Spiking Neural Networks

Eugen Rusakov Institute of Computer Engineering,
University of Heidelberg (ZITI)
Germany, 68131 Mannheim,
Email: rusakov@stud.uni-heidelberg.de

***Abstract*—This paper presents the spiking neural networks as a third generation model of neural networks. This model represents a more realistic neural network behavior extending the old models by further timing behaviour. With the idea not to fire at each timestep, but rather fire only when a threshold value is reached. By adding an activation level within the neurons, the spiking neural networks has the opportunity to increase or decrease the potentials in accordance with the signals passing this neurons. With the approach of a more realistic model, spiking neural networks improves especially the learning behavior of neural network applications. This paper gives also a survey about simulators working with spiking neural networks.**

***Index Terms*—Artificial Intelligence, Spiking Neural Networks, spike coding, simulator, learn behavior, neuron, synapses.**

I. Introduction

ONE of the greatest challenges facing 21st century science, is to understand how the human brain works. Especially the fascinating behaviour in learning and teaching itself to improve its performance and abilities, and creating solution for complex problems, are big challenges these days. Inspired by the functionality and intelligence of the human brain, the research area "Artificial Intelligence" [1] try to adapt and inherit this behaviour into the computer systems.

Artificial Intelligence brings along huge advantages. By emulating the human brain, systems with artificial intelligence have the potential to imitate human behaviour, which means in turn that this kind of systems have the possibilities to take over every work achieved by humans these days. Artificial Intelligence can find a way in every kind of work, which is done by humans, for support and unloading. The technical production is one of these areas, today many computers and robots handles and manage production work faster and with more efficiency like production pipelines within the automotive industry. But not only physical work can be handled by this systems, artificial intelligence computer capable to support doctors with diagnoses by processing more patients documents and continuously upgrading there knowledge about the medicine. One further step is to develop computer systems helping humans with research. The idea are computer systems testing models or concepts made by human for correctness, helping them to research and achieve knowledge faster.

One other reason why this field is so attractive, are the research opportunities on human brain diseases. Emulating the human brain, means to understand how it works and creating new simulation models to investigate the development of diseases like Depression or Alzheimer.

The artificial neural networks are one of the highest potentials with relation to autonomous learning and building thus a big foundation for autonomous thinking.

A. *The idea of Artificial Neural Networks*

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the human brain, process different information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems. Artificial Neural Networks are like people, they learn by example, sometimes with the trail-and-error strategy. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurones.

Following an initial period of enthusiasm, the field survived a period of frustration and disrepute. As the Artificial Neural Networks are mathematical models, the early researcher had no opportunity to proceed in this field due to massive constrained performance of computer systems in the 1950s. Except for a few simulations on computer systems, the Artificial Neural Networks had no chance to exploit their strengths with such unperformed computational power, based on their operating principle of massively parallel realization. Currently, the neural network field enjoys a resurgence of interest and a corresponding increase in funding. Today the researchers are capable to investigate many of the neural network models due to a continuously increasing performance of computer systems with stupendous result in this field. As the research area of Artificial Neural Networks is interdisciplinary, even the neuro science, biology and medicine conceive great expectations of neural networks simulations. [2]

B. *First Generation*

Based on their knowledge of the neurology of neural networks, the first models of neural networks was developed by Warren McCulloch and Walter Pitts in 1943 [3]. This kind of neural networks could practically compute every logical and even arithmetical function. Their networks were based on simple neurons which were considered to be binary devices with fixed thresholds. The results of their model were simple logic functions such as "a or b" and "a and b". The neurons are only capable to generate binary values on their outputs. To decide whether a output generates a ”1” or a ”0”, the neurons contains a threshold value. If the sum of all incoming signals on the inputs exceed the threshold, the neuron generates a ”1”, otherwise a ”0” is generated. This kind of neural networks model is known as the first generation of artificial neural networks. [2]

C. Second Generation

In 1949 the psychologies Donald O.Hebb established a hypothesis [4] that learning is based on a activating or a inhibiting activity of a synapses. The neural networks needed a spike-timed operating principle.

The so-called Perceptron-Model was introduced by Frank Rosenblatt in 1958 [5]. This approach extended the McCulloch-Pitts-Model by a continuous activation function within the neurons. Instead of a simple step function, the new model incorporate a sigmoid or hyperbolic tangent functions. Furthermore new topologies was introduction within the second generation by using more than one neuron layer. Typical examples of neural networks consisting of neurons of these types are feed-forward and recurrent neural networks. These are more powerful than their first generation predecessors: when equipped with a threshold function at the output layer of the network they are universal for digital computations, and do so with fewer neurons than a network of the first generation. In addition they can approximate any analog function arbitrarily well, making these networks universal for analog computations. [6]

D. Third Generation

The Spiking Neural Networks (SNNs) fall into the third generation of neural network models, increasing the level of realism in a neural simulation. In addition to neuronal and synaptic state, spiking neural networks also incorporate the concept of time into their operating model. The idea is that neurons in the SNNs do not fire at each propagation cycle (as it happens with typical multi-layer perceptron networks), but rather fire only when a membrane electrical charge reaches a specific value [7]. In context of spiking neural networks, the current activation level is normally considered to be the neuron’s state, with incoming spikes pushing this value higher, and then either firing or decaying over time.[7] The new approach within spiking neural networks consists of interpreting the outgoing spike-train as a real-value number, either rely on the frequency of spikes (rate coding), or the timing between spikes (temporal coding), to encode information. Unfortunately the spiking neural networks are nowadays in a evaluative and investigative phase, for that reason no concrete applications were found with spiking neural networks.

II. Functionality of Artificial Neural Networks

Inspired by the biology of the human brain, artificial neural networks (ANNs)[1] try to adapt the functionality of the brain. Nevertheless ANNs are a model and abstraction of information processing. This models consists of nodes, representing neurons as a computation unit, to interpret the incoming signals and compute their output signals. The nodes are ordered as a interconnected network with many connections between the nodes. This connections representing the axons from the biological view and the inputs of the nodes representing the synapses. Figure 1 shows the transmission of signals.

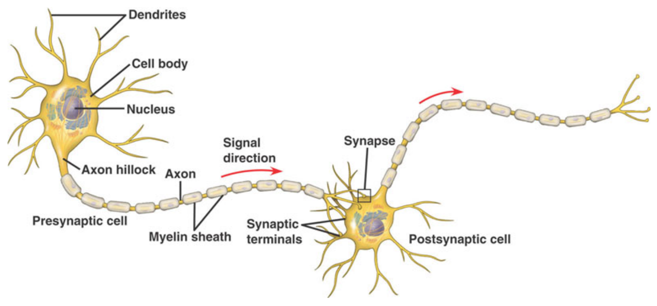

Fig. 1. Biological interconnection between neurones [8]

The nucleus surrounded by the cell body are the neurones generating signals, which are transmitted via the axons. After the signal has reach the next neuron, the synapses receive this signal an forward it to the neurones. Analog to this biological model, the neural networks has the same structure as shown in Figure 2. The nodes are connected among themselves, ordered

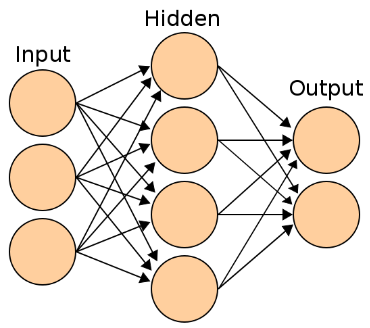

Fig. 2. Structure of artificial neural network [1]

in layers. The three nodes on the left side, symbolize the input layer as an access point to this network. The two nodes on the right side, are the output nodes for signal generation to the next network segment. The four nodes in the middle, are the hidden layer and characterizing the complexity of a network. The connections between the nodes are equivalent to the synapses in the biological systems. This edges between the nodes possess weights, which get multiplied with the signals passing through. Although Figure 1 shows axons as the connection edge between neurons, the axons has no influence on the signals. Only synapses perform changes on signals by increasing or decreasing their value.

The neurons representing the computational unit of neural networks [2]. They consist of inputs, which carry the signals multiplied by the corresponding weights. The neuron sums up all input values and compares the new value with the activation function. The activation function is a kind of border and make a decision whether the value on the output increase, decrease or gets completely suppress. This activation function has different characters as shown in figure 4. As the first generation of neural networks works with binary values, the step-function was applied. From the second generation a more natural way was chosen by using sigmoid-functions. This approach suits the dynamic natural behaviour of biological neurons, but also poses a problem in modelling a sigmoid-function within the signals. To overcome this problem the signals changed from binary values to pulses, so the sigmoid function can be created by rate coding (pulses per second) [15].

A. Techniques

The field of artificial intelligence pursue different approaches and created four methods to separate this approaches in groups. Beside the symbolic artificial intelligence, which pursue a top-down approach converging to the intelligence performance from the conceptual level and the phenomenological artificial intelligence, where only results matters without scrutinizing the processing procedures, the neural networks belongs to the neural artificial intelligence, pursuing a bottom-up approach by learning the brain behaviour in detail. Neural Networks also belong to the simulation methods group, which orientate towards cognitive processes of human brain. [9]

For a concrete realisation of artificial intelligence there are five areas of techniques.

Searching: Artificial intelligence often concerned with searching defined solution for given problems. Path-finding is a example of searching problems, which gets negotiated with search-algorithms.

Planing: The goal of the planing is to create action sequences, which have to be followed by agent systems to achieve their goals, whereby the planing consists of two phases. The first is the goal formulation, where goals get defined based on the actual state. The second phase is to formulate the problem, after the goal is known. In this phase the actions and states should be considered.

Optimization: Sometimes tasks cannot be solved by analysing methods, therefore different optimization technique exists. Some problems cannot be handled exactly due to unknown parameter within the system, to negotiate such problems mathematical programming is used, for example to find a local optimum.

Logical deduction: This technique primarily concerned with knowledge presentations. The idea is to create systems, which are capable to support humans for example in mathematical proofs.

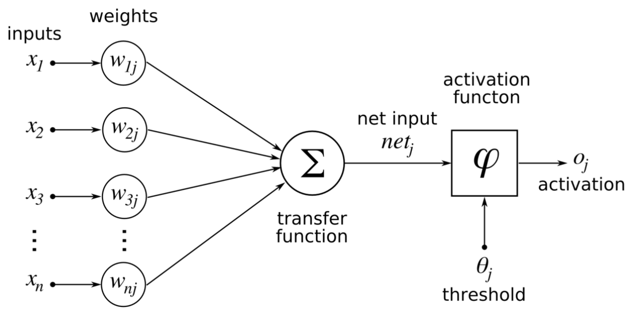

Fig. 3. Structure of artificial neurons [9]

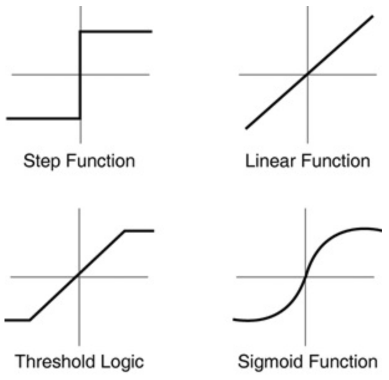

Fig. 4. Activation functions with different characters [10]

Approximation: In many applications there are approximation problems, where general rules are deduced from a volume of data. So neural networks was suggested as a possible solution way, but at these days mathematical approaches are used to solve the most problems within this field.

B. Topologies

From the beginning with few neurons within the neural network, more complex topologies was introduced to negotiate different logical and behavioural problems. Whereas the first neural networks consisted of single layer topologies, where the input neurons also represented the output neurons, the newer generations of neural networks extended their topologies to multi layer or even recurrent layer [9]. As shown in figure

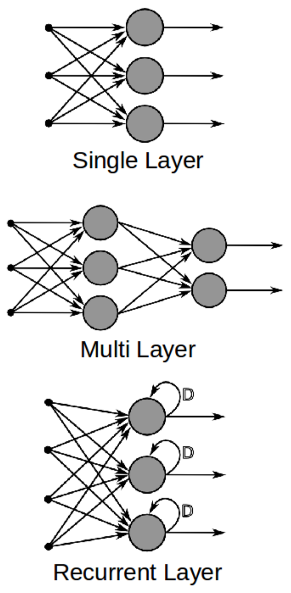

Fig. 5. Three types of topologies within the neural networks [9]

5, there are three types of topologies. Single layer topologies are the most simplest but also bring along several disadvantages. The simple structure enables only a small spectrum of behaviour thus their are not able to solve very complex problems. To overcome such disadvantages more layer are required, as shown in figure 6. To solve the XOR-Problem [1], a multi layer structure was introduced within the perceptrons [5], because single layer perceptrons was not able to handle the XOR-Problem. Furthermore the recurrent topology was introduced. This kind of topologies can contain feedback-loops by directing connections from the output to the input. Feedback-loops enables a dynamical behaviour and equips the neural networks with a memory. Generally the structure of a topology depends on the utilized learn procedure. For example the least-mean-square algorithm works only for single layer, whereas the back-propagation algorithms can work on multi layer neural networks. The structure does not necessarily have to be homogeneous, there are also approaches combining different topologies to bring out several advantages.

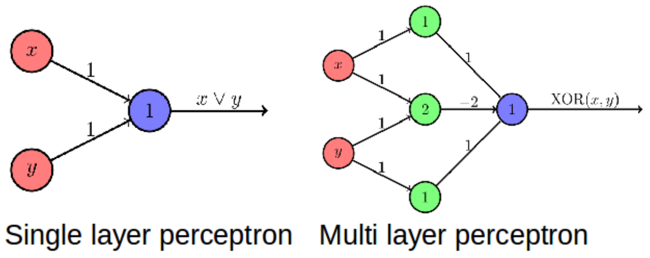

Fig. 6. Single and multi layer perceptron, extended to solve the XOR-Problem [5]

C. Learning Behaviour

One of the major strength of neural networks is the mostly autonomous learning behaviour. To achieve such high developed behaviour the neural networks have to modify itself based on learning rules. Overall there are seven rules as following explained:

- Develop new connections
- Delete existing connections
- Modify weights of connections
- Modify threshold values of neurons
- Modify activation-, propagation or output functions
- Initiate new neurons
- Eliminate existing neurons

The third method of modifying weights of connection is the mostly used learning rule [11]. For the concrete implementation, three approaches are often used within learning behavior. The first is the hebbian theory [12], which is the oldest and simples of the learning rules. It describe the changes of weights between two neurons which are both active, either to increase or decrease. The second rule is called the least-mean-square algorithm. By monitoring the actual output and compare it with the requested output, this algorithm tries to compute the new weights between neurons. The third one is the back-propagation [13], consisting of three phases. At first the neural networks gets an input pattern, which get propagates forward through the neurons. Than the actual output is getting compared with the requested output values and the difference between them represents the error of the network. This error is now propagating backward the network, where the weights get change according to the error.

Furthermore three types of learning behavior was introduced, so the learning algorithms could be assigned to this types.

supervised: Within this type of learning behavior the input and also the corresponding output is given. The neural network is learning by comparing the actual output value with the desired value.

unsupervised: Only input values are given without evaluating the output values. The network try to classify the input autonomously. One famous variant is the learning vector quantization, which was introduced by Teuvo Kohonen [14].

reinforcement: Learning by reinforcement do not require given input or output values, instead this type of learning behavior finds the right output values on its own. The network only need to know if it generates the right output values. The disadvantage is the time factor, because learning without known output values takes obviously more time.

III. Spiking Neural Networks

Spiking neural Networks (SNNs) are referred as the third generation of neural networks. Inspired by the natural computing of the human brain and recent advances in neurosciences, the SNNs represent a more accurate modelling of neuron computing and synaptic interactions between them by incorporating the concept of time [7]. The approach with spiking neurons achieve more computational power within neural networks as the previous generations consisting of thresholds or sigmoid units. Spiking neural networks brings along several benefits such as higher speed or, as the computer scientists would say, more bandwidth. The model of spiking neurons is capable of transmitting and receiving considerably more information within only a few spikes, which allows to develop faster and more efficient implementations. The next benefit is the possibility of real-time actions due to pulse coding and the already mentioned time concept [16]. Spiking neural networks can perform synchronous and also asynchronous, that implies better methods for pattern recognition for example a synchronous timing for neurons recognizing the main pattern and a asynchronous timing for neurons recognizing the background. Spiking neurons also decrease the complexity of neural networks, due to their computational strength their are capable to compute any function a second generation network can but with fewer neurons [11].

Spike Coding vs Rate Coding

The neural networks of the third generation was expanded by spike coding due to recent advances in neuro science [15]. The human brain analyse and classify different pattern within the visual input in approximately 100ms. Furthermore the brain needs 10 synaptic steps from the retina to the temporal lobe, which leaves about 10ms of processing-time per neuron. Rate coding, like it is used within the older generations of neural networks, is not capable to process these amount of

[1]A single layer perceptron network, as shown in figure 6 on the left side, is not capable to realize a XOR circuit, this disadvantage is called the XOR-Problem. To solve this problem the perceptron network must be extended by a further layer as shown in figure 6 on the right side.

data. To overtime such problems spike coding can be used due to its functionality not only to count the spikes per second but rather coding informations within the timing between spike (temporal coding).

Hodgkin-Huxley model

The pioneer of the spiking neurons are the conductance-based neuron models as one of the famous and well-known Hodgkin-Huxley electrical model [16]. This model simulate neurons close by biological circumstances, membrane potentials and dynamics of spike firing are produced by this model as a realistic variation. This strong realistic model, implies that it is rarely used within simulation due to the complicated computation.

Integrate-and-Fire model

A less complicated approach, which better suits the computation behaviour of simulations, is the integrate and fire model (I&F) [11] with several variants. The Integrate-and-Fire neuron only models explicit a passive leakage current through the membrane and the generation of action potentials is replaced by a threshold mechanism. As the name suggests, the model sums up all synaptic input currents (integrate) and fires a action potential when the threshold value is reached (fire). Due to the differential equations which now can be solves explicit, this model is used within mathematical analyses of brain functionalities and networks simulations on computers.

IV. Simulators

Additional to the research work within spiking neural networks several simulators were developed to produce and investigate concrete applications with this networks. In this chapter a few simulators are presented to give a small overview of the possibilities nowadays. With the achieved performance of computer systems, simulators and their applications experienced a boost of capabilities and shooting up like mushrooms in the last years. Besides the simulators work on CPU architectures, scaling their applications on multi-processors or even cluster systems, the GPU architectures become more and more interesting for simulating neural networks. With the massive parallel computations, GPU architectures suits the brain functionality.

Brian Simulator

An intuitive and highly flexible simulator for rapidly developing new models is the Brian Simulator [17]. The Brian Simulator is written in the programming language Phyton, due to the programming language scientific libraries can be used for defining models and analysing data. Despite the overhead of interpreted languages, efficient simulations can be developed using vectorisation techniques. Alternatively to Matlab or C, Brian is especially valuable for working on non-standard neuron models, which are not easily covered by existing software. Furthermore Brian is interesting due to the easy and intuitive syntax, make this simulator attractive for teaching computational neuroscience. As Brian is written in Phyton, the simulator tries to overcome the lower performance of interpreter languages by using vectorisation. For large networks the overhead ratio is not significant high as well as a large computation time within one neuron. But small networks have to carry a bigger overhead ratio, nevertheless favourable scenarios for Brian is the simulation of a small network for a long biological time.

NEural Simulation Tool - NEST

The NEST Simulator [18][19] is build to simulate large, structured neuronal systems, designed in the context of their anatomical, morphological and electrophysiological properties. This Simulator is optimized for large networks of spiking neurons incorporates the representation of spikes in continuous time. NEST is written in C++ for an object-oriented style and as a part of a simulation language interpreter (SLI) various modules are combined by a module loader. Furthermore the developer try to keep the simulator platform independent by using GNU developer tools [19]. Basically the NEST Simulator consists of three main components [18]:

- **Nodes** The neurons, devices and sub-networks are handled as nodes for easier mapping on computer systems. The dynamic states within the nodes can be influenced by incoming events.
- **Events** Events representing parts of information as the most common event is the spike-event. Besides the spike-event, voltage- and current events are other types.
- **Connections** To establish communication between nodes, connections are integrated as channels which exchange events. The connections possess weights, information directions and they work mostly with one event type.

Spiking Neural Networks on GPUs

Due to the computation functionality of neural networks, several approaches are investigated on graphical processing units (GPUs). With the processing model of massive parallel computations, GPUs fits the massive parallel structure of neural networks. Nevertheless GPUs have a tremendous disadvantage due to its small memory capacities. Mapping large networks with many neurons and even more connections between them, needs to swap data from GPUs to main memory, resulting in a bandwidth bottle neck due to a relative slow PCI interface. Furthermore if the networks are big enough and the main memory has not enough space, simulators for CPU versions can use cluster systems, mapping the networks on several devices. Although GPUs are capable to swap data to main memory, communicating between nodes resulting in huge bandwidth penalties and the GPUs lose their massive parallel computation advantage. However there are measured results which present significant speedups in context of small networks executed on one device. One experiment [20] shows that GPUs can achieve a 22 times higher throughput (spikes arrivals per second). The CPU simulator was using a Xeon E5420 with 4 cores achieved 25M throughput and the GPU simulator was a Tesla C1060 with 30 cores achieved 510M throughput. The size of the networks was 30K neurons with 1000 synapses per neuron. All in all GPUs are high potentials in context of neural networks especially as GPU developer are working on reducing or even eliminating the memory bandwidth bottlenecks.

V. Conclusion

Spiking neural networks are high potentials in context of emulating the human brain. With the most realistic model of biological neurons nowadays, even neuroscientist and biologists have big expectations due to investigation of brain diseases. This neuron models initialize new opportunities to develop better learning behaviour resulting in more intelligent systems. With the continuously increasing performance of computer systems spiking neural networks can be simulated on a grand scale, building large networks with a more complex structure achieving higher artificial intelligence. Despite the fact that spiking neural networks are a stupendous realistic model of natural neural networks, they are transient due to major advances of neuroscience. In the future there will be a newer generation of neural networks models, with a more realistic behaviour representing the human brain in a more detailed way.

References

[1] "Artificial neural network," Jan. 2015, page Version ID: 641559560. [Online]. Available: http://en.wikipedia.org/w/index.php?title=Artificial_neural_network&oldid=641559560

[2] D. S. Christos Stergiou, "Neural networks." [Online]. Available: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html#Introduction%20to%20neural%20networks

[3] "McCulloch-pitts-zelle," Mar. 2014, page Version ID: 128969223. [Online]. Available: http://de.wikipedia.org/w/index.php?title=McCulloch-Pitts-Zelle&oldid=128969223

[4] "Hebbsche lernregel," Dec. 2014, page Version ID: 136471908. [Online]. Available: http://de.wikipedia.org/w/index.php?title=Hebbsche_Lernregel&oldid=136471908

[5] "Perzeptron," Oct. 2014, page Version ID: 135068161. [Online]. Available: http://de.wikipedia.org/w/index.php?title=Perzeptron&oldid=135068161

[6] J. Vreeken, "Spiking neural networks, an introduction," *Institute for Information and Computing Sciences, Utrecht University*, 2002.

[7] "Spiking neural network," Dec. 2014, page Version ID: 632401924. [Online]. Available: http://en.wikipedia.org/w/index.php?title=Spiking_neural_network&oldid=632401924

[8] "The human brain and neural communication | main | thoughtblox." [Online]. Available: https://thoughtblox.com/bloc/207/the-human-brain-and-neural-communication/public/?VTJPN8

[9] "Knstliches neuronales netz," Jan. 2015, page Version ID: 137273138. [Online]. Available: http://de.wikipedia.org/w/index.php?title=K%C3%BCnstliches_neuronales_Netz&oldid=137273138

[10] "Game development." [Online]. Available: http://www.yaldex.com/game-development/1592730043_ch20lev1sec2.html

[11] A. Schnorr, "Simulation neuronaler netze," *Hochschule Aachen, Medizintechnik und Technomathematik*, pp. 15–18, Dec. 2010.

[12] "Hebbian theory," Dec. 2014, page Version ID: 627455554. [Online]. Available: http://en.wikipedia.org/w/index.php?title=Hebbian_theory&oldid=627455554

[13] "Backpropagation," Jan. 2015, page Version ID: 637185486. [Online]. Available: http://en.wikipedia.org/w/index.php?title=Backpropagation&oldid=637185486

[14] "Selbstorganisierende karte," Jun. 2014, page Version ID: 126422307. [Online]. Available: http://de.wikipedia.org/w/index.php?title=Selbstorganisierende_Karte&oldid=126422307

[15] S. Thorpe, A. Delorme, and R. Van Rullen, "Spike-based strategies for rapid processing," *Neural Networks: The Official Journal of the International Neural Network Society*, vol. 14, no. 6-7, pp. 715–725, Sep. 2001.

[16] S. B. Helene Paugam-Moisy, "Computing with spiking neuron networks," *Universit de Lyon Laboratoire de Rechereche en Informatique & CWI Amsterdam*, pp. 1,5,7,11,12, 2012.

[17] R. B. Dan Goodman, "Brian: A simulator for spiking neural networks in python," *Front Neuroinformatics*, pp. 1,8,9, Oct. 2008.

[18] A. M. Marc-Oliver Gewaltig and H. E. Plesser, "NEST by example: an introduction to the neural simulation tool NEST," *Springer*, 2012.

[19] M.-O. G. Markus Diesmann, "NEST: An environment for neural systems simulations," *Plesser, T. & Macho, V. (ed.) Forschung und wisschenschaftliches Rechnen*, 2002.

[20] M. P. S. Andreas K. Fidjeland, "Accelerated simulation of spiking neural networks using GPUs," *WCCI 2010 IEEE World Congress of Computational Intelligence*.