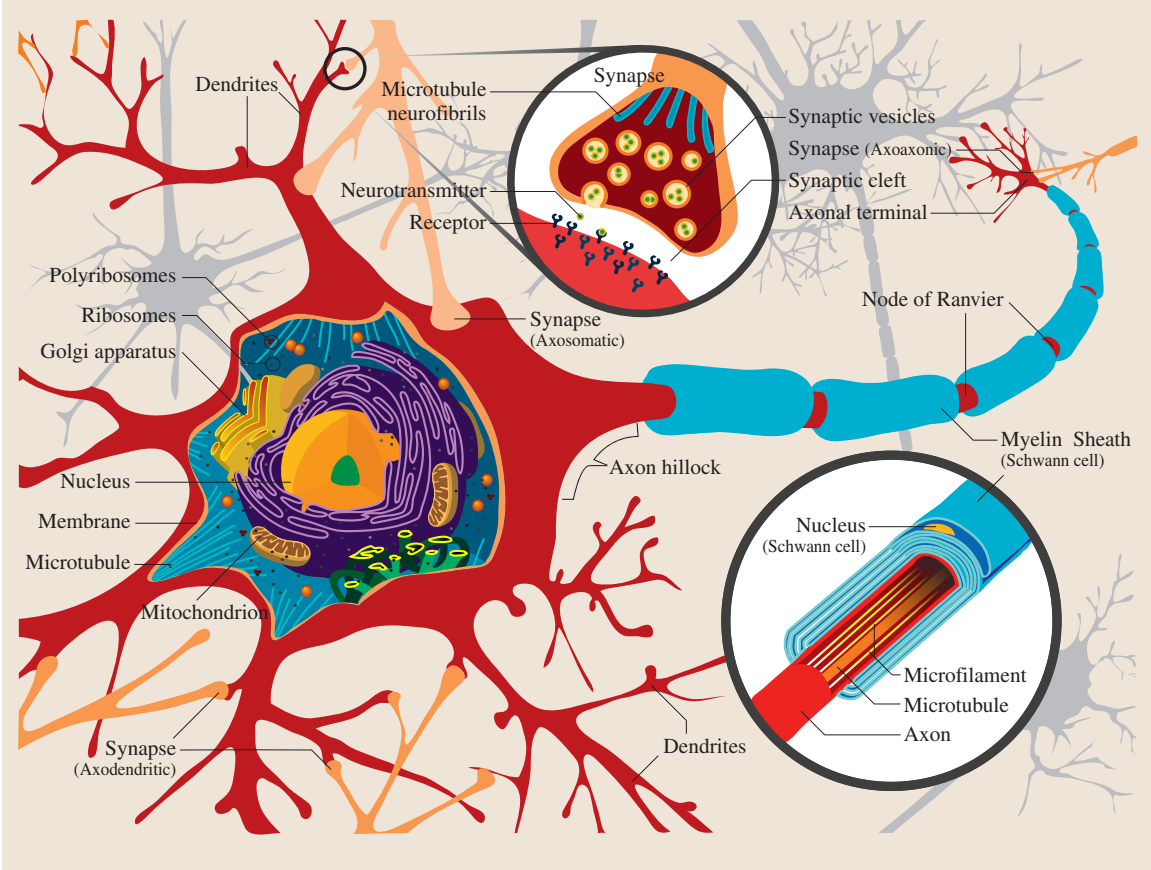


Fig. 37.1 Schematic illustration of a typical neuron CE[1] in the human brain. The main part of the neuron is the soma containing the genetic information, the dendrites, and the axon, which are responsible for the reception and emission of electrical signals. Signal transmission occurs at the synapse between two neurons, see text for detailed explanations

formation of the cell, i. e., the DNA, from which genes are expressed and proteins constructed that are important for the functioning of the cell. The cell body has a number of cellular branch-like extensions known as dendrites. Dendrites are specialized for *receiving* electrical signals from other neurons that are connected to them. These signals are short pulses of electrical activity, also known as spikes or action potentials. If a neuron is stimulated by the spike activity of surrounding neurons and the excitation is strong enough, the cell triggers a spike. The spike is propagated via the axon, a long thin wire-like extension of the cell body, to the axonal terminals. These terminals in turn are connected to the dendrites of surrounding neurons and allow the transfer of information from one neuron to the other. Thus an axon is responsible for *sending* information to other neurons connected to it. An axon may be covered by myelin sheaths that allow a faster propagation of electrical signals. These sheaths act as insulators and prevent the dissipation of the depolarization wave caused by an electrical spike triggered in the soma.

Information exchange between two neurons occurs at a synapse, which is a specialized structure that links two neurons together. A synapse is illustrated in the upper middle part of Fig. 37.1. The sending neuron is called pre-synaptic neuron, while the neuron receiving the signal is called post-synaptic. Sending information involves the generation of an action potential in the soma of the pre-synaptic cell. As described before, this potential is propagated through the axon of the neuron to the axonal terminals. These terminals contain the synapses in which neurotransmitter chemicals are stored. Whenever a spike is propagated through the axon, a portion of these neurotransmitters is released

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

into a small gap between the two neurons also known as the synaptic cleft. The neurotransmitter diffuses into the cleft and interacts with specialized receptor proteins of the post-synaptic neuron. The activation of these receptors causes the sodium ion channels to open, which in turn results in the flow of sodium ions from the extracellular liquid into the post-synaptic cell. The ionic concentration across the membrane equalizes rapidly and the membrane depolarizes. Immediately after the depolarization the potassium channels open. As a consequence potassium ions stream outside the cell, which causes the re-polarization of the membrane. The process of de- and re-polarization, i. e., the action potential, lasts only around 2 ms, which explains the name spike or pulse.

A synaptic transmission can be either excitatory or inhibitory depending on the type of the transmitting synapse. Different neurotransmitters and receptors are involved in excitatory and inhibitory synaptic transmissions, respectively. Excitatory synapses release a transmitter called L-glutamate and increase the likelihood of the post-synaptic neuron triggering an action potential following stimulation. On the other hand, inhibitory synapses release a neurotransmitter called GABA and decrease the likelihood of a post-synaptic potential.

The efficacy of a synapse, i. e., the strength of the post-synaptic response due to the neurotransmitter release in the synapse, is not fixed. The increase or decrease of the efficacy of a synapse is called *synaptic plasticity* and it enables the brain to learn and to memorize. There are several different possibilities to accomplish synaptic plasticity. One way is to change the time period of receptor activity in the post-synaptic neuron. Longer periods of receptor activity cause the ion channels to remain open for a longer time, which in turn results in a larger amount of ions flowing into the post-synaptic cell. Thus, the post-synaptic response increases. Short periods of receptor activity have the opposite effect.

Another way to change the synaptic efficacy is to increase or decrease the number of receptors, which would have a direct impact on the number of opened ion channels and as a consequence on the post-synaptic potential. The third possibility is a change of the amount of neurotransmitter chemicals released into the synaptic cleft. Here larger/smaller amounts would increase/decrease the synaptic efficacy.

Comprehensive information and details about the structure, functions, chemistry, and physiology of neurons can be found in a standard textbook on the matter by [37.1].

37.2 Models of Spiking Neurons

The remarkable information processing capabilities of the brain have inspired numerous mathematical abstractions of biological neurons. Spiking neurons represent the third generation of neural models, incorporating the concepts of time, neural, and synaptic state explicitly into the model [37.2]. Earlier artificial neural networks were described in terms of mean firing rates and used continuous signals for transmitting information between neurons. Real neurons, however, communicate by short pulses of electrical activity. In order to simulate and describe biologically plausible neurons in a mathematical and formal way, several different models have been proposed in the recent past. Figure 37.2 illustrates schematically the mathematical abstraction of a biological neuron.

Neural modeling can be described on several levels of abstraction. On the microscopic level, the neuron model is described by the flow of ions through the channels of the membrane. This flow may, among other

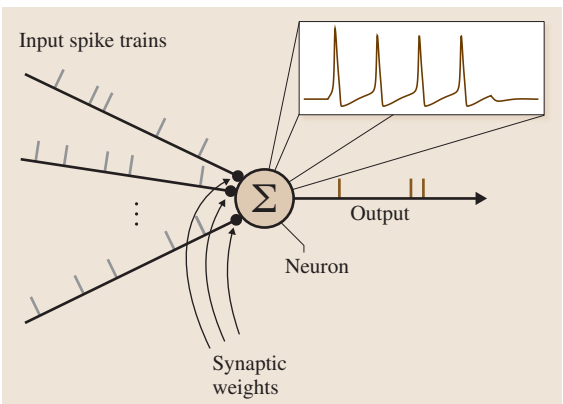


Fig. 37.2 Schematic illustration of a mathematical neuronal model. The model receives electrical stimulation in form of spikes through a number of connected pre-synaptic neurons. The efficacy of a synapse is modeled in the form of synaptic weights. Most models focus on the dynamics of the post-synaptic potential only. Output spikes are propagated via the axon to connected post-synaptic neurons ▶

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

things, depend on the presence or absence of various chemical messenger molecules. Models at this level of abstraction include the Hodgkin–Huxley model [37.3] and the compartment models that describe separate segments of a neuron by a set of ionic equations.

On the other hand, the macroscopic level treats a neuron as a homogeneous unit, receiving and emitting spikes according to some defined internal dynamics. The underlying principles of how a spike is generated and carried through the synapse, dendrite, and cell body are not relevant. These models are typically known under the term integrate-and-fire models.

In the next sections the major neural models are discussed and their functions are explained. Since the macroscopic neuronal models are more relevant, the focus of the survey is put on these models. The only microscopic model presented here is the Hodgkin–Huxley model, due to its high significance for the research area of neuroscience.

37.2.1 Hodgkin–Huxley Model

This model dates back to the work of *Alan Lloyd Hodgkin* and *Andrew Huxley* in 1952 where they performed experiments on the giant axon of a squid [37.3]. Due to the significance of their contribution to neuroscience, both received the 1963 Nobel Prize in Physiology and Medicine. The model is a detailed description of the influences of the conductance of ion channels on the spike activity of the axon. The diameter of the squid's giant axon is approximately 0.5 mm and is visible to the naked eye. Since electrodes had to be inserted into the axon, its large size was a big advantage for biological analysis at that time.

Hodgkin and *Huxley* discovered three different ion currents in a neuron: a sodium, potassium, and a leak current. Voltage-dependent ion channels control the flow of ions through the cell membrane. Due to an active transport mechanism, the ion concentration within the cell differs from that in the extra-cellular liquid, resulting in an electrical potential across the cell membrane. In the mathematical model such a membrane is described as an electrical circuit consisting of a capacitor, resistors, and batteries that model the ion channels, (Fig. 37.3). The current I at a time t splits into the current stored in the capacitor and the additional currents passing through each of the ion channels

$$I(t) = I_{\text{cap}}(t) + \sum_k I_k(t)\,, \tag{37.1}$$

where the sum runs over all ion channels.

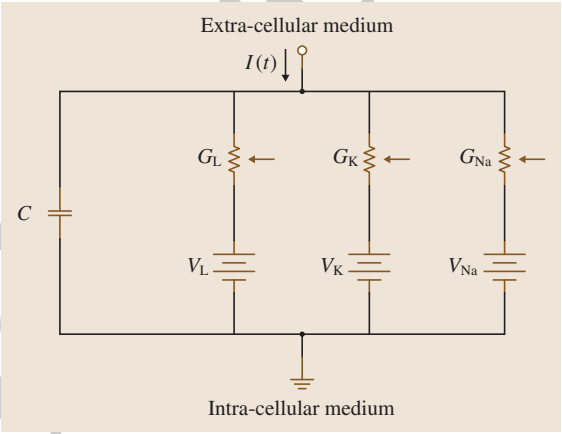


Fig. 37.3 Schematic illustration of the Hodgkin–Huxley model in the form of an electrical circuit (after [37.3]). The model represents the biophysical properties of the cell membrane of a neuron. The semipermeable cell membrane separates the interior of the cell from the extra-cellular liquid and thus acts as a capacitor. Ion movements through the cell membrane (in both directions) are modeled in the form of (constant and variable) resistors. In the diagram the conductance of the resistors $G_x = 1/R_x$ is shown. Three ionic currents exist: A sodium current (Na ions), potassium current (K ions), and a small leakage current (L) that is primarily carried by chloride ions

Substituting $I_{\text{cap}}(t) = C\,\mathrm{d}u/\mathrm{d}t$ by applying the definition of the capacitance $C = Q/u$, where Q is the charge and u the voltage across the capacitor leads to

$$C\frac{\mathrm{d}u}{\mathrm{d}t} = -\sum_k I_k(t) + I(t)\,. \tag{37.2}$$

As mentioned earlier, in the Hodgkin–Huxley model three ion channels are modeled: A sodium current, potassium current, and a small leakage current that is primarily carried by chloride ions. Hence the sum in (37.1) consists of three different components that are formulated as

$$\sum_k I_k(t) = G_{\text{Na}} m^3 h(u - V_{\text{Na}}) + G_{\text{K}} n^4 (u - V_{\text{K}}) + G_{\text{L}}(u - V_{\text{L}})\,, \tag{37.3}$$

where V_{Na}, V_{K}, and V_{L} are constants called reverse potentials. Variables G_{Na} and G_{K} describe the maximum conductance of the sodium and potassium channel, respectively, while the leakage channel is voltage-independent with a conductance of G_{L}. The variables m, n, and h are gating variables whose dynamics are

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

Table 37.1 Parameters of the Hodgkin–Huxley model. The membrane capacitance is $C = \mu\text{F/cm}^2$. The voltage scale is shifted in order to have a resting potential of zero

x	V_x (mV)	G_x (mS/cm^2)
Na	115	120
K	−12	36
L	10.6	0.3
x	$\alpha_x(u)$	$\beta_x(u)$
n	$\frac{0.1-0.01u}{\exp(1-0.1u)-1}$	$0.125\exp\left(-\frac{u}{80}\right)$
m	$\frac{2.5-0.1u}{\exp(2.5-0.1u)-1}$	$4\exp\left(-\frac{u}{18}\right)$
h	$0.07\exp\left(-\frac{u}{20}\right)$	$\frac{1}{\exp(3-0.1u)+1}$

described by differential equations of the form

$$\frac{m}{\mathrm{d}t} = \alpha_m(u)(1-m) - \beta_m(u)m \,, \tag{37.4}$$

$$\frac{n}{\mathrm{d}t} = \alpha_n(u)(1-n) - \beta_n(u)n \,, \tag{37.5}$$

$$\frac{h}{\mathrm{d}t} = \alpha_h(u)(1-h) - \beta_h(u)h \,, \tag{37.6}$$

where m and h control the sodium channel and variable n the potassium channel. Functions α_x and β_x, where $x \in \{m, n, h\}$, represent empirical functions of the voltage across the capacitor u, that need to be adjusted in order to simulate a specific neuron. Using a well-parameterized set of the above equations, *Hodgkin* and *Huxley* were able to describe a significant amount of data collected from experiments with the giant axon of a squid. The parameters discovered of the model are given in Table 37.1.

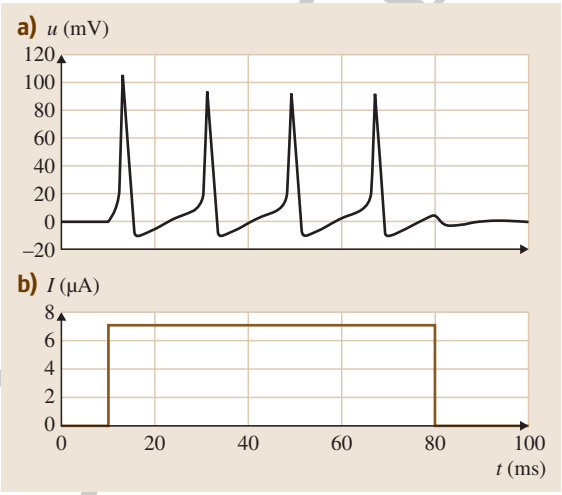


Fig. 37.4 **(a)** Evolution of the membrane potential u for a content input current I_0 using the Hodgkin–Huxley model. The current is switched on at time $t = 10\,\text{ms}$ for a duration of 70 ms. **(b)** The stimulus is strong enough to generate a spike train across the cell membrane (*upper diagram*). As soon as the input current vanishes ($I = 0$), the electrical potential returns to its resting potential ($u = 0$)

The dynamics of the Hodgkin–Huxley model are presented in Fig. 37.4. For the simulation, the parameter values from Table 37.1 are utilized. The membrane is stimulated by a constant input current $I_0 = 7\,\mu\text{A}$, switched on at time $t = 10\,\text{ms}$ for a duration of 70 ms. The current is switched off at time $t = 80\,\text{ms}$. For $t < 10\,\text{ms}$, no input stimulus occurs and the potential across the membrane stays at the resting potential. For $10\,\text{ms} \le t \le 80\,\text{ms}$ the current is strong enough to generate a sequence of spikes across the cell membrane. At time $t > 80\,\text{ms}$ and input current $I = 0$, the electrical potential returns to its resting potential.

Additional reading on the Hodgkin–Huxley model can be found in the excellent review of [37.4], which also summarizes the historical developments of the model. A guideline for computer simulations of the model using the simulation platform GENESIS (general neural simulation system) can be found in [37.5].

37.2.2 Leaky Integrate-and-Fire Model (LIF)

The Hodgkin–Huxley model can reproduce electrophysiological measurements very accurately. Nevertheless, the model is computationally costly and simpler, more phenomenological models are required for the simulation of larger networks of spiking neurons. The leaky integrate-and-fire neuron (LIF) may be the best-known model for simulating spiking networks efficiently. The model has a long history and was first proposed by *Lapicque* in 1907, long before the actual mechanisms of action potential generation were known [37.6]. Discus-

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

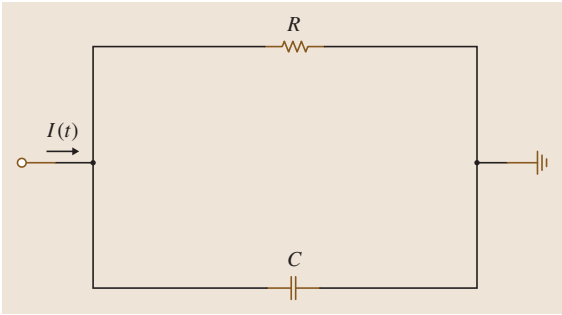


Fig. 37.5 Schematic illustration of the leaky integrate-and-fire model in the form of an electrical circuit. The model consists of a capacitor C in parallel with a resistor R, driven by a current $I = I(R) + I_{\text{cap}}$

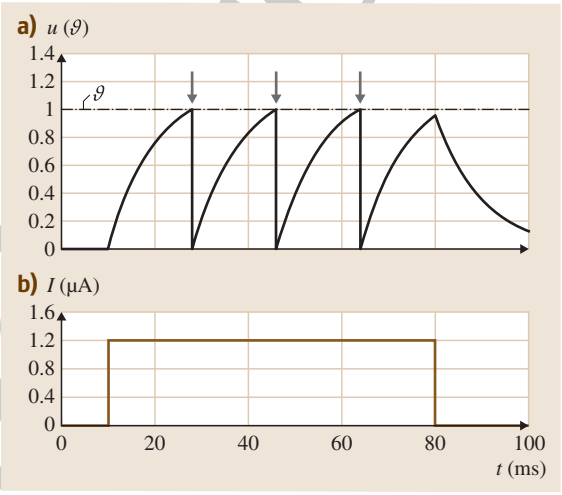


Fig. 37.6 **(a)** Evolution of the potential u for a constant input current I_0 using the leaky integrate-and-fire model. The membrane potential u is given in units of the threshold ϑ. The current is switched on at time $t = 10$ ms for a duration of 70 ms. **(b)** The stimulus is strong enough to generate a sequence of spike trains (*dark arrows*). As soon as the input current vanishes, the potential returns to its resting potential

sions of this work can be found in [37.7, 8]. However, it was *Knight* who introduced the term *integrate-and-fire* in [37.9]. He called these models *forgetful*, but the term *leaky* quickly became more popular.

Similar to the Hodgkin–Huxley model, the LIF model is based on the idea of an electrical circuit (Fig. 37.5). The circuit contains a capacitor with capacitance C and a resistor with a resistance R, where both C and R are assumed to be constant. The current $I(t)$ splits into two currents

$$I(t) = I_{\text{R}} + I_{\text{cap}} \,, \tag{37.7}$$

where I_{cap} charges the capacitor and I_{R} passes through the resistor. Substituting $I_{\text{cap}} = C\,\mathrm{d}u/\mathrm{d}t$ using the definition for capacity, and $I_{\text{R}} = u/R$ using Ohm's law, where u is the voltage across the resistor, one obtains

$$I(t) = \frac{u(t)}{R} + C\frac{\mathrm{d}u}{\mathrm{d}t} \,. \tag{37.8}$$

Replacing $\tau_{\text{m}} = RC$ yields the standard form of the model

$$\tau_{\text{m}}\frac{\mathrm{d}u}{\mathrm{d}t} = -u(t) + RI(t) \,. \tag{37.9}$$

The constant τ_{m} is called the membrane time constant of the neuron. Whenever the membrane potential u reaches a threshold ϑ, the neuron fires a spike and its potential is reset to a resting potential u_{r}. It is noteworthy that the shape of the spike itself is not explicitly described in the traditional LIF model. Only the firing times are considered to be relevant. Nevertheless, it is possible to include the shape of spikes as well [37.10].

A LIF neuron can be stimulated by either an external current I_{ext} or CE2 by the synaptic input current I_{syn} from pre-synaptic neurons. The external current $I(t) = I_{\text{ext}}(t)$ may be constant or represented by a function of time t. Figure 37.6 presents the dynamics of a LIF neuron stimulated by an input current $I_0 = 1.2$. The current is strong enough to increase the potential u until the threshold ϑ is reached. As a consequence, a spike is triggered and the potential resets to $u_{\text{r}} = 0$. After the reset, the integration process starts again. At $t = 80$ ms, the current is switched off and the potential returns to its resting potential due to leakage.

If a LIF neuron is part of a network of neurons, it is usually stimulated by the activity of its pre-synaptic neurons. The resulting synaptic input current of a neuron i is the weighted sum over all spikes generated by pre-synaptic neurons j with firing times $t_j^{(\text{f})}$

$$I(t) = I_{\text{syn}_i}(t) = \sum_j w_{ij} \sum_f \alpha\left(t - t_j^{(\text{f})}\right) \,. \tag{37.10}$$

The weights w_{ij} reflect the efficacy of the synapse from neuron j to neuron i. Negative weights correspond to inhibitory synapses, while positive weights correspond to excitatory synapses. The time course of the post-synaptic current $\alpha(\cdot)$ can be defined in various ways. In the simplest form it is modeled by Dirac pulse $\delta(x)$, which has a non-zero function value for $x = 0$ and zero for all others. Thus the input current caused by a pre-synaptic neuron decreases/increases the potential u in

CE2 Please check that this is the intended meaning.

Editor's or typesetter's annotations (will be removed before the final TeX run)

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

a step-wise manner. More realistic models often employ different functions usually in the form $x\exp(-x)$, which is typically referred to as an α function.

In Fig. 37.7, a LIF neuron is stimulated by a spike train from a single pre-synaptic neuron. The post-synaptic current is modeled in the form of a Dirac pulse as described above. This results in a step-wise increase of the post-synaptic potential. If the potential reaches the threshold ϑ, a spike is triggered and the potential resets. Due to its simplicity, many LIF neurons can be connected to form large networks, while still allowing an efficient simulation.

Extensive additional information about the LIF model can be found in the excellent textbook [37.11] and in the two recent reviews by *Burkitt* [37.12, 13].

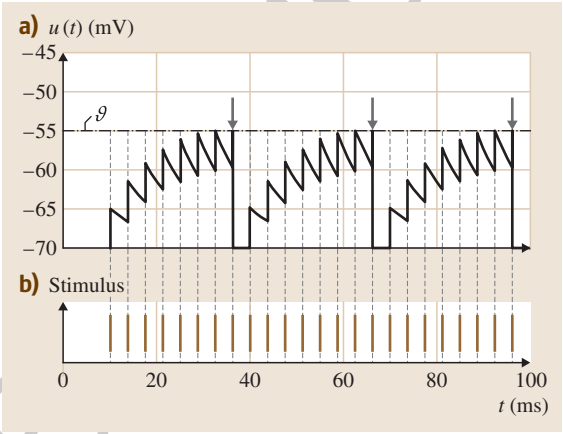


Fig. 37.7 Dynamics of the leaky integrate-and-fire model. The potential u increases due to the effect of pre-synaptic input spikes. If the membrane potential crosses a threshold ϑ, a spike is triggered (*straight dark arrows*). The shape of this action potential is not explicitly described by the model, only the time of the event is of relevance. The synapse may have either an inhibitory or an excitatory effect on the post-synaptic potential that is determined by the sign of the synaptic weights

37.2.3 Izhikevich Model

Another neural model was proposed in [37.14]. It is based on the theory of dynamical systems. The model claims to be as biologically plausible as the Hodgkin–Huxley model while offering the computational complexity of LIF models. Depending on its parameter configuration, the model reproduces different spiking and bursting behavior of cortical neurons. Its dynamics are governed by two variables

$$\frac{\mathrm{d}v}{\mathrm{d}t} = 0.04v^2 + 5v + 140 - u + I\,, \tag{37.11}$$

$$\frac{\mathrm{d}u}{\mathrm{d}t} = a(bv - u)\,, \tag{37.12}$$

where v represents the membrane potential of the neuron and u is a membrane recovery variable, which provides negative feedback for v. If the membrane potential reaches a threshold $\vartheta = 30\,\mathrm{mV}$, a spike is triggered and a reset of v and u occurs

$$\text{if } v \geq 30\,\mathrm{mV}, \quad \text{then } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases}. \tag{37.13}$$

Variables a, b, c, d are parameters of the model. Depending on their setting, a large variety of neural characteristics can be modeled. Each parameter has an associated interpretation. Parameter a represents the decay rate of the membrane potential, b is the sensitivity of the membrane recovery, and c and d reset v and u, respectively.

In Fig. 37.8, the meaning of the parameters is graphically explained along with their effect on the dynamics of the model. For example, if we want to produce a regular spiking neuron, we would set $a = 0.02$, $b = 0.25$, $c = -65$, and $d = 8$.

More information on this model can be found in the textbook on dynamical systems in neuroscience [37.15]. There are also a number of articles on the topic, e.g., the work on the suitability of mathematical models for simulation of cortical neurons [37.16] and the large-scale simulation of a mammalian thalamocortical system [37.17], which involves one million neurons and almost half a billion synapses.

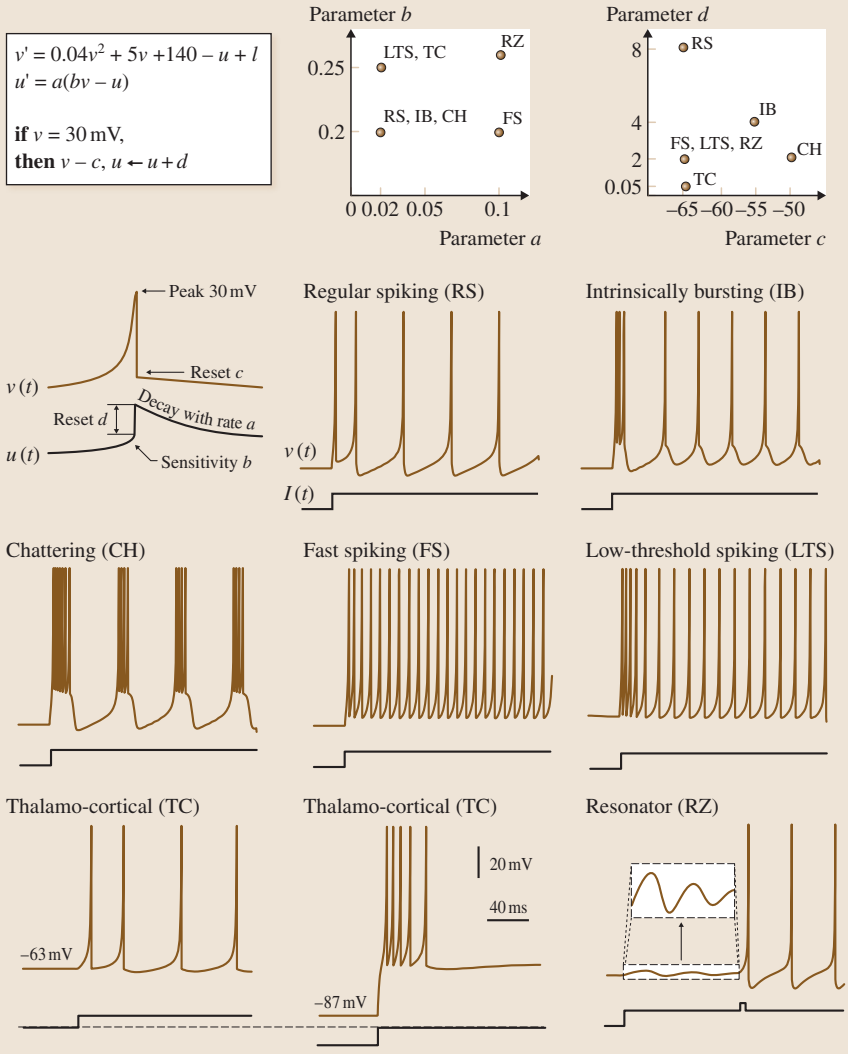
37.2.4 Spike Response Model (SRM)

The spike response model (SRM) is a generalization of the LIF model and was introduced in [37.11]. In this model, the state of a neuron is characterized by a single variable u. A number of different kernel functions describe the impact of pre-synaptic spikes and external stimulation on u, but also the shape of the actual spike and its after-potential. Whenever the state u reaches a threshold ϑ from below, i. e., $u(t) = \vartheta$ and $\mathrm{d}u(t)/\mathrm{d}t > 0$, a spike is triggered. In contrast to the LIF model, the threshold ϑ in SRM is not required to be fixed, but may depend on the last firing time $\hat{t}_i$ of neuron i. For example, the threshold might be increased after the neuron has spiked (also known as the refractory period) to avoid triggering another spike during that time.

TS3 Please check this URL for correctness. It seems no longer to be valid.

Editor's or typesetter's annotations (will be removed before the final TeX run)

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET



Let $u_i(t)$ be the state variable that describes neuron i at time t and $\hat{t}_i$ is the last time when the neuron emitted a spike, then the evolution of $u_i(t)$ can be formulated as

$$u_i(t) = \eta(t - \hat{t}_i) + \sum_j w_{ij} \sum_f \epsilon_{ij}(t - \hat{t}_i, t - t_j^{(f)}) + \int_0^\infty \kappa(t - \hat{t}_i, s) I_{\text{ext}}(t - s)\,\mathrm{d}s\,, \qquad (37.14)$$

where $t_j^{(f)}$ are the firing times of pre-synaptic neurons j, while w_{ij} represents the synaptic efficacy between neuron j and i.

Functions η, ϵ, and κ are response kernels. The first kernel, η, is the reset kernel. It describes the dynamics of an action potential and becomes non-zero each time a neuron fires. This kernel models the reset of the state u and its after-potential. A typical implementation is[TS4]

$$\eta(t - \hat{t}_i) = \eta_0 \left\{ K_1 \exp\left(-\frac{t - \hat{t}_i}{\tau_m}\right) - K_2 \left[\exp\left(-\frac{t - \hat{t}_i}{\tau_m}\right)\right] - \exp\left(-\frac{t - \hat{t}_i}{\tau_s}\right) \right\} \theta(t - \hat{t}_i)\,, \qquad (37.15)$$

[TS4] Added missing bracket and adjusted brackets to handbook style. Please confirm.

Editor's or typesetter's annotations (will be removed before the final TeX run)

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

Fig. 37.8 Dynamics of the Izhikevich model. Depending on the settings of the parameters a, b, c, and d, different neuron characteristics are modeled (generated by a freely available simulation tool provided by Izhikevich on http://www.izhikevich.com) TS[3] ◀

where $\eta_0 = \vartheta$ equals the firing threshold of the neuron. The first term in (37.15) models the positive pulse with a decay rate τ_m and the second one is the negative spike after-potential with a decay rate τ_s, while K_1 and K_2 act as scaling factors. Function $\Theta(\cdot)$ is a step function known as the Heaviside function

$$\Theta(s) = \begin{cases} 0 & \text{if } s < 0 \\ 1 & \text{if } s \geq 0 \end{cases} , \tag{37.16}$$

which ensures that the effect of the η kernel is zero if the neuron has not emitted a spike, i. e., $t < \hat{t}$. The shape of this kernel is presented in Fig. 37.9a, where $K_1 = 1$, $K_2 = 5$, $\tau_s = 0.005$, and $\tau_m = 0.01$ were used.

The second kernel determines the time course of a post-synaptic potential whenever the neuron receives an input spike. The kernel depends on the last firing time of the neuron $t - \hat{t}$ and on the firing times $t - t_j^{(f)}$ of the pre-synaptic neurons j. Due to the first dependence the post-synaptic neuron may respond differently to input spikes received immediately after a post-synaptic spike. A typical implementation of this kernel is e.g.,

$$\epsilon\left(t - \hat{t}, t - t_j^{(f)}\right) = \left[\exp\left(-\frac{t - t_j^{(f)}}{\tau_m}\right) - \exp\left(-\frac{t - t_j^{(f)}}{\tau_s}\right)\right] \times \Theta(t - t_j^{(f)}) , \tag{37.17}$$

where $\Theta(\cdot)$ once more corresponds to the Heaviside function, the two exponential functions model a positive and a negative pulse with the corresponding decay rates, and $t_j^{(f)}$ is the spike time of a pre-synaptic neuron j. In (37.17), the first dependency of ϵ is neglected, which corresponds to a special case of the model, namely the simplified SRM. This simplified version of SRM is discussed in the next section. The time course of the ϵ kernel of (37.17), is presented in Fig. 37.9a. For the figure $\tau_s = 0.005$ and $\tau_m = 0.01$ were used. The implementations for the response kernels η and ϵ are adopted from the study on spike timing dependent plasticity in [37.18].

The third kernel function κ represents the linear response of the membrane to an input current I_{ext}. It depends on the last firing time of the neuron $t - \hat{t}$ and the time prior to t. It is used to model the time course of u due to external stimuli to the neuron.

A comprehensive discussion of the spike response model and its derivatives can be found in the excellent textbook [37.11] and also in [37.19].

Simplified Spike Response Model (SRM$_0$)

In a simplified version of SRM, the kernels ϵ and κ are replaced

$$\epsilon_0(s) = \epsilon_{ij}(\infty, s) , \tag{37.18}$$

$$\kappa_0(s) = \kappa_{ij}(\infty, s) , \tag{37.19}$$

which makes the kernels independent of the index j of pre-synaptic neurons and also of the last firing time $\hat{t}_i$ of the post-synaptic neuron. Using simple implementations of these kernel functions reduces the computational cost significantly. Hence, this model has been used to analyze the computational power of spiking neurons [37.20, 21], of network synchro-

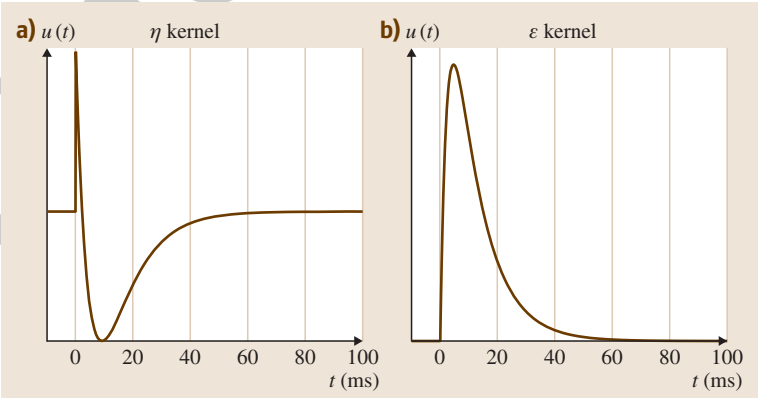


Fig. 37.9a,b Shape of the response kernels η and ϵ. **(a)** A spike is triggered at time $t = t^{(f)} = 0$, which results in the activation of the η kernel. The shape of the spike and its after potential are modeled by this kernel function. **(b)** The neuron receives an input spike at time $t = 0$, which results in the activation of the ϵ kernel. If no further stimulus is received, the potential u returns to its resting potential

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

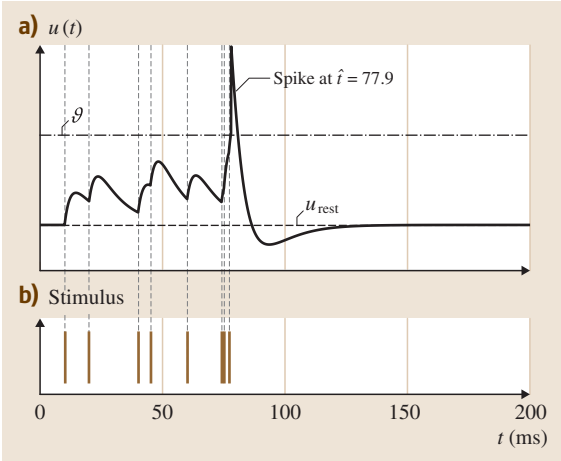


Fig. 37.10 Dynamics of the spike response model (SRM). In the post-synaptic neuron, spikes change the membrane potential described by the kernel function ϵ. If the membrane potential crosses a threshold ϑ, a spike is triggered. The shape of this action potential is modeled by the function η.

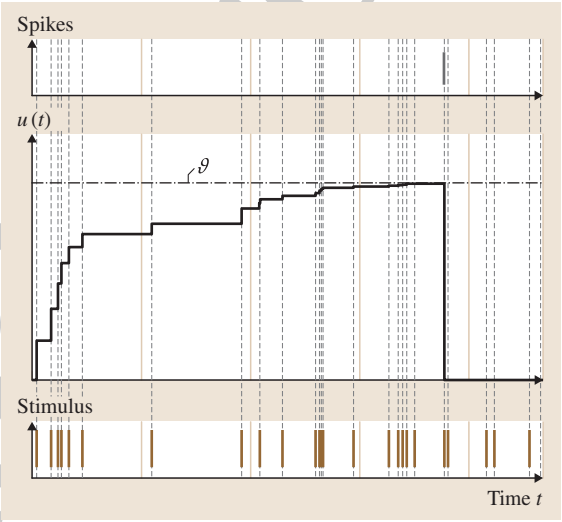


Fig. 37.11 Evolution of the post-synaptic potential (PSP) of the Thorpe neuronal model for a given input stimulus. If the potential reaches threshold ϑ, a spike is triggered and the PSP is set to 0 for the rest of the simulation, even if the neuron is still stimulated by incoming spike trains

nization [37.22] and collective phenomena of coupled networks [37.23].

The dynamics of the SRM_0 model are presented in Fig. 37.10. For the diagram, the ϵ and η kernels are defined by (37.15) and (37.17), respectively. The neuron receives a pre-synaptic stimulus in the form of several spikes which impact the potential u according to the response kernel ϵ. Due to the pre-synaptic activity, an action potential is triggered at time $t = 77.9$ ms, which results in the activation of the η kernel and the modeling of the spike shape and the after-potential. Figure 37.10 only presents excitatory synaptic activity.

37.2.5 Thorpe Model

A simplified LIF model was formally proposed in [37.24]. However, the general idea of the model can be traced back to publications from as early as 1990 [37.25]. This model lacks the post-synaptic potential leakage. The spike response of a neuron depends only on the arrival time of pre-synaptic spikes. The importance of early spikes is boosted and affects the post-synaptic potential more strongly than later spikes. This concept is very interesting due to the fact that the brain is able to compute even complex tasks quickly and reliably. For example, for the processing of visual data the human brain requires only approximately 150 ms [37.26], see also a similar study on rapid visual categorization of natural and artificial objects [37.27]. Since it is known that this type of computation is partly sequential and several parts of the brain involving millions of neurons participate in the computation, it was argued in [37.28, 29] that each neuron has time and energy to emit only very few spikes that can actually contribute to the processing of the input. As a consequence, few spikes per neuron are biologically sufficient to solve a highly complex recognition task in real time.

Similar to other models, the dynamics of the Thorpe model are described by the evolution of the post-synaptic potential $u_i(t)$ of a neuron i as

$$u_i(t) = \begin{cases} 0 & \text{if fired} \\ \sum_{j \mid f(j) < t} w_{ji} m_i^{\text{order}(j)} & \text{else} \end{cases}, \qquad (37.20)$$

where w_{ji} is the weight of a pre-synaptic neuron j, $f(j)$ is the firing time of j, and $0 < m_i < 1$ is a parameter of the model, namely the modulation factor. Function order(j) represents the rank of the spike emitted by neuron j. For example, a rank order$(j) = 0$ would be assigned if neuron j is the first among all pre-synaptic neurons of i that emits a spike. In a similar fashion, the spikes of all pre-synaptic neurons are ranked and then used in the computation of u_i. A neuron i fires a spike when its potential reaches a certain threshold ϑ. After

SPIN: 12742608 (Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

emitting a spike, the potential resets to $u_i = 0$. Each neuron is allowed to emit only a single spike at most. The threshold $\vartheta = cu_{\max}$ is set to a fraction $0 < c < 1$ of the maximum potential $u_{\max}$ reachable for a neuron. Figure 37.11 presents the change of the post-synaptic potential for the Thorpe neural model if a series of input spikes stimulates the neuron through different synapses.

These simplifications allow a very fast real-time simulation of large networks. Due to its low computational costs this model was mainly used for studying image and speech recognition methods involving thousands of connected neurons [37.30, 31]. Many studies have investigated the Thorpe model, e.g., for face recognition [37.32, 33]. Additional studies utilizing this model are presented in Sect. 37.6, where principles and applications of the evolving spiking neural network architecture are discussed in the next sections.

37.3 Neural Encoding

This section addresses a fundamental question in neuroscience: What is the code used by neurons to transmit information? Is it possible for an external observer to read and understand the message of neural activity? Traditionally, there are two main theories about neural encoding – pulse codes and rate codes. Both theories are discussed below.

37.3.1 Rate Codes

The first theory assumes that the mean firing rate of a neuron carries the most, maybe even all the information of a transmission. These codes are referred to as rate codes and have inspired the classical perceptron approaches. The mean firing rate v is usually understood as the ratio of the average number of spikes n_{sp} observed over a specific time interval T, and T itself

$$v = \frac{n_{\text{sp}}}{T} \,. \tag{37.21}$$

This concept has been especially successful in the context of sensory or motor neural system, cf. e.g., the pioneering work by *Adrian* on the direct relationship between the firing rate of stretch receptor neurons and the applied force in the muscles of frog legs [37.34]. Nevertheless, the idea of a mean firing rate has been repeated CE5 [37.35]. The main argument is the comparably slow transmission of information from one neuron to another, since each neuron has to integrate the spike activity of pre-synaptic neurons at least over a time T. Especially, the extremely short response times of the brain for certain stimuli, cannot be explained by the temporal averaging of spikes. For example, in [37.26] it was shown that the human brain can recognize a visual stimulus in approximately 150 ms. It is known that a moderate number of neural layers are involved in the processing of visual stimuli. If every layer had to wait a period T to receive the information from the previous layer, the recognition time would be much longer.

However, there is also another interpretation for the concept of the mean firing rate. It is defined as the average spike activity over a population of neurons. The principle of this interpretation is explained in Fig. 37.12. A post-synaptic neuron receives stimulating inputs in the form of spikes emitted by a population of pre-synaptic neurons. This population produces a certain spike activity A, which is defined as the fraction of neurons being active within a short interval $[t, t + \Delta t]$

$$A = \frac{1}{\Delta t} \frac{n_{\text{act}}(t, t + \Delta t)}{N} \,, \tag{37.22}$$

where $n_{\text{act}}(t, t + \Delta t)$ denotes the number of active neurons in interval $[t, t + \Delta t]$, and N is the total number

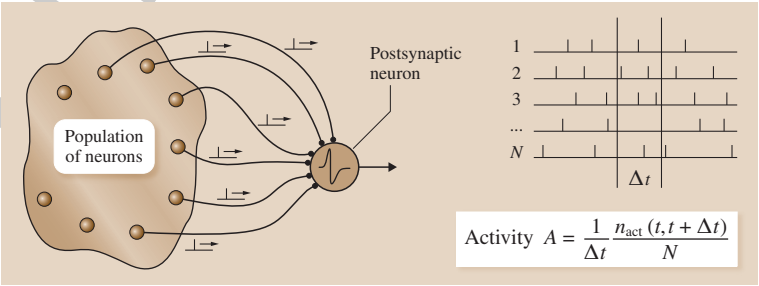


Fig. 37.12 A neuron receives input spikes from a population of pre-synaptic neurons producing a certain activity A. The activity is defined as the fraction of neurons being active within a short interval $[t, t + \Delta t]$, divided by the population size N and the time period Δt. (After [37.11])

CE5 Please check that this is the intended meaning.

Editor's or typesetter's annotations (will be removed before the final TeX run)

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

neuron in the population. The activity of a population may vary rapidly and thus allow fast responses of the neurons to changing stimuli [37.36, 37].

37.3.2 Pulse Codes

The second type of neural encoding is referred to as a spike or pulse code. These codes assume the precise spike time as the carrier of information between neurons. Experimental evidence for temporal correlations between spikes has been given through computer simulations [37.38], where integrate-and-fire models are investigated, but also through biological experiments, cf. the electrophysiological recordings and staining procedures in [37.39]; see also the in vivo measurements described in [37.40] in which spatio–temporal patterns of neuronal activity are analyzed in order to predict the behavior responses of rats.

A pulse code based on the timing of the first spike after a reference signal was discussed in [37.26]. This encoding is called time-to-first-spike and was inspired by the visual processing of the human eye. It was argued that each neuron has time to emit only a few spikes that can contribute to the overall processing of a stimulus. Indeed, it was also shown in [37.41] that a new stimulus is processed in the first 20–50 ms after its onset. Thus, earlier spikes carry most information about the stimulus. A specific neural model, namely the Thorpe model that boosts the importance of early spikes, was discussed already in Sect. 37.2.5.

Other pulse codes consider correlation and synchrony to be important. Neurons that represent a similar concept, object, or label are *labeled* by firing synchronously [37.42]. More generally, any precise spatio–temporal pulse pattern may be potentially meaningful and encode a particular information. Neurons that fire with a certain relative time delay may signify a certain stimulus.

As a practical example, the so-called rank order population encoding is presented in Sect. 37.6.1. Additional information about neural encoding in general can be found in the book by *Rieke* et al. [37.35].

37.4 Learning in SNN

This section presents some typical learning methods in the context of spiking neurons. A variety of problems impair the development of learning procedures for SNN. The explicit time dependence results in asynchronous information processing that commonly requires complex software and/or hardware implementations to simulate these neural networks. Additional difficulties are added by the fact that recurrent network topologies are commonly used in SNN and thus the formulation of a straightforward learning method, such as back-propagation for MLP, is not possible.

Similar to traditional neural networks, three different learning paradigms can be distinguished in SNN, which are referred to as unsupervised, reinforcement, and supervised learning. Reinforcement learning in SNN is probably the least common among the three. Some algorithms have been successfully applied in robotic applications [37.43], but were also theoretically analyzed in [37.44–46]. Unsupervised learning in the form of Hebbian learning is the most biologically realistic learning scenario. The so-called *spike-timing dependent plasticity* (STDP) belongs to this category and is discussed in the next section. Supervised techniques impose a certain input–output mapping on the network which is essential for practical applications of SNN. Two methods are discussed in greater detail in the next sections. The learning algorithm employed in the eSNN architecture is discussed separately in Sect. 37.6.2. An excellent comparison of supervised learning methods developed for SNN can be found in [37.47].

37.4.1 STDP – Spike-Timing Dependent Plasticity

Spike-timing dependent plasticity is inspired by the experiments of Hebb published in his famous book *The Organization of Behavior* [37.48]. His essential postulate is often referred to as Hebb's law:

> *When an axon of cell A is near enough to excite cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.*

First experimental evidence that supports Hebb's postulate was given 20 years later in [37.49, 50]. Today, it is known that the change of synaptic efficacy in the brain is correlated to the timing of pre- and post-synaptic activity of a neuron [37.51–53]. Whenever the efficacy of a synapse is strengthened or weakened, we speak of

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

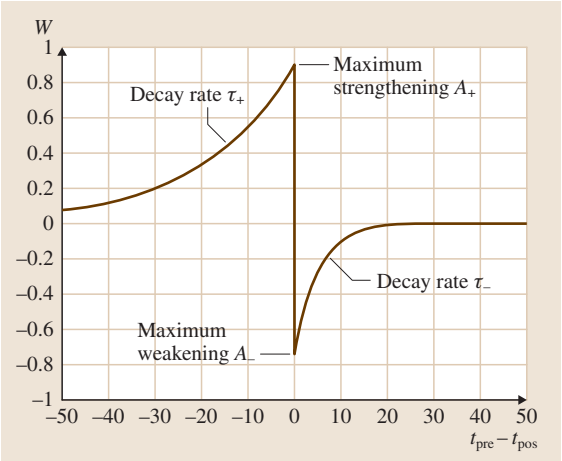


Fig. 37.13 STDP learning window W as function of the time difference $t_{pre} - t_{post}$ of pre- and post-synaptic spike times. The function presented is based on (37.23) using the following parameter setting: $A_+ = 0.9$, $A_- = -0.75$, $\tau_+ = 20$, and $\tau_- = 5$

long-term potentiation (LTP) or long-term depression (LTD), respectively. STDP is described by a function $W(t_{pre} - t_{post})$ that determines the fractional change of the synaptic weight in dependence of the difference between the arrival time t_{pre} of a pre-synaptic spike and the time t_{post} of an action potential emitted by the neuron. Function W is also known as the STDP window. Typical approximations of W are, e.g.,

$$W(t_{pre} - t_{post}) = \begin{cases} A_+ \exp\left(\frac{t_{pre} - t_{post}}{\tau_+}\right) & \text{if } t_{pre} < t_{post}\,, \\ A_- \exp\left(-\frac{t_{pre} - t_{post}}{\tau_+}\right) & \text{if } t_{pre} > t_{post}\,, \end{cases} \tag{37.23}$$

where parameters τ_+ and τ_- determine the temporal range of the pre- and post-synaptic time interval, while A_+ and A_- denote the maximum fractions of synaptic modification, if $t_{pre} - t_{post}$ is close to zero. Figure 37.13 presents the STDP window W according to (37.23).

The parameters for A_+, A_-, τ_+, and τ_- are adjusted according to the particular neuron to be modeled. The window W is usually temporally asymmetric, i. e., $A_+ \neq A_-$ and $\tau_+ \neq \tau_-$. However, there are also some exceptions, e.g., synapses of layer 4 spiny stellate neurons in the rat barrel cortex appear to have a symmetric window [37.54].

A study investigated the dynamics of synaptic pruning as a consequence of the STDP learning rule [37.55]. Synaptic pruning is a general feature of mammalian brain maturation and refines the embryonic nervous system by removing inappropriate synaptic connections between neurons, while preserving appropriate ones. Later studies extended this work by including apoptosis (genetically programmed cell death) into the analysis [37.56], and the identification of spatio–temporal patterns in the pruned network indicating the emergence of cell assemblies [37.57].

More information on STDP can be found in the excellent review on the matter in [37.58–61].

37.4.2 Spike-Prop

Traditional neural networks, like the multi-layer perceptron, usually employ some form of gradient based descent, i. e., error back-propagation, to modify synaptic weights in order to impose a certain input-output mapping on the network. However, the topological recurrence of SNN and their explicit time dependence do not allow a straightforward evaluation of the gradient in the network. Special assumptions need to be made to develop a version of back-propagation appropriate for spiking neurons.

In [37.62, 63] a back-propagation algorithm called spike-prop is proposed, which is suitable for training SNN. It is derived from the spike-response model discussed in Sect. 37.2.4. The aim of the method is to learn a set of desired firing times t_j^{d} of all output neurons j for a given input pattern presented to the network. Spike-prop minimizes the error E defined as the squared difference between all network output times t_j^{out} and desired output times t_j^{d}

$$E = \frac{1}{2} \sum_j \left(t_j^{\mathrm{out}} - t_j^{\mathrm{d}}\right)^2 . \tag{37.24}$$

The error is minimized with respect to the weights w_{ij}^k of each synaptic input

$$\Delta w_{ij}^k = -\eta \frac{\mathrm{d}E}{\mathrm{d}w_{ij}^k} , \tag{37.25}$$

with η defining the learning rate of the update step.

A limitation of the algorithm is given by the requirement that each neuron is allowed to fire only once, which is similar to the limitations of the Thorpe neural model presented in Sect. 37.2.5. This simplification allows the error function defined in (37.24) to depend entirely on the difference between actual and desired spike time. Thus, only time-to-first-spike encoding is suitable in combination with spike-prop.

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

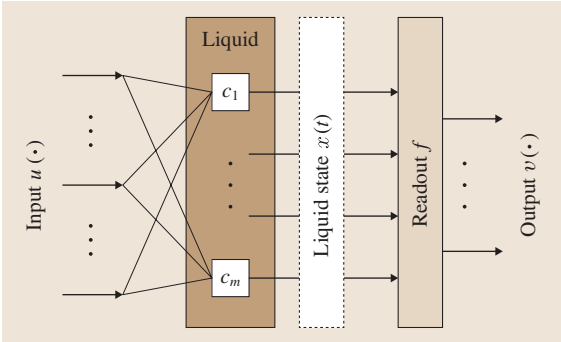


Fig. 37.14 Principle of the liquid state machine (LSM). The liquid transforms inputs u into a liquid state x, which in turn is mapped by a (linear) readout function f into the output v of the network. (After [37.67])

The algorithm was modified in a number of studies. In [37.64] a momentum term was included in the update of the weights, while [37.65] extended the method to learn additional neural parameters, such as synaptic delays, time constants, and neuron thresholds. An extension towards recurrent network topologies was presented in [37.66].

37.4.3 Liquid State Machine (LSM)

A very different approach to neural learning was proposed with the liquid state machine (LSM) introduced in [37.69]. The method is a specific form of reservoir computing [37.70], that constructs a recurrent network of spiking neurons, for which all parameters of the network, i. e., synaptic weights, connectivity, delays, neural parameters, are randomly chosen and fixed during simulation. Such a network is also referred to as a *liquid*. If excited by an input stimulus, the liquid exhibits very complex non-linear dynamics that are expected to reflect the inherent information of the presented stimulus. The response of the network can be interpreted by a learning algorithm.

Figure 37.14 illustrates the principle of the LSM approach. As a first step in the general implementation of LSM a suitable liquid is chosen. This step determines, for example, the employed neural model along with its parameter configuration, as well as the connectivity strategy of the neurons, network size, and other network-related parameters. After creating the liquid, so-called liquid states $x(t)$ can be recorded at various time points in response to numerous different (training) inputs $u(t)$. Finally, a supervised learning algorithm is applied to a set of training examples of the form $(x(t), v(t))$ to train a readout function f, such that the actual outputs $f(x(t))$ are close to $v(t)$.

It was argued in [37.67] that LSM has universal computational power. A very appealing feature of the applied training method, i. e., the readout function, is its simplicity, since only a single layer of weights is actually modified, for which a linear training method is sufficient.

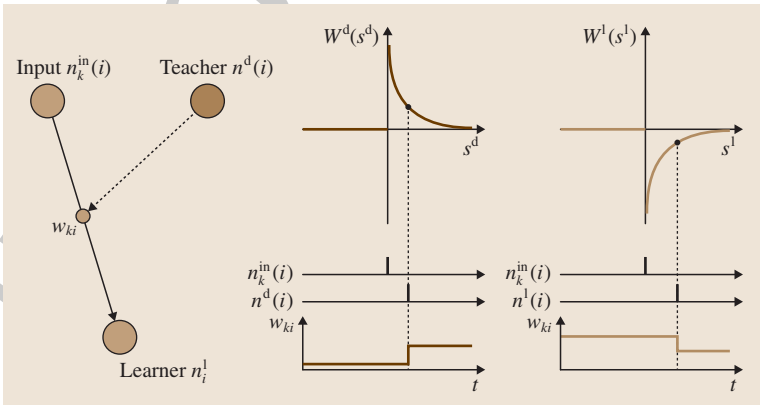


Fig. 37.15 Schematic illustration of the remote supervised method (ReSuMe). The synaptic change depends on the correlation of spike activities between input, teaching, and learning neurons. Spikes emitted by neuron input $n_k^{\text{in}}(i)$ followed by a spike of the teacher neuron $n^{\text{d}}(i)$ leads to an increase of synaptic weight w_{ki}. The value of w_{ki} is decreased, if $n_k^{\text{in}}(i)$ spikes before the learning neuron n_i^{l} is activated. The amplitude of the synaptic change is determined by two functions $W^{\text{d}}(s^{\text{d}})$ and $W^{\text{l}}(s^{\text{l}})$, where s^{d} is the temporal difference between the spike times of teacher neuron and input neuron, while s^{l} describes the difference between the spike times of learning neuron and input neuron. (After [37.68])

SPIN: 12742608 (Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

A specific implementation of the readout, the so-called remote supervised method (ReSuMe) introduced in [37.68], is presented here. The goal of ReSuMe is to impose a desired input–output spike pattern on an SNN, i. e., to produce target spike trains in response to a certain input stimulus. The method is based on the already presented STDP learning window as described in Sect. 37.4.1 for details, in which two opposite update rules for the synaptic weights are balanced. Additional teacher neurons are defined for each synapse, which remotely supervise the evolution of its synaptic weight. The teacher neuron is not explicitly connected to the network, but generates a reference spike signal which is used to update the connection weight in an STDP-like fashion. The post-synaptic neuron, whose activity is influenced by the weight update, is called the learning neuron.

Figure 37.15 illustrates the principle of ReSuMe. Let n_i^{l} denote the learning neuron which receives spike sequences from pre-synaptic neuron $n_k^{\mathrm{in}}, (i)$, the corresponding synaptic weight being w_{ki}, and neuron $n^{\mathrm{d}}(i)$ being the teacher for weight w_{ki}. If input neuron $n_k^{\mathrm{in}}, (i)$ emits a spike that is followed by a spike of the teacher neuron $n^{\mathrm{d}}(i)$, the synaptic weight w_{ki} is increased. On the other hand, if $n_k^{\mathrm{in}}, (i)$ spikes before the learning neuron n_i^{l} is activated, the synaptic weight is decreased. The amplitude of the synaptic change is determined by two functions $W^d(s^d)$ and $W^l(s^l)$, where s^d is the temporal difference between the spike times of teacher neuron and input neuron, while s^{l} describes the difference between the spike times of learning neuron and input neuron. Thus, the precise time difference of spiking activity defines the strength of the synaptic change.

A few studies on LSM can be found in the overview paper in [37.71] and the specific case study for isolated word recognition in [37.72]. More information on ReSuMe is available in [37.73–75].

37.5 Applications of SNN

Traditionally, SNN have been applied in the area of neuroscience to better understand brain functions and principles, the work by *Hodgkin* and *Huxley* [37.3] being among the pioneering studies in the field. A number of main directions for understanding the functioning of the nervous system are given in [37.76]. Here it is argued that a comprehensive knowledge about the anatomy of individual neurons and classes of cells, pathways, nuclei, and higher levels of organization is very important, along with detailed information about the pharmacology of ion channels, transmitters, modulators, and receptors. Furthermore, it is crucial to understand the biochemistry and molecular biology of enzymes, growth factors, and genes that participate in brain development and maintenance, perception, and behavior, learning, and diseases. A range of software systems for analyzing biologically plausible neural models exist, NEURON [37.77] and GENESIS [37.78], being the most prominent ones. Modeling and simulation are fundamental for the understanding of neural processes.

A number of large-scale studies have been undertaken recently to understand the complex behavior of ensembles of spiking [37.17, 79]. The review presented in [37.80] discusses challenges for implementations of spiking neural networks on FPGAs in the context of large-scale experiments.

SNN are also applied in many real-world applications. Notable progress has been made in areas such as speech recognition [37.72], learning rules [37.63], associative memory [37.81], and function approximation [37.82]. Other applications include biologically more realistic controllers for autonomous robots; see [37.83–85] for some interesting examples in this research area.

In Sect. 37.6 we focus on a few applications of the evolving spiking neural network architecture.

37.6 Evolving Spiking Neural Network Architecture

Based on [37.86], an evolving spiking neural network architecture (eSNN) was proposed in [37.87], which was initially designed as a visual pattern recognition system. Other studies have utilized eSNN as a general classification method, e.g., in the context of classifying water and wine samples [37.88]. The method is based on the already discussed Thorpe neural model, in which the importance of early spikes (after the onset of a cer-

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

Index entries on this page
one-pass learning algorithm
topology
eSNN
rank order population encoding
receptive field
Gaussian receptive field
spike-based processing
rank order coding
speech recognition
one-pass learning

tain stimulus) is boosted (Sect. 37.2.5). Synaptic plasticity is employed by a fast supervised one-pass learning algorithm that is explained as part of this section.

In order to classify real-valued data sets, each data sample, i. e., a vector of real-valued elements, is mapped into a sequence of spikes using a certain neural encoding technique. In the context of eSNN, the so-called rank order population encoding is employed, but other encoding may be suitable as well. The topology of eSNN is strictly feed-forward and organized in several layers. Weight modification only occurs on the connections between the neurons of the output layer and the neurons of either the hidden layer or the input layer. The weight modification occurs between the neurons of the output layer and the neurons of either hidden or input layer, only.

In Sect. 37.6.1 the encoding principle used in eSNN is presented, followed by the description of the one-pass learning method and the overall functioning of the eSNN method. Finally, a variety of applications based on the eSNN architecture are reviewed and summarized.

37.6.1 Rank Order Population Encoding

Rank order population encoding is an extension of the rank order encoding introduced in [37.24]. It allows the mapping of vectors of real-valued elements into a sequence of spikes. An implementation based on arrays of receptive fields was first[CE6] described in [37.63]. Receptive fields allow the encoding of continuous values by using a collection of neurons with overlapping sensitivity profiles. Each input variable is encoded independently by a group of M one-dimensional receptive fields. For a variable n an interval $[I^n_{\min}, I^n_{\max}]$ is defined. The Gaussian receptive field of neuron i is given by its center μ_i

$$\mu_i = I^n_{\min} + \frac{2i-3}{2}\frac{I^n_{\max} - I^n_{\min}}{M-2} \tag{37.26}$$

and width σ is

$$\sigma = \frac{1}{\beta}\frac{I^n_{\max} - I^n_{\min}}{M-2}\,, \tag{37.27}$$

with $1 \le \beta \le 2$. Parameter β directly controls the width of each Gaussian receptive field. Figure 37.16 depicts an example encoding of a single variable. For the diagram, $\beta = 2$ was used, the input interval $[I^n_{\min}, I^n_{\max}]$ was set to $[-1.5, 1.5]$, and $M = 5$ receptive fields were used.

More information on rank order coding strategies can be found in [37.89] and the accompanying article [37.33]. Very interesting is also the review on rapid spike-based processing strategies in the context of image recognition presented in [37.90], where most work on the Thorpe neural model and rank order coding is summarized. Rank order coding was also explored for speech recognition problems [37.91] and is a core part of the eSNN architecture.

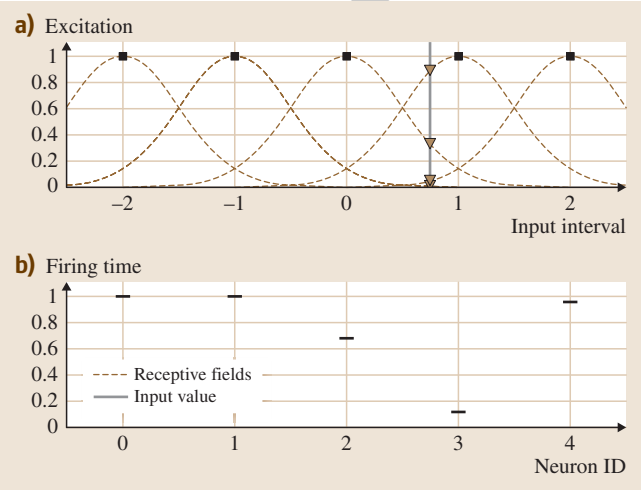


Fig. 37.16a,b Population encoding based on Gaussian receptive fields. **(a)** For an input value $v = 0.75$ (*thick straight line in top figure*) the intersection points with each Gaussian is computed (*triangles*), which are in turn translated into spike time delays **(b)** left [TS7]

37.6.2 One-Pass Learning

The aim of the learning method is to create output neurons, each of them labeled with a certain class label $l \in L$. The number and value of class labels depends on the classification problem to solve, i. e., L corresponds to the set of class labels of the given data set. After presenting a certain input sample to the network, the corresponding spike train is propagated through the SNN, which may result in the firing of certain output neurons. It is also possible that no output neuron is activated and the network remains silent. In this case, the classification result is undetermined. If one or more output neurons have emitted a spike, the neuron with the shortest response time among all activated output neurons is determined, i. e., the output neuron with the earliest spike time. The label of this neuron represents the classification result for the input sample presented.

[CE6] Please check that this is the intended meaning.
[TS7] Please reformulate the last sentence to give a better description for panel b.

Editor's or typesetter's annotations (will be removed before the final TeX run)

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

Algorithm 37.1 Training an evolving spiking neural network

Require: m_l, s_l, c_l for a class label $l \in L$
1: initialize neuron repository $R_l = \{\}$
2: **for all** samples $X^{(i)}$ belonging to class l **do**
3: $w_j^{(i)} \leftarrow (m_l)^{\text{order}(j)}, \forall j \mid j$ pre-synaptic neuron of i
4: $u_{\max}^{(i)} \leftarrow \sum_j w_j^{(i)} (m_l)^{\text{order}(j)}$
5: $\vartheta^{(i)} \leftarrow c_l u_{\max}^{(i)}$
6: **if** $\min(d(w^{(i)}, w^{(k)})) < s_l, \quad w^{(k)} \in R_l$ **then**
7: $w^{(k)} \leftarrow$ merge $w^{(i)}$ and $w^{(k)}$ according to (37.31)
8: $\vartheta^{(k)} \leftarrow$ merge $\vartheta^{(i)}$ and $\vartheta^{(k)}$ according to (37.32)
9: **else**
10: $R_l \leftarrow R_l \cup \{w^{(i)}\}$
11: **end if**
12: **end for**

The learning algorithm successively creates a repository of trained output neurons during the presentation of training samples. For each class label $l \in L$ an individual repository is evolved. The procedure is described in detail in Algorithm 37.1. For each training sample i with class label $l \in L$ a new output neuron is created and fully connected to the previous layer of neurons resulting in a real-valued weight vector $w^{(i)}$, with $w_j^{(i)} \in \mathbb{R}$ denoting the connection between the pre-synaptic neuron j and the created neuron i. In the next step, the input spikes are propagated through the network and the value of weight $w_j^{(i)}$ is computed according to the *order* of spike transmission through a synapse j, see line 6 in Algorithm 37.1

$$w_j^{(i)} = (m_l)^{\text{order}(j)} \forall j | j \text{ pre-synaptic neuron of } i \,. \tag{37.28}$$

Parameter m_l is the modulation factor of the Thorpe neural model. Differently labeled output neurons may have different modulation factors m_l. Function order(j) represents the rank of the spike emitted by neuron j. For example, a rank order(j) = 0 would be assigned, if neuron j is the first among all pre-synaptic neurons of i that emits a spike. In a similar fashion the spikes of all pre-synaptic neurons are ranked and then used in the computation of the weights.

The firing threshold $\vartheta^{(i)}$ of the created neuron i is defined as the fraction $c_l \in \mathbb{R}$, $0 < c_l < 1$, of the maximal possible potential $u_{\max}^{(i)}$, see lines 4 and 5 in Algorithm 37.1,

$$\vartheta^{(i)} = c_l u_{\max}^{(i)} \,, \tag{37.29}$$

$$u_{\max}^{(i)} = \sum_j w_j^{(i)} (m_l)^{\text{order}(j)} \,. \tag{37.30}$$

The fraction c_l is a parameter of the model and for each class label $l \in L$ a different fraction can be specified.

The weight vector of the trained neuron is then compared the ones of neurons that are already stored neurons in the repository, cf. line 6 in Algorithm 37.1. If the minimal Euclidean distance between the weight vectors of the neuron i and an existing neuron k is smaller than a specified similarity threshold s_l, the two neurons are considered too *similar* and both the firing thresholds and the weight vectors are merged according to

$$w_j^{(k)} \leftarrow \frac{w_j^{(i)} + N w_j^{(k)}}{1+N}, \quad \forall j \mid j \text{ pre-synaptic neuron of } i \tag{37.31}$$

$$\vartheta^{(k)} \leftarrow \frac{\vartheta^{(i)} + N \vartheta^{(k)}}{1+N} \,. \tag{37.32}$$

Integer N denotes the number of samples previously used to update neuron k. The merging is implemented as the (running) average of the connection weights, and the (running) average of the two firing thresholds. After the merging, the trained neuron i is discarded and the next sample processed. If no other neuron in the repository is similar to the trained neuron i, the neuron i is added to the repository as a new output neuron.

Figure 37.17 depicts the eSNN architecture. Due to the incremental evolution of output neurons, it is possible to accumulate knowledge as it becomes available. Hence, a trained network is able to learn new data without the need for re-training on already learnt samples. Real-world applications of the eSNN architecture are discussed in the next section.

37.6.3 Applications

The eSNN architecture is used in a variety of applications that are described and summarized here.

Visual Pattern Recognition

Among the earliest applications of eSNN is the visual pattern recognition system presented in [37.87], which extends the work of [37.92, 93] by including the on-line learning technique described before. In [37.87, 94] the method was studied on an image data set consisting of 400 faces of 40 different persons. The task here

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

evolving spiking neural network architecture (eSNN)
Gaussian receptive field
auditory pattern recognition

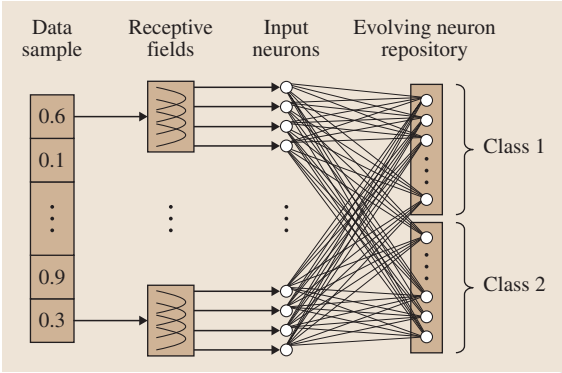


Fig. 37.17 Schematic illustration of the evolving spiking neural network architecture (eSNN). Real-valued vector elements are mapped into the time domain using rank order population encoding based on Gaussian receptive fields. As a consequence of this transformation input neurons emit spikes at pre-defined firing times, invoking the one-pass learning algorithm of the eSNN. The learning iteratively creates repositories of output neurons, one repository for each class. Here a two-class problem is presented. Due to the evolving nature of the network, it is possible to accumulate knowledge as it becomes available, without the requirement of re-training with already learnt samples

was to predict the class labels of presented images correctly. The system was trained on a subset of the data and then tested on the remaining samples of the data. Classification results were similar to [37.92, 93] with the additional advantages of the novel on-line learning method.

In a later study another processing layer was added to the system, which allows efficient multi-view visual pattern recognition [37.95]. The additional layer accumulates information over several different views of an image in order to reach a final decision about the associated class label of the frames. Thus, it is possible to perform an efficient on-line person authentication through the presentation of a short video clip to the system, although the audio information was ignored in this study.

The main principle of this image recognition method is briefly outlined here. The neural network is composed of four layers of Thorpe neurons, each of them grouping a set of neurons into several two-dimensional maps, so-called neural maps. Information in this network is propagated in a feed-forward manner, i. e., no recurrent connections exist. An input frame in the form of a gray-scale image is fed into the first neural layer (L_1), each pixel of the image corresponding to one neuron in a neural map of L_1. Several neural maps may exist in this layer. The map consists of *on* and *off* neurons that are responsible for the enhancement of the high contrast parts of the image. Each map is configured differently and thus is sensitive to different gray scales in the image. The output of this layer is transformed into the spike domain using rank order encoding as described in [37.24]. As a consequence of this encoding, pixels with higher contrast are prioritized in the neural processing.

The second layer, denoted L_2, consists of orientation maps. Each map is selective for different directions, e.g., 0, 45, . . . , 315°, and is implemented by appropriately parameterized Gabor functions. It is noted that the first two layers are passive filters that are not subject to any learning process. In the third layer, L_3, the learning occurs using the one-pass learning method described in Sect. 37.6.2. Here neural maps are created and merged according to the rules of the learning algorithm. Finally, the fourth layer L_4, consists of a single neuron for each output class, which accumulates opinions about the class label of a certain sequence of input frames. The weights between L_3 and L_4 are fixed to a constant value, usually 1, and are not subject to learning. The first L_4 neuron that is activated by the stimuli presented determines the classification result for the input. After the activation of an L_4 neuron the system stops.

Experimental evidence about the suitability of this pattern recognition system is provided in [37.95] along with a comparison to other typical classification methods.

Auditory Pattern Recognition

A similar network, but in an entirely different context, was investigated in [37.96], where a text-independent speaker authentication system is presented. The classification task in this work consisted of the correct labeling of audio streams presented to the system.

Speech signals are split into temporal frames, each containing a signal segment over a short time period. The frames are first pre-processed using the mel-frequency cepstral coefficients (MFCCs) [37.97] and then used to invoke the eSNN. The MFCC frame is transformed into the spike domain using rank order encoding [37.24] and the resulting stimulus is propagated to the first layer of neurons. This layer, denoted L_1, contains two neural ensembles representing the speaker and the background model, respectively. While the former model is trained on the voice of a certain speaker, the latter is trained on the background noise of the audio stream. This system also collects opinions about the

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

class label of the presented sequence of input frames, which is implemented by the second layer of the network. Layer L_2 consists of only two neurons, each of which accumulates information about whether a given frame corresponds to a certain speaker or to the background noise. Whenever an L_2 neuron is activated, the simulation of the network stops and the classification output is presented.

Audio-Visual Pattern Recognition

The two recognition systems presented above were successfully combined, forming an audio-visual pattern recognition method. Both systems are trained individually, but their output is propagated to an additional supra-modal layer. The supra-modal layer integrates incoming sensory information from individual modalities and cross-modal connections enable the influence of one modality upon the other. A detailed discussion of this system along with experimental evidence is given in [37.98, 99].

Case Study on Ecological Modeling

In [37.100, 101], the eSNN was applied on a real world data set in the context of an ecological modeling problem. For many invertebrate species little is known about their response to environmental variables over large spatial scales. That knowledge is important since it can help to identify critical locations in which a species that has the potential to cause great environmental harm might establish a new damaging population. The usual approach to determine the importance of a range of environmental variables that explain the global distribution of a species is to train or fit a model to its known distribution using environmental parameters measured in areas where the species is present and where it is absent.

Meteorological data that comprised 68 monthly and seasonal temperature, rainfall, and soil moisture variables for 206 global geographic sites were compiled from published records. These variables were correlated to global locations where the Mediterranean fruit-fly (*Ceratitis capitata*), a serious invasive species and fruit pest, was recorded at the time of the study, as either present or absent [37.102]. Motivated by inadequate results [37.103–105] used a different method, namely the multi-layer perceptron (MLP); this study aimed to identify important features relevant for predicting the presence/absence of this insect species. The results obtained may also be of importance to evaluate the risk of invasion of certain species into specific geographical regions.

Taste Recognition

The last application of eSNN being discussed here investigates the use of SNN for taste recognition in a gustatory model. The classification performance of eSNN was experimentally explored based on water and wine samples collected from [37.106, 107]. The topology of the model consists of two layers. The first layer receives an input stimulus obtained from the mapping of a real-valued data sample into spike trains using a rank order population encoding (Sect. 37.6.1) The weights from the first neural layer are subject to training according to the already discussed one-pass learning method. Finally, the output of the second neural layer determines the class label of the presented input stimulus.

The method was investigated in a number of scenarios, where the size of the data sets and the number of class labels was varied. Generally, eSNN reported promising results on both large and small data sets, which has motivated an FPGA hardware implementation of the system [37.108].

References

37.1 E.R. Kandel: *Principles of Neural Science* (McGraw-Hill, Columbus 2000)
37.2 W. Maass: Networks of spiking neurons: The third generation of neural network models, Neural Netw. **10**(9), 1659–1671 (1997)
37.3 A.L. Hodgkin, A.F. Huxley: A quantitative description of membrane current and its application to conduction and excitation in nerve, J. Physiol. **117**(4), 500–544 (1952)
37.4 M. Nelson, J. Rinzel: The Hodgkin-Huxley model. In: *The Book of Genesis*, ed. by J.M. Bower, D. Beeman (Springer, Berlin, Heidelberg 1995) pp. 27–51
37.5 J.M. Bower, D. Beeman: *The Book of Genesis* (Springer, Berlin, Heidelberg 1995)
37.6 L. Lapicque: Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation, J. Physiol. Pathol. Gen. **9**, 620–635 (1907)
37.7 L.F. Abbott: Lapicque's introduction of the integrate-and-fire model neuron (1907), Brain Res. Bull. **50**(5/6), 303–304 (1999)
37.8 N. Brunel, M.C.W. van Rossum: Lapicque's 1907 paper: From frogs to integrate-and-fire, Biol. Cybern. **97**(5), 337–339 (2007)

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

37.9 B.W. Knight: Dynamics of encoding in a population of neurons, J. Gen. Physiol. **59**, 734–766 (1972)

37.10 H. Meffin, A.N. Burkitt, D.B. Grayden: An analytical model for the "large, fluctuating synaptic conductance state" typical of neocortical neurons in vivo, J. Comput. Neurosci. **16**, 159–175 (2004)

37.11 W. Gerstner, W.M. Kistler: *Spiking Neuron Models: Single Neurons, Populations, Plasticity* (Cambridge Univ. Press, Cambridge 2002)

37.12 N. Burkitt: A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input, Biol. Cybern. **95**(1), 1–19 (2006)

37.13 N. Burkitt: A review of the integrate-and-fire neuron model: II. Inhomogeneous synaptic input and network properties, Biol. Cybern. **95**(2), 97–112 (2006)

37.14 E.M. Izhikevich: Simple model of spiking neurons, IEEE Trans. Neural Netw. **14**(6), 1569–1572 (2003)

37.15 E.M. Izhikevich: *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting* (MIT Press, Cambridge 2006)

37.16 E.M. Izhikevich: Which model to use for cortical spiking neurons?, IEEE Trans. Neural Netw. **15**(5), 1063–1070 (2004)

37.17 E.M. Izhikevich, G.M. Edelman: Large-scale model of mammalian thalamocortical systems, Proc. Natl. Acad. Sci. USA **105**(9), 3593–3598 (2008)

37.18 T. Masquelier, R. Guyonneau, S.J. Thorpe: Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains, PLoS ONE **3**, e1377 (2008)

37.19 W. Maass, C.M. Bishop (Eds.): *Pulsed Neural Networks* (MIT Press, Cambridge 1999)

37.20 W. Maass: Lower bounds for the computational power of networks of spiking neurons, Electron. Colloq. Comput. Complex. (ECCC) **1**(19), TS8 (1994)

37.21 W. Maass: Computing with spiking neurons. In: *Pulsed Neural Networks* (MIT Press, Cambridge 1999) pp. 55–85

37.22 W. Gerstner, J.L. van Hemmen, J.D. Cowan: What matters in neuronal locking?, Neural Comput. **8**(8), 1653–1676 (1996)

37.23 W.M. Kistler, R. Seitz, J.L. van Hemmen: Modeling collective excitations in cortical tissue, J. Phys. D **114**(3/4), 273–295 (1998)

37.24 S.J. Thorpe, J. Gautrais: *Rank Order Coding* (Plenum, New York 1998) pp. 113–118

37.25 S.J. Thorpe: Spike arrival times: A highly efficient coding scheme for neural networks. In: *Paralle Processing in Neural Systems and Computers*, ed. by R. Eckmiller, G. Hartmann, G. Hauske (Elsevier, Amsterdam 1990) pp. 91–94

37.26 S.J. Thorpe, D. Fize, C. Marlot: Speed of processing in the human visual system, Nature **381**, 520–522 (1996)

37.27 R. Van Rullen, S.J. Thorpe: Rate coding versus temporal order coding: What the retinal ganglion cells tell the visual cortex, Neural Comput. **13**(6), 1255–1283 (2001)

37.28 S.J. Thorpe, J. Gautrais: Rapid visual processing using spike asynchrony, Proc. Adv. Neural Inf. Process. Syst., Vol. 9 (NIPS), Denver 1996 (MIT Press, Cambridge 1996) pp. 901–907

37.29 S.J. Thorpe: How can the human visual system process a natural scene in under 150 ms? On the role of asynchronous spike propagation, ESANN 1997, 5th Eur. Symp. Artif. Neural Netw., D-Facto (1997)

37.30 A. Delorme, S.J. Thorpe: SpikeNET: An event-driven simulation package for modelling large networks of spiking neurons, Network **14**, 613–627 (2003)

37.31 S.J. Thorpe, R. Guyonneau, N. Guilbaud, J.-M. Allegraud, R. Van Rullen: SpikeNet: Real-time visual processing with one spike per neuron, Neurocomputing **58–60**, 857–864 (2004)

37.32 R. Van Rullen, J. Gautrais, A. Delorme, S. Thorpe: Face processing using one spike per neurone, Biosystems **48**(1–3), 229–239 (1998)

37.33 A. Delorme, L. Perrinet, S.J. Thorpe: Networks of integrate-and-fire neurons using rank order coding B: Spike timing dependent plasticity and emergence of orientation selectivity, Neurocomputing **38–40**, 539–545 (2001)

37.34 E.D. Adrian: The impulses produced by sensory nerve endings, J. Physiol. (London) **61**, 49–72 (1926)

37.35 F. Rieke, D. Warland, R.R. van Steveninck, W. Bialek: *Spikes: Exploring the Neural Code* (MIT Press, Cambridge 1999)

37.36 W. Gerstner: Population dynamics of spiking neurons: Fast transients, asynchronous states, and locking, Neural Comput. **12**(1), 43–89 (2000)

37.37 N. Brunel, F.S. Chance, N. Fourcaud, L.F. Abbott: Effects of synaptic noise and filtering on the frequency response of spiking neurons, Phys. Rev. Lett. **86**, 2186–2189 (2001)

37.38 R. Lestienne: Determination of the precision of spike timing in the visual cortex of anaesthetised cats, Biol. Cybern. **74**(1), 55–61 (1995)

37.39 M.P. Nawrot, P. Schnepel, A. Aertsen, C. Boucsein: Precisely timed signal transmission in neocortical networks with reliable intermediate-range projections, Front. Neural Circuits **3**(2), 1–11 (2009)

37.40 A.E.P. Villa, I.V. Tetko, B. Hyland, A. Najem: Spatiotemporal activity patterns of rat cortical neurons predict responses in a conditioned task, Proc. Natl. Acad. Sci. USA **96**(3), 1106–1111 (1999)

37.41 M.J. Tovee, E.T. Rolls, A. Treves, R.P. Bellis: Information encoding and the responses of single neurons in the primate temporal visual cortex, J. Neurophysiol. **70**(2), 640–654 (1993)

37.42 C. von der Malsburg: *The Correlation Theory of Brain Function*, Internal Report, Vol. 81-2 (Max-Planck-Institute for Biophysical Chemistry, Göttingen 1981)

37.43 R.V. Florian: A reinforcement learning algorithm for spiking neural networks, Proc. 7th Int. Symp.

TS8 *Please supply the page range.*

Editor's or typesetter's annotations (will be removed before the final TeX run)

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

Symb. Numer. Algorithms Sci. Comput. (SYNASC 2005) (IEEE, Los Alamitos 2005) pp. 299–306

37.44 R.V. Florian: Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity, Neural Comput. **19**(6), 1468–1502 (2007)

37.45 H.S. Seung: Learning in spiking neural networks by reinforcement of stochastic synaptic transmission, Neuron **40**(6), 1063–1073 (2003)

37.46 X. Xie, H.S. Seung: Learning in neural networks by reinforcement of irregular spiking, Phys. Rev. E **69**(4), 041909 (2004)

37.47 A.J. Kasinski, F. Ponulak: Comparison of supervised learning methods for spike time coding in spiking neural networks, Int. J. Appl. Math. Comput. Sci. **16**, 101–113 (2006)

37.48 D.O. Hebb (Ed.): *The Organization of Behavior* (Wiley, New York 1949)

37.49 T.V.P. Bliss, T. Lomo: Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path, J. Physiol. **232**(2), 331–356 (1973)

37.50 T.V.P. Bliss, A.R. Gardner-Medwin: Long-lasting potentiation of synaptic transmission in the dentate area of the unanaesthetized rabbit following stimulation of the perforant path, J. Physiol. **232**(2), 357–374 (1973)

37.51 C.C. Bell, V.Z. Han, Y. Sugawara, K. Grant: Synaptic plasticity in a cerebellum-like structure depends on temporal order, Nature **387**, 278–281 (1997)

37.52 H. Markram, J. Lubke, M. Frotscher, B. Sakmann: Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs, Science **275**(5297), 213–215 (1997)

37.53 G.-Q. Bi, M.M. Poo: Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type, J. Neurosci. **18**(24), 10464–10472 (1998)

37.54 V. Egger, D. Feldmeyer, B. Sakmann: Coincidence detection and changes of synaptic efficacy in spiny stellate neurons in rat barrel cortex, Nat. Neurosci. **2**, 1098–1105 (1999)

37.55 J. Iglesias, J. Eriksson, F. Grize, M. Tomassini, A.E.P. Villa: Dynamics of pruning in simulated large-scale spiking neural networks, Biosystems **79**(1–3), 11–20 (2005)

37.56 J. Iglesias, A.E.P. Villa: Neuronal cell death and synaptic pruning driven by spike-timing dependent plasticity. In: *Artificial Neural Networks ICANN 2006*, Lecture Notes in Computer Science, Vol. 4132, ed. by TS[9] (Springer, Berlin, Heidelberg 2006) pp. 953–962

37.57 J. Iglesias, A.E.P. Villa: Effect of stimulus-driven pruning on the detection of spatiotemporal patterns of activity in large neural networks, Biosystems **89**(1–3), 287–293 (2007)

37.58 G.-Q. Bi, M.-M. Poo: Synaptic modification by correlated activity: Hebb's postulate revisited, Annu. Rev. Neurosci. **24**(1), 139–166 (2001)

37.59 R. Kempter, W. Gerstner, J.L. van Hemmen: Hebbian learning and spiking neurons, Phys. Rev. E **59**(4), 4498–4514 (1999)

37.60 W. Gerstner, W.K. Kistler: Mathematical formulations of Hebbian learning, Biol. Cybern. **87**(5/6), 404–415 (2002)

37.61 W.M. Kistler: Spike-timing dependent synaptic plasticity: A phenomenological framework, Biol. Cybern. **87**(5/6), 416–427 (2002)

37.62 S.M. Bohte, J.N. Kok, J.A. La Poutré: SpikeProp: Backpropagation for networks of spiking neurons, ESANN Proc. (2000) pp. 419–424

37.63 S.M. Bohte, J.N. Kok, J.A. La Poutré: Error-backpropagation in temporally encoded networks of spiking neurons, Neurocomputing **48**(1–4), 17–37 (2002)

37.64 J. Xin, M.J. Embrechts: *Supervised Learning with Spiking Neural Networks* (IEEE, Bellingham 2001) pp. 1772–1777

37.65 B. Schrauwen, J. van Campenhout: Improving SpikeProp: Enhancements to an error-backpropagation rule for spiking neural networks, Proc. 15th ProRISC Workshop (2004) pp. 104–174

37.66 P. Tiňo, A.J.S. Mills: Learning beyond finite memory in recurrent networks of spiking neurons, Neural Comput. **18**(3), 591–613 (2006)

37.67 T. Natschläger, W. Maass, H. Markram: The "liquid computer": A novel strategy for real-time computing on time series. In: *Special Issue on Foundations of Information Processing of Telematik*, Vol. 8 (2002), pp. 39–43 TS[10]

37.68 F. Ponulak: *ReSuMe – New supervised learning method for Spiking Neural Networks*, Tech. Rep. (Institute of Control and Information Engineering, Poznan University of Technology 2005)

37.69 W. Maass, T. Natschläger, H. Markram: Real-time computing without stable states: A new framework for neural computation based on perturbations, Neural Comput. **14**(11), 2531–2560 (2002)

37.70 D. Verstraeten, B. Schrauwen, M. D'Haene, D. Stroobandt: An experimental unification of reservoir computing methods, Neural Netw. **20**(3), 391–403 (2007)

37.71 T. Natschläger, H. Markram, W. Maass: Computer models and analysis tools for neural microcircuits. In: *Neuroscience Databases: A Practical Guide*, ed. by TS[9] (Kluwer Academic, Dordrecht 2003) pp. 123–138

37.72 D. Verstraeten, B. Schrauwen, D. Stroobandt: Isolated word recognition using a Liquid State Machine, Proc. ESANN (2005) pp. 435–440

37.73 A.J. Kasinski, F. Ponulak: TS[18]. In: *Experimental demonstration of learning properties of a new supervised learning method for the spiking neural networks*, Lecture Notes in Computer Science, Vol. 3696 (Springer, Berlin, Heidelberg 2005) pp. 145–152

TS[9] *Please supply the editor(s).*

TS[10] Please check this reference. Is this a journal article? Please provide all missing information (i.e. journal name).

TS[18] *Please supply the chapter/article/book title.*

Editor's or typesetter's annotations (will be removed before the final TeX run)

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

37.74 F. Ponulak, A.J. Kasinski: Generalization properties of spiking neurons trained with ReSuMe method, Proc. ESANN (2006) pp. 629–634

37.75 F. Ponulak: Analysis of the ReSuMe learning process for spiking neural networks, Appl. Math. Comput. Sci. **18**(2), 117–127 (2008)

37.76 N.T. Carnevale, M.L. Hines: *The NEURON Book* (Cambridge Univ. Press, University 2006)

37.77 NEURO, Yale University, Newhaven, available online at http://www.neuron.yale.edu/neuron TS13

37.78 GENESIS, General Neural Simulation System, available online at http://genesis-sim.org, TS13

37.79 B.P. Glackin, T.M. McGinnity, L.P. Maguire, Q. Wu, A. Belatreche: A novel approach for the implementation of large scale spiking neural networks on FPGA Hardware, Proc. IWANN, Vol. 3512 (Springer 2005) pp. 552–563

37.80 L.P. Maguire, T.M. McGinnity, B. Glackin, A. Ghani, A. Belatreche, J. Harkin: Challenges for large-scale implementations of spiking neural networks on FPGAs, Neurocomputing **71**(1–3), 13–29 (2007)

37.81 A. Knoblauch: Neural associative memory for brain modeling and information retrieval, Inf. Process. Lett. **95**(6), 537–544 (2005)

37.82 N. Iannella, L. Kindermann: Finding iterative roots with a spiking neural network, Inf. Process. Lett. **95**(6), 545–551 (2005)

37.83 D. Floreano, C. Mattiussi: TS18. In: *Evolution of spiking neural controllers for autonomous vision-based robots*, Lecture Notes in Computer Science, Vol. 2217, ed. by TS9 (Springer, Berlin, Heidelberg 2001) pp. 38–61,

37.84 D. Floreano, Y. Epars, J.-C. Zufferey, C. Mattiussi: Evolution of spiking neural circuits in autonomous mobile robots: Research articles, Int. J. Intell. Syst. **21**(9), 1005–1024 (2006)

37.85 X. Wang, Z.-G. Hou, A. Zou, M. Tan, L. Cheng: A behavior controller based on spiking neural networks for mobile robots, Neurocomputing **71**(4–6), 655–666 (2008)

37.86 N. Kasabov: *Evolving Connectionist Systems: The Knowledge Engineering Approach* (Springer, New York 2006)

37.87 S.G. Wysoski, L. Benuskova, N.K. Kasabov: Adaptive Learning Procedure for a Network of Spiking Neurons and Visual Pattern Recognition. In: *ACIVS'06 Proc. 8th Int. Conf. Adv. Concepts for Intelligent Vision Systems*, Lecture Notes in Computer Science, Vol. 4179, ed. by TS9 (Springer, Berlin 2006) pp. 1133–1142,

37.88 S. Soltic, S. Wysoski, N. Kasabov: Evolving spiking neural networks for taste recognition, Proc. IJCNN 2008 (IEEE, Bellingham 2008) pp. 2091–2097

37.89 L. Perrinet, A. Delorme, M. Samuelides, S.J. Thorpe: Networks of integrate-and-fire neuron using rank order coding A: How to implement spike time dependent Hebbian plasticity, Neurocomputing **38–40**, 817–822 (2001)

37.90 S.J. Thorpe, A. Delorme, R. van Rullen: Spike-based strategies for rapid processing, Neural Netw. **14**(6/7), 715–725 (2001)

37.91 S. Loiselle, J. Rouat, D. Pressnitzer, S. Thorpe: Exploration of rank order coding with spiking neural networks for speech recognition, Proc. IJCNN 2005 (IEEE, Bellingham 2005) pp. 2076–2080

37.92 A. Delorme, J. Gautrais, R. Van Rullen, S. Thorpe: SpikeNET: A simulator for modeling large networks of integrate and fire neurons, Neurocomputing **26–27**, 989–996 (1999)

37.93 A. Delorme, S.J. Thorpe: Face identification using one spike per neuron: Resistance to image degradations, Neural Netw. **14**(6/7), 795–803 (2001)

37.94 S.G. Wysoski, L. Benuskova, N. Kasabov: On-line learning with structural adaptation in a network of spiking neurons for visual pattern recognition, Proc. ICANN 2006, Part I, Vol. 4131 (Springer, Berlin, Heidelberg 2006) pp. 61–70

37.95 S.G. Wysoski, L. Benuskova, N. Kasabov: Fast and adaptive network of spiking neurons for multi-view visual pattern recognition, Neurocomputing **71**(13–15), 2563–2575 (2008)

37.96 S.G. Wysoski, L. Benuskova, N. Kasabov: TS18. In: *Text-independent speaker authentication with spiking neural networks*, Lecture Notes in Computer Science, Vol. 4669, ed. by TS9 (Springer, Berlin, Heidelberg 2007) pp. 758–767,

37.97 L. Rabiner, B.-H. Juang: *Fundamentals of Speech Recognition* (Prentice-Hall, Upper Saddle River 1993)

37.98 S.G. Wysoski, L. Benuskova, N. Kasabov: TS18. In: *Adaptive Spiking Neural Networks for Audiovisual Pattern Recognition*, Lecture Notes in Computer Science, Vol. 4985, ed. by TS9 (Springer, Berlin, Heidelberg 2008) pp. 406–415,

37.99 S.G. Wysoski: Evolving Spiking Neural Networks for Adaptive Audiovisual Pattern Recognition. Ph.D. Thesis (Auckland University, Auckland 2008)

37.100 S. Schliebs, M. Defoin-Platel, S. Worner, N. Kasabov: Quantum-inspired feature and parameter optimisation of evolving spiking neural networks with a case study from ecological modeling, Proc. IJCNN (IEEE Computer Society, Washington 2009) pp. 2833–2840

37.101 S. Schliebs, M. Defoin-Platel, S. Worner, N. Kasabov: Integrated feature and parameter optimization for an evolving spiking neural network: Exploring heterogeneous probabilistic models, Neural Netw. **22**(5–6), 623–632 (2009)

37.102 CABI: *Crop Protection Compendium. Global Module*, 5th edn. (CABI, TS17 2003),

37.103 S. Worner, G. Lankin, S. Samarasinghe, D. Teulon: Improving prediction of aphid flights by temporal analysis of input data for an artificial neural network, N. Z. Plant Prot. **55**, 312–316 (2002)

37.104 N. Cocu, R. Harrington, M.D. Rounsevell, S.P. Worner, M. Hulle: Geographical location, climate and land

TS13 *Please supply more information.*
TS13 *Please supply more information.*
TS18 *Please supply the chapter/article/book title.*
TS18 *Please supply the chapter/article/book title.*
TS18 *Please supply the chapter/article/book title.*
TS17 *Please supply the publisher's location.*

SPIN: 12742608 (Springer Handbo dbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:0

use influences on the phenology and numbers of the aphid, *Myzus persicae*, in Europe, J. Biogeogr. **32**(4), 615–632 (2005)

37.105 M.J. Watts, S.P. Worner: Using MLP to determine abiotic factors inuencing the establishment of insect pest species, Proc. 2006 Int. Joint Conf. Neural Netw. (IJCNN 2006) (IEEE, New York 2006) pp. 3506–3511

37.106 H.C. de Sousa, A. Riul Jr.: Using MLP networks to classify red wines and water readings of an electronic tongue, VII Braz. Symp. Neural Netw. (SBRN'02) (2002)

37.107 A. Riul, H.C. de Sousa, R.R. Malmegrim, D.S. dos Santos, A.C.P.L.F. Carvalho, F.J. Fonseca, O.N. Oliveira, L.H.C. Mattoso: Wine classification by taste sensors made from ultra-thin films and using neural networks, Sens. Actuators B **98**(1), 77–82 (2004)

37.108 A. Zuppicich, S. Soltic: FPGA Implementation of an evolving spiking neural network. In: TS18, ed. by H.M. Köppen, N.K. Kasabov, G.G. Coghill (Springer, Berlin, Heidelberg 2009) pp. 1129–1136

SPIN: 12742608 (Springer Handbook of Springer Handbook of Nanomaterials)
MS ID: hb23-037 Proof 1
Created on: 30 January 2013 15:05 CET

TS18 *Please supply the chapter/article/book title.*

Editor's or typesetter's annotations (will be removed before the final TeX run)