Does Soft Computing Classify Research in Spiking Neural Networks?

Liam Maguire

Intelligent Systems Research Centre
School of Computing and Intelligent Systems
University of Ulster, Derry, Northern Ireland, BT48 7JL, UK
lp.maguire@ulster.ac.uk

Received: 17-11-2009
Accepted: 28-05-2010

Abstract

The last fifty years has witnessed considerable activity in research that develops computational approaches inspired by nature. There are a number of umbrella terms used by researchers to classify their contributions. This can cause problems in disseminating and sharing results and potentially restricts research due to a lack of knowledge of the varied contributions. This paper reviews research in spiking neural networks and attempts to determine if the term Soft Computing can be used to classify contributions in this area.

Keywords: Spiking neural networks, classification.

1. Introduction

For thousands of years man has taken inspiration from his environment to solve everyday problems. Initial examples include the ability to use and craft objects for hunting, to ignite and then control fire, to use natural materials for clothing and shelter, and to innovate and create technical solutions such as the wheel. Initial developments were undoubtedly prompted by survival instincts, but progressive advancements have resulted in the human race becoming so sophisticated that man is attempting to mimic the natural world so that we can imbue artificial objects with human characteristics. Although such an objective is associated with considerable ethical and technical challenges it is clear that there has been significant acceleration towards this goal over the last fifty years due to the complementary developments in computing power. The current availability of massively parallel, high speed processors offer not only a readymade prototyping test bed but also enable us to estimate the potential that we could achieve in the future.

These developments have also introduced a range of scientific disciplines that support and introduce further innovation. However, the inherent knowledge accumulated in these disparate disciplines has necessitated the specialization of researchers (particularly during the last 100 years) to enable them to operate and contribute to the advancement in their respective domains. Unfortunately this specialization restricts the ability of researchers to work in other domains and has created academic silos with one community often ignorant of the developments in the other. It is generally accepted that much of the new innovation will be spawned from the collaboration of experts across the boundaries of specialist areas. However, this is often restricted by the different language/terminology, methodologies and reference material drawn on by each discipline and which is not readily accessible or indeed easily shared across the boundary. As a result many researchers from different backgrounds have identified the need to establish large collaborative groupings across these boundaries. Fortunately this has also been recognized by national and international funding bodies who are encouraging such collaboration.

However, one of the major stumbling blocks towards improved collaboration and cooperation is the different terminology that has emerged. The area of research that attempts to draw on inspiration from biology or nature is not immune to this characteristic. Many young and idealistic PhD students despair at the variety of terminology defining similar areas and contributions within the area. This paper attempts to explore these different classifications in the context of the author's research. Section 2 introduces the main terms used to describe and represent the area whereas section 3 introduces the author's current research and reviews the current contributions in that domain. Section 4 attempts to classify that research and the paper concludes with a discussion of these issues.

2. Nomenclature

Recent developments have introduced spiking neurons which closely resemble the current understanding of neural mechanisms within the human brain. Such models range from computationally efficient yet biologically implausible models to more biologically accurate approaches that are associated with huge computational overheads. This trade-off can be overcome by drawing on advances from neuroscience research to determine those biological features that improve computational capability and yet enable effective description of the inherent neuron dynamics. Current research has presented network architectures, hardware/software implementations and introduced learning strategies for this new generation of spiking neural networks. However, the longer term challenge is to provide a computational approach that implements learning and reasoning in a human fashion that can be used by engineers to solve real world problems. This area of research has been represented by a number of widely used terms that are summarized here for convenience:

Computational Intelligence describes the area of research that emulates nature for problem solving. The techniques include neural networks, fuzzy systems, and evolutionary computing which have been inspired by learning, reasoning and adaptive processes within the natural world. Computational Intelligence research has been characterised by a progressively greater emphasis on providing biological plausibility and encouraging the integration of different techniques.

Soft Computing is defined as that area of research which is tolerant of imprecision, uncertainty, partial truth, and approximation as inspired by the human mind. The main constituents of Soft Computing are fuzzy logic, neural computing, evolutionary computation, machine learning and probabilistic reasoning. The techniques listed as constituents of Soft Computing are complementary rather than competitive.

Artificial Intelligence is the area of computer science focusing on creating machines that can engage on behaviors that humans consider intelligent. The field of Artificial Intelligence has split into two basic approaches; bottom-up and top-down. Bottom-up theorists believe the best way to achieve artificial intelligence is to build electronic replicas of the human brain's complex network of neurons ie neural networks and parallel computation whereas the top-down approach attempts to mimic the brain's behavior with approaches such as expert systems.

Intelligent Systems is that area of research that includes areas like artificial intelligence, models and computational theories of human cognition, perception and motivation; brain models, artificial neural nets and neural computing. It covers contributions from the social, human and computer sciences to the analysis and application of information technology. The field also covers critical analysis of intelligent systems, and addresses philosophical questions that arise.

Biologically-inspired (or bio-inspired) systems is the area of research into the use of computers to model nature, and simultaneously the study of nature to improve the usage of computers. The area relies heavily on the fields of biology, computer science and mathematics and the inherent computational techniques include evolutionary computation, swarm intelligence, neural networks, fuzzy systems, rough sets, and quantum computing.

The similarity and indeed huge overlap of these five terms is immediately obvious as each area claims common technologies; for example neural networks are a component technology in all five terms! These five generic names are widely used in the literature and are often interchanged by authors within their papers. Table 1 summarises searches performed using IEEE Xplore in an attempt to present a representative usage of these classifications by the research community. The ordering of Table 1 highlights (not surprisingly) that Artificial Intelligence is the most commonly used generic classification term which may be largely explained by its longevity. There is some anecdotal evidence to suggest that many researchers deliberately turned away from using this term in the 1980s and 1990s as they believed the area to be discredited due to the failure to deliver on the ambitious claims made by the initial contributors to such research. This may explain why in the last 10 years there appears to be more frequent reference to newer terms such as soft computing and bio-inspired systems as evidenced by the last column in Table 1. However, the use of the term artificial intelligence continues to dominate as the most prevalent term.

Table 1: Summary of searches using IEEE Xplore

Term	Term in Article Title	Term in all fields	Term in all fields with search restricted to post 2000 (% of total)
Artificial Intelligence	692	39858	27100 (68%)
Computational Intelligence	512	10777	7818 (73%)
Intelligent Systems	616	10445	7069 (68%)
Soft Computing	312	1142	857 (75%)
Bio-inspired systems	11	828	761 (92%)

Unfortunately, this classification is also associated with a small number of dedicated researchers who appear to be resolute in their defense and protection of their area to the exclusion of others working in related areas. While this contradicts the spirit of openness and enquiry of researchers it also highlights the difficulties for researchers working in those areas that naturally transcend such boundaries. Neural network research is representative of one such area and the following section reviews the recent contributions relating to spiking neurons.

3. Spiking Neural Network Research

Experimental studies in neurobiology have attempted to define the dynamics of the neuron and in particular, the synapse, in greater detail than ever before. A neuron is classified as either excitatory, and thus responsible for routing information through the network, or it is inhibitory and its function is to regulate the activity of excitatory neurons. Unsurprisingly, there are typically more excitatory than inhibitory neurons[1] although the role and configuration of inhibitory to excitatory neurons in any given biological network is unclear beyond these basic insights. Additionally it is known that synaptic transmission is unreliable.[2,3] In vivo experiments have shown that repeated stimulus of a neuron can lead to varying responses in the resulting transmission of spikes at a synapse.[4] Synapses have limited resources that they consume and replenish by varying rates. Typically, two types of behaviour of the synapse are distinguished, that of facilitating and depressing.[5] Facilitating synapses relay information through biological networks whereas depressing synapses are coincidence detectors. Facilitating synapses consume their resources gradually and have abundantly more resources than depressing synapses. In contrast, depressing synapses consume all their resources in the first few spikes they transmit, taking significant time to replace them. Similarly, the disposition of these types of synapses in a network is unclear.

With regard to learning in a SNN, it is known that synaptic efficacy is altered by coincidental firing between neurons. This is the basis of the well known Hebbian-type learning and explains the algorithm's historical endurance.[6] It is also clear that learning occurs strictly in a local sense. Hebbian learning of course, is typically an unsupervised learning algorithm in ANN research. However, it has also been adopted for supervised learning algorithms in SNN research due to its biological plausibility. SNNs exploit time as a resource so Hebbian learning algorithms needs to be temporal. STDP is an example of a temporal interpretation of Hebbian learning.[7,8,9] However, STDP is an unsupervised learning algorithm, and as such is not suited to tasks requiring a specific goal definition. Additionally, whereas a supervised learning algorithm will meet these requirements, it must also be locally-based.

Perhaps the only reasonable biological interpretation of supervised learning is supervision of a neuron by another neuron. In this instance, the supervisory neuron causes the supervised neuron to spike at desired times by transmitting spikes into the supervised neuron.[10, 11] In practice this supervised Hebbian learning (SHL) scheme has some drawbacks, chiefly among them that the weights of the network continue to be adjusted even after the desired output has been reached. A variation of the SHL learning algorithm can be made where the supervisory spike trains are not actually delivered to the network but are used 'remotely' to modify the weights. Employing a combination of STDP and anti-STDP, the Remote Supervision Method (ReSuMe)[12] is capable of 'propping up' inactive synaptic weights. ReSuMe is characterised by a capability to produce precise spike timing in an accurate and stable manner.

Modelling biologically plausible SNNs presents a significant challenge given the vast scale of real networks. An indication of the complexity required to model such architectures is readily demonstrated by considering the relationship between a biological neuron and the basic building block of the Integrated Circuit (IC), the transistor. The human brain is estimated to contain in the region of 10^{11} neurons, whereas the most advanced processor devices currently contain approximately $2x10^9$ individual transistors.[13] However, the comparison is further complicated by the recognition that. the inherent dynamics of the underlying neuron behaviour is significantly more complex than the basic switching principle of the transistor. Nonetheless, a number of strategies to address this significant challenge have been reported in the literature.

Given the high level of flexibility afforded, it is unsurprising that software based approaches have been heavily investigated. Examples of this include specialised simulation tools such as Neuron[21], Genesis[14], Emergent[15], SNNS[16], SpikeNET[17] and third party toolboxes such as BNN[18] and Neurosolutions[19] which are integrated within commercial general purpose simulation tools such as Matlab.[20] Whilst such tools provide powerful simulation environments for studying relatively small populations of neurons, the sequential Von Neumann serial processing architecture is simply not scalable towards simulating networks on the biological scale without incurring significant computation times. With the growing advent of multi-core and many-core processing, in recent years an increasing emphasis has been placed on the development of parallel simulators for multi-processor systems or computer clusters. For example, the developers of Neuron have released a parallel version of their software, for Genesis the PGenesis version is available whilst simulators such as neocortical simulation (NCS)[22] and neural simulation toolbox (NEST)[23] also offer support for parallel simulations. Whilst primarily developed to accelerate graphics rendering, the highly parallel structure of Graphics Processing Units (GPUs) has more recently attracted increasing interest amongst researchers aiming to model NN structures. [24-27] On a much larger scale, approaches such as the Blue Brain Project[28] and SpiNNaker[29] propose the use of a multitude of processing cores for the purpose of modeling very large scale SNNs. For example, the Blue Brain Project has performed simulations of the neocortex comprising up to 22 million neurons and 11 billion synapses utilising up to 8,192 processors operating simultaneously on the IBM Blue Gene/L supercomputer platform.[30] SpiNNaker has proposed a target architecture containing over 1 million ARM968 processor cores capable of simulating up to 1 billion neurons in real time. Whilst such approaches have significant potential, the reality is that access to such supercomputing power is not readily available.

As an alternative approach, extensive research has been conducted into custom hardware based NN implementations under the premise that when implemented in hardware, NNs can take full advantage of their inherent parallelism in a manner which is difficult to achieve with software.[31] Digital based approaches[32-34] are attractive in that they tend to offer high computational precision, high reliability, and high programmability. The primary disadvantages of these approaches however include the large amount of silicon area and power that is required for computation circuits such as multiplication and the relative slowness of computation. Analogue Application Specific Integrated Circuits (ASICs) on the other hand, offer interesting opportunities for NN implementations.[35] Calculations which, when modelled using digital techniques can be computationally intensive, can be reduced to simpler physical processes such as summing of currents or charges. Such devices, where analogue circuitry is used to model neuro-biological architectures, are often referred to as neuromorphic hardware.[36] Disadvantages of analogue technology meanwhile include the susceptibility to noise, heat and process-parameter variations that limit computational precision and the difficulty in managing weight storage. Hence, a large number of groups have targeted the development of mixed signal ASIC devices for neural processing, combining the strengths of both digital and analogue techniques.[37-44] In reality, although an attractive option in some regards, developing custom ASIC devices for NNs is both time consuming and expensive and as such, their use tends to be concentrated in areas where very either high performance is required or where large quantities will be deployed such as consumer products. An important aspect of biological networks is their inherent plasticity, something not easily integrated in ASIC design. While ASICs are very flexible in the design stage as they can implement any desired function, once manufactured they can no longer be modified without a re-spin and a modification of the basic neuron model would require a new development cycle to be undertaken. As a platform for neuroscientists and engineers to explore neuron models, parameters and network topologies, such a constraint is an important factor to be considered.

Significant advancements in the domain of reconfigurable computing has resulted in FPGAs becoming increasingly popular for implementing complex computational systems.[45] FPGAs permit the implementation of digital systems, providing an array of logic components that can be configured in a desired way by a configuration bitstream.[31] These devices, which can be quickly and easily programmed and reprogrammed to perform a large variety of functions, [46] are particularly appealing when attempting to recreate to some degree the natural plasticity and self-adaptation of biological systems in electronic hardware. Early approaches that reported on the use of FPGAs for modelling SNNs sought to maximise the network population density by using highly simplified neuron models that were highly abstracted from the biological principles of neuronal signalling in the brain.[47-53] Whilst such approaches have reported on the successful implementation of networks ranging from 3 to 100 neurons, they are simply not scalable towards simulating large scale network sizes, irrespective of how much the neuron model can be optimised. Also, there will always be a demand for platforms that will provide more biological plausibility. An implementation of a highly complex model such as that by Mak et al[54] illustrates that while this is achievable, the logic demands were such that only a single neuron could be accommodated on the target FPGA device. Such approaches however, utilizing fully parallel implementation approaches, reported huge performance increases in terms of the computation time. A number of researchers therefore explored the possibility of balancing this speed/area trade off, employing the use of resource sharing or TDM to increase the network density achievable whilst still offering performance improvements over software based implementations. One such approach that reports on the use of time multiplexing has been presented by Graas et al.[55] As opposed to the more simplified models, this approach is built upon highly detailed neuron models closely aligned with biological principles. For example, two neuron models were implemented in FPGA hardware, the Hodgkin Huxley (HH) model[56] and the Booth and Rinzel two-compartment model of a motoneuron.[57] The target platform incorporated a Xilinx Virtex series XCV1000 FPGA device and the Xilinx System Generator (XSG) tool was used as the design environment. Extensive use of pipelining was used in conjunction with a simple TDM control mechanism to support the simulation of up to 170 independent neurons whereby a speed-up factor of 16 was achieved over a software simulation on a 1.3GHz AMD processor.

A key limitation of this implementation however was the lack of support for modelling networks at the population level although a subsequent revision of the system addressed this issue.[58] The updated approach was validated with a 40-neuron population model consisting of HH style conductances and fully interconnected synapses[59] implemented on a Xilinx Virtex 4 series XC4VSX35 device. While the authors state that the approach caters for neural populations in the 10’s-100’s range they identify that further refinements are required to enable networks containing thousands of neurons. Whilst this work comprises a significant contribution to the research field, alternative strategies are required to enable the simulation of large scale networks. Using the less complex Leaky Integrate and Fire (LIF) neuron model, Schrauwen et al[60] reported on the implementation of networks containing 200 neurons. Using serial processing to sum the synaptic inputs to the neuron and parallel arithmetic for the neuron computation, the authors found that performance of up to 347 times faster than real time could be achieved using 60% of the available FPGA logic resources on a Xilinx Virtex 4 XC4VSX35 device. Alternatively, using serial processing to sum the synaptic inputs and serial arithmetic for the neuron computation the authors found that real time performance could be maintained whilst using considerably less hardware resources, i.e. 4% of the available FPGA logic on the XC4VSX35. Whilst this work demonstrated that real time performance could be guaranteed for relatively modest networks containing up to 200 neurons, an approach offering similar real time performance for larger networks has been reported by Pearson.[61,62] Using biologically plausible mammalian neuron topologies such as sections of the basal ganglia and the trigeminal sensory complex of the rodent brain stem, networks incorporating up to 1,100 LIF neurons have been successfully implemented on a XC2V1000 Xilinx Virtex 2 series FPGA device. A limitation of this approach however, is the relatively small ratio of synapses per neuron (16:1) that can be accommodated. As acknowledged by the authors, this is insufficient for many networks. A related approach by Ros et al, employing the Spike Response Model (SRM) as the basic underlying neuron model, suggests a more scalable architecture whilst also targeting guaranteed real time performance.[63] A hardware software partitioning strategy is adopted whereby the hardware component consists of a Peripheral Component Interface (PCI) based FPGA development board acting as a reconfigurable neuroprocessor while the software component running on the host PC is responsible for maintaining the network connectivity, for routing spikes between neurons and for implementing learning algorithms. The authors make extensive use of pipelining in their neuron computation circuitry implemented on the FPGA hardware and also employ several Processing Units (PUs), operating in parallel to maximise system performance. Using the RC1000 prototyping platform, which incorporates a Xilinx Virtex-2000E device, the authors have implemented and tested the system with 1,024 neurons with up to four PUs operating in parallel. Unlike many of the other approaches, this architecture also offers support for on chip training. A limitation of the approach however is that the learning algorithms are computed in the software component of the system with significant performance degradation being observed if training is employed for more than 5% of the network synapses.

While significant progress has already been made in understanding neurons dynamics, considerably less has been achieved in developing efficient spiking neural learning mechanisms. To date, a number of supervised and unsupervised learning methods have been developed, most of which do not scale up and would require retraining in a continuously changing environment.

SpikeProp is an adaptation of the classical backpropagation algorithm that can perform complex non-linear classification in fast temporal coding just as well as rate-coded networks.[64] Moore[65] attempted to replicate the findings of Bohte and the weights were initialized with the values that led the network to successful training in a similar number of iterations, but with large learning rates. Xin and Embrechts[66] proposed a modification of the learning algorithm by including the momentum term in the weight update equation. It has been demonstrated that this modification significantly speeded up the convergence of SpikeProp. Additional learning rules were introduced to make it possible to learn not only the weights, but also the synaptic delays, time constants and the neurons' thresholds. This resulted in smaller network topologies and with faster algorithm convergence. Inspired by learning rules for locally recurrent analog neural networks, Schrauwen and Campenhout[67] also presented a new learning rule for spiking neurons that used the general population-temporal coding model. As a result the learning rule was able to operate on a broad class of output codings smoothly and quickly. Tiňo and Mills[68] extended SpikeProp to recurrent network topologies, to account for the temporal dependencies in the input stream. Wu et al[69] applied weight limitation constraints to the SpikeProp algorithm and presented a novel solution to the problem raised by non-firing neurons which makes the learning algorithm converge reliably and efficiently. Silva made corrections and improvements to the standard SpikeProp training based on the Levenberg-Marquardt method by introducing a new encoding scheme with fast convergence.[70] Finally McKennoch et al[71] developed and analyzed SNN versions of Resilient Propagation (RProp) and QuickProp, both training methods used to speed up training in ANNs by making certain assumptions about the data and the error surface.

Modifications were made to both algorithms to adapt them to SNNs. Results generated on standard XOR and IRIS dataset were shown an average of 80% faster than using SpikeProp on its own.

Neither the original SpikeProp method nor any of the proposed modifications enable learning of patterns composed of more than one spike per neuron. But the fact that temporal events can happen more than once, make temporal patterns actually different from static patterns. So in order to process such patterns with an SNN, the neurons have to be able to spike more than once. Xie and Seung[72] presented a synaptic update rule for learning in networks of spiking neurons. It was shown that irregular spiking similar to that observed in biological neurons could be used as the basis for a learning rule. The learning rule was derived based on a special class of model networks in which neurons fire spike trains. The learning rule was on average performing gradient ascent on an expected reward function. Booij and Hieu[73] presented a supervised learning rule for SNN that can cope with neurons that spike multiple times. The rule was developed by extending the existing SpikeProp algorithm, and was successfully tested on a classification task of Poissonian spike trains. These gradient based algorithms are computationally powerful, are often regarded non-biologically plausible because these algorithms required nonlocal spread of error signals from one synapse to another.

Synfire Chains based on a Hebbian learning rule describe neural maps organised in a feedforward manner with random connections between maps showing synchronous activity.[74] It was assumed that the time of postsynaptic neuron firing depends mostly on the signal propagation delay in the presynaptic neurons. The 'time-weight' dependence is neglected and the topology of the network is modified to obtain the desired delay between input and output. Ruf and Schmitt presented a supervised-Hebbian learning methods, one of the first spike-based methods for classification task.[10] In Ref 75, a supervised-Hebbian learning method was realised by the extra input currents injected to the learning neuron, the learning neuron was forced to fire at the target points in time and prevented it from firing at other times. The learning algorithm was able to approximate the given target transformations quite well although parameters continued to be changed even if the neuron fired already exactly at the desired times. The presented method proved high ability to implement the precise spike timing coding scheme. ReSuMe[12] integrated the concept of learning-windows with the novel concept of remote supervision. The experiment confirmed that ReSuMe can efficiently learn the desired temporal sequences of spikes and that the learning process converges quickly. This method enables the network to learn multiple patterns of spikes. Since the synaptic weights are updated in an incremental manner and the method is suitable for online processing. In other work that also used the STDP rule, a supervised training algorithm was developed that affected weights both locally and at network level.[76]

In Ref 77, learning was achieved by synaptic changes that depended on the firing of pre- and postsynaptic neurons, and that were modulated with a global reinforcement signal. The efficacy of the algorithm was verified in a biologically-inspired experiment. In Ref 78, the modulation of the STDP by a global reward signal led to reinforcement learning. However, it was only applied to the XOR problem and still need to be demonstrated on large datasets. Seung[79] provided the explanations for different dynamics of the synaptic plasticity related to the reward signal. Based on direct reinforcement learning algorithm, overall qualitative performance was comparable to the classic algorithms based on temporal difference and value function approximation, but with a higher computational cost.[80] Based on a use dependent synaptic potentiation and depotentiation, a self-organisation algorithm[81] was developed and successfully performed in autonomous robot application. The time needed for the training using self-organization method was much less than with genetic evolution. Amin[82] presented a new learning algorithm that can perform learning in one step and utilises only synaptic weights for learning. The proposed algorithm was simpler than past approaches and more practical to implement in hardware. It was demonstrated on sound classification and function approximation. A new learning algorithm for SNNs that uses the inter-spike times within a spike train was introduced.[83] The learning algorithm utilised the spatio-temporal pattern produced by the spike train input mapping unit and adjusts synaptic weights during learning and the approach was demonstrated on classification problems.

Belatreche et al. proposed a derivative-free supervised learning algorithm and used an evolutionary strategy to minimise the error between the output firing times and the corresponding desired firing times.[84] Also in Ref 85, based on evolutionary computation techniques, the ability of the robots to distinguish sounds composed of parts of real canary songs and to navigate to the recognised signal was evaluated. Palvidis et al[86] developed a Parallel Differential Evolution algorithm which was successfully tested on well-known and widely used classification problems. Johnston et al[87] developed a hybrid learning algorithm fusing STDP with genetic algorithms based explicit delay learning. As the training is an evolutionary strategy-based iterative process, the training procedure was extremely time consuming and is not suitable for online learning. Barber[88] proposed a statistical learning criterion to derive a supervised spike-based learning algorithm. The method considered supervised learning for neurons operating on the discrete time scale. Pfister et al. extended this study to the continuous case.[89]

Unsupervised spike-based learning methods, such as Long-term Potentiation (LTP), Long-term Depression (LTD), Spike-Timing Dependent Plasticity (STDP) and Hebbian learning have already been widely investigated and described in the literature.[90-92] In Ref 91, several mathematical formulations of correlation-based Hebbian learning were reviewed. The state of the presynaptic neuron was described either by a firing rate or by presynaptic spike arrival. The state of the postsynaptic neuron can be described by its firing rate, its membrane potential or the timing of backpropagating action potentials. Due to its intrinsic normalization properties, Hebbian synaptic plasticity stabilised postsynaptic firing rates and led to subtractive weight normalisation. Kistler[92] presented a phenomenological model of STDP that was based on a Volterra series-like expansion. Integral kernels were used to describe synaptic weight changes as a function of the relative timing of pre- and postsynaptic spikes.

In Ref 93 it was showed how a SNN based on spike-time coding and Hebbian learning performed unsupervised clustering successfully from realistic data. In SNNs, delay learning is achieved through delay selection and delay shift. Adibi et al[90] introduced a new delay shift approach for learning in RBF-like SNNs. There were single delayed connections between the input and the RBF neurons and the delays were adapted in an unsupervised learning process. It was shown the clustering precision of the proposed network was considerably higher than that of the similar neural networks. Wade et al[94] presented an unsupervised training algorithm for SNNs that merged the Bienenstock-Cooper-Munro (BCM) learning rule with STDP, stimulated using spike trains. The BCM rule combined with STDP modulated the height of the plasticity window. The network was applied to the IRIS dataset and the results showed convergence accuracy comparable to other SNN training algorithms.

Wysoski et al[95] presented a simple online procedure to perform learning for a four layers of hierarchical neural network of two-dimensional integrate-and-fire neuronal maps. The training was done through synaptic plasticity and adaptive network structure. Event driven approach was used to optimize computation speed in order to simulate networks with large number of neurons. The training procedure was applied to a publicly available face recognition dataset, and the obtained performance was comparable to the optimised off-line method. In Ref 96, a simple artificial gustatory model was used in SNNs for taste recognition. An evolving learning algorithm was developed based on simple integrate-and-fire neurons with rank order coded inputs. How the information encoding in a population of neurons influenced the performance of the networks was also explored. However, these approaches still need to address a number of issues such as fine tuning of learning parameters, automatic update of learning parameters in continuously changing environments (as these were set manually), improving learning speed for large size datasets, and the effect of handling imbalanced datasets on the training performance. Alnajjar et al[97] developed a novel self-adaptation system to train a real mobile robot for optimal navigation in dynamic environments by training SNNs having the STDP property. All the trained SNNs were stored in a tree-type memory structure that were used as experiences for the robot to enhance its navigation ability in new and previously trained environments. The memory was designed to have a simple searching mechanism. Forgetting and on-line dynamic clustering techniques were used in order to control the memory size. Experimental results showed that a robot provided with learning and memorizing capabilities was able to survive in complex and dynamic environments.

There has been considerable research focus on developing offline approaches for SNNs, but very little has been achieved in developing online learning approaches for SNNs. Developing efficient online learning approaches for SNNs is thus very important for increasing the applicability of SNNs as an intelligent system capable of handling continuous streams of information, scaling up and adapting to continuously changing environments. Similarly there are ongoing challenges of scale such that we can realize more realistic networks that can solve real world problems. This will involve hardware implementations to ensure real time operation. Thus further research contributions will involve input from biologists, engineers, mathematicians, physicists, psychologists, computer scientists etc and will thus demand a common terminology to ensure ease of communication across these diverse disciplines.

4. Commentary

The previous section presented a review of the recent research contributions in the area of spiking neural networks which has highlighted the various engineering contributions that attempt to address the challenges facing this embryonic research area, many of which are common to the more traditional neural networks. It is thus very clear that spiking neural networks are just another variant of classical neural networks and indeed have already been identified as the third such generation of neural networks.[98] However, the question now remains on how to classify this research area within the more generic terms. Initially this appears as quite a trivial challenge as neural networks are already associated with each area but surely a more careful consideration is required if not simply to offer a term that provides an onomatopoeia-like reference for other researchers. Furthermore, such a definition should also consider that such research often draw heavily on other areas and indeed are characterized by developments in hybrid technologies. For example, a recent development in spiking neural networks has witnessed the introduction of fuzzy reasoning that has introduced a new computational paradigm.[99]

Before attempting to consider the suitability of each term it is more appropriate to first define the motivation for research into spiking neural networks and thus map its rationale onto such a generic term. Neural networks are essentially connectionist models of biological neurons that attempt to harness the massively parallel, distributed computation of biological brains. Neural network research is characterised by a progressively greater emphasis paid to biological plausibility. Spiking neurons are based on the realisation that the precise mechanism by which biological neurons encode and process information is still poorly understood. In particular, biological neurons communicate using action potentials also known as spikes or pulses. The spatio-temporal distribution of spikes in biological neurons is believed to ’hold the key’ to understanding the brain’s neural code. Spiking neurons model this form of input stimulus and in this way exploit time as a resource in the neural code. There exists a multitude of spiking neuron models that can be employed in SNNs. The models range from the computationally efficient on the one hand to the biologically accurate on the other[100]; the former are typically of the integrate-and-fire variety and the latter are of the Hodgkin-Huxley type. All the models in this range exploit time as a resource in their computations but vary significantly in the number and kinds of neuro-computational attributes.

The definition of Soft Computing includes neural networks and thus would also include spiking neural network research; although its name does not suggest its inclusion. In the author’s opinion the term “soft computing” is not descriptive of this area and in many ways misleading; the use of the word “soft” may suggest easy, rather than a collection of intelligent technologies. This may be controversial within the constraints of this special issue but the term Soft Computing is more closely associated with the fuzzy community, and it is undoubtedly more appropriate in that context as it highlights and represents the vagueness of decision boundaries.

Similarly the term Artificial Intelligence in some ways suggests that the inherent intelligence is not natural but rather contrived as artificial, and thus not inspired from the biology. Many of the initial contributions in Artificial Intelligence were indeed motivated by this premise as researchers developed computing techniques and algorithms that could be described as intelligent. Thus Artificial Intelligence does not seem to be an appropriate generic term for spiking neural networks as this research domain is primarily motivated by biological plausibility. Spiking neural network researchers would rarely use the term Artificial Intelligence and generally avoid the subsymbolic descriptions of their networks in this domain. The field of artificial intelligence is normally concerned with a more abstract or high level approach to representing intelligence. Artificial Intelligence is now confined to the discipline of Computer Science encompassing expert systems, high level reasoning, database mining, knowledge based systems, and natural language processing.

The term Intelligent Systems provides a more generic term for the research community and the name provides a clear definition of the area. However, in many ways the term is too generic as it can be used to classify any engineering system that demonstrates intelligent behaviour. Such systems could be motivated from diverse areas such as social or mechanical engineering and as a result may have little or no computational requirements. Similarly the term Bio-Inspired systems is also representative but would also suffer from this "too generic" label. However both terms are highly descriptive of the spiking neural network research area but should be reserved for higher level adjectives describing the wider context of the research area.

The final term to consider is Computational Intelligence which in the author's opinion is the most suitable umbrella term for this area. This term clearly identifies a research area that attempts to emulate intelligent systems on a computational platform. The area includes a collection of research techniques that have been inspired from biology and which includes neural networks, fuzzy reasoning and evolutionary computing which have been inspired by learning, reasoning and adaptive processes within the natural world. Computational Intelligence research has been characterised by a progressively greater emphasis on providing biological plausibility and encouraging the integration of different techniques. Such biological plausibility is an important motivation for spiking neural network researchers and the computational aspects clearly differentiate the area from related developments in the wider neuroscience community. The definition of the area is also not restrictive and will readily accommodate new computational developments in the area.

5. Conclusion

This paper has attempted to navigate through the various definitions of the research area that claim to represent the contributions in spiking neural network research. The review section provided an overview of spiking neural network research including its motivation and main contributions. The previous discussion presented the author's opinion on this classification and the use of terms to represent the area. It is clear there is widespread acceptance of a number of descriptive terms and that no single consensus can be reached. However, the discussion concluded that a number of terms are more suitable to represent the research area and suggested that there is a hierarchy of such terms.

At the bottom of this hierarchy is the research area of spiking neural networks used by the author to illustrate this classification. This area is clearly related to the more traditional neural network research and thus related to other areas such as fuzzy logic and evolutionary computing. These techniques are all characterized by computational approaches to emulate the learning, reasoning and adaptive processes within the natural world. Thus the most appropriate term to define these areas is Computational Intelligence. Other terms at this next level that could be considered were Soft Computing and Artificial Intelligence. However, both were discounted as they did not reflect the biological plausibility of the research area or indeed did the terms suggest its inclusion. There is no doubt that there is some overlap with the inherent technologies associated with the three terms at this level; in many ways the boundaries between each are vague or fuzzy! This vagueness of decision boundaries is further compounded if we consider that at this level the term Computational Neuroscience could also be used to classify spiking neural networks in the wider research domain of Neuroscience. However, the important term in this context is computation as it sets the context that the research is to be realised on a computational platform. Finally, the higher level term that should be used to represent the areas of Computational Intelligence, Soft Computing and Artificial Intelligence is Intelligent Systems/Bio-inspired Systems. The use of word "systems" clearly highlights a higher level representation of the area and provides an all-encompassing onomatopoeia-like reference for other researchers.

Acknowledgements

The author acknowledges the support of the Intelligent Systems Research Centre, University of Ulster and the

contribution of the following colleagues in this research: Martin McGinnity, Liam McDaid, Ammar Belatreche, Brendan Glackin and Cornelius Glackin.

References

1. Braitenberg, V; Schuz, A: "Cortex-statistics and geometry of neuronal connectivity," Springer, 1998.
2. Allen, C; Stevens, CF: "An evaluation of causes for unreliability of synaptic transmission," Proceedings of the National Academy of Sciences, vol. 91, no. 22, pp. 10380-10383, 1994.
3. Senn, W; Schneider, M; Ruf, B: "Activity-dependent development of axonal and dendritic delays, or, why synaptic transmission should be unreliable," Neural Computation, vol. 14, no. 3, pp. 583-619, 2002.
4. Thomson, AM; Deuchars, J: "Temporal and spatial properties of local circuits in neocortex," Trends in Neuroscience, vol. 17, no. 3, pp. 119, 1994.
5. Charlton, MP; Smith, SJ; Zucker, RS: "Role of presynaptic calcium ions and channels in synaptic facilitation," The Journal of Physiology, vol. 323, no. 1, pp. 173-193, 1982.
6. Hebb, DO: "The Organization of Behavior: A Neuropsychological Theory," Wiley, 1949.
7. Bi, GQ; Poo, MM: "Distributed synaptic modification in neural networks induced by patterned stimulation," Nature, vol. 401, no. 6755, pp. 792-795, 1999.
8. Song, S; Miller, KD; Abbott, LF: "Competitive Hebbian learning through spike-timing-dependent synaptic," Nature Neuroscience, vol. 3, pp. 919-926, 2000.
9. Abbott, LF; Nelson, SB: "Synaptic plasticity: Taming the beast," Nature Neuroscience, vol. 3, pp. 1178-1183, 2000.
10. Ruf, B; Schmitt, M: "Learning temporally encoded patterns in networks of spiking neurons," Neural Processing Letters, vol. 5, no. 1, pp. 9-18, 1997.
11. Legenstein, R; Naeger, C; Maass, W: "What can a neuron learn with spike-timing-dependent plasticity?," Neural Computation, vol. 17, no. 11, pp. 2337-2382, 2005.
12. Kasinski, A; Ponulak, F: "Experimental demonstration of learning properties of a new supervised learning method for the spiking neural networks," Lecture Notes In Computer Science, vol. 3696, pp. 145, 2005.
13. Intel, Press Release, Intel Previews Intel Xeon 'Nehalem-EX' Processor, 2009.
14. Aisa, B; Mingus, B; O'Reilly, R: "The emergent neural modeling system," Neural Networks, vol. 21, pp. 1045–1212, 2008.
15. Bower J; Beeman, D: The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System, 2nd Ed. Springer-Verlag, New York, 2008.
16. Petron, E: "Stuttgart neural network simulator: Exploring connectionism and machine learning with snns," Linux Journal, 1999. Available: http://portal.acm.org/citation.cfm?id=327908
17. Delorme, D; Gautrais, J; Rullen, R; Thorpe, S: "Spikenet: A simulator for modeling large networks of integrate and fire neurons," Neurocomputing, vol. 26-27, pp. 998–996, 1999.
18. BNN, Biological Neural Networks Toolbox for MATLAB: User Guide, Version 1.0, August 2004. Available:http://www.ymer.org/research/files/bnntoolbox /ver1.1/BNNtoolbox.pdf
19. NeuroDimension, Neurosolutions Software Simulator Tool.. Available: http://www.neurosolutions.com/products/ns/
20. Mathworks, Matlab User Guide. Available: http://www.mathworks.com/access/helpdesk/help/pdf doc/matlab/getstart.pdf
21. Hines, M; Carnevale, N: "The neuron simulation environment," Neural Computation, vol. 9, no. 6, pp. 1179–1209, 1997.
22. Zirpe, M: "Rain and ncs 5 benchmarks," Master's thesis, University of Nevada, December 2007.
23. Morrison, A; Mehring, C; Geisel, T; Aertsen, A; Diesmann, M: "Advancing the boundaries of high-connectivity network simulation with distributed computing." Neural Comput, vol. 17, no. 8, pp. 1776–1801, Aug 2005.
24. Luo, Z; Liu, H; Yang, Z; Wu, X: "Artificial neural network computation on graphic process unit," in Proc. IEEE International Joint Conference on Neural Networks IJCNN '05, vol. 1, 2005, pp. 622–626 vol. 1.
25. Lahabar, S; Agarwal, P; Narayanan, P: "High performance pattern recognition on gpu," in Proceedings of National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics, 2008.
26. Jang, H; Park, A; Jung, K: "Neural network implementation using cuda and openmp," in Proc. DICTA '08.Digital Image Computing: Techniques and Applications, 1–3 Dec. 2008, pp. 155–161.
27. Bernhard, F: "Spiking neurons on gpus," Int Conference on Computational Science (4), 2006, pp. 236–243.
28. Markram, H: "The blue brain project," Nat Rev Neurosci, vol. 7, no. 2, pp. 153–160, Feb 2006.
29. Furber, S; Temple, S; Brown, A: "High-performance computing for systems of spiking neurons," in Proceedings of the AISB'06 workshop on GC5: Architecture of Brain and Mind, vol. 2, Bristol, UK, April 2006, pp. 29–36.
30. Djurfeldt, M; Lundqvist, M; Johansson, C; Rehn, M; Ekeberg, O; Lansner, A: "Brain-scale simulation of the neocortex on the IBM blue gene/l supercomputer," IBM Journal of Research and Development, vol. 52, no. 1, pp. 31–41, 2008.
31. Upegui, A; Pea-Reyes, C; Sanchez, E: "A methodology for evolving spiking neural-network topologies on line using partial dynamic reconfiguration," in Proceedings of International Conference on Computational Intelligence (ICIC'03), Medellin, Colombia, 2003.

32. Jahnke, A; Roth, U; Klar, H: "A simd/dataflow architecture for a neurocomputer for spike-processing neural networks (nespinn)," in Proc 6th Int Conf on Microelectronics for Neural Networks (Feb 1996, pp. 232–237.
33. Schoenauer, T; Mehrtash, N; Jahnke, A; Klar, H "Maspinn: Novel concepts for a neuro-accelerator for spiking neural networks," Proceedings of SPIE Workshop on Virtual Intelligence/Dynamic Neural Networks, vol. 3728, pp. 87–96, 1998.
34. Schoenauer, T; Atasoy, S; Mehrtash, N; Klar, H: "Simulation of a digital neuro-chip for spiking neural networks," in Proc. IEEE Int Joint Conf on Neural Networks IJCNN 2000, vol. 4, 2000, pp. 490–495 vol.4.
35. Chicca, E; Badoni, D; Dante, V; D'Andreagiovanni, M; Salina, G; Carota, L; Fusi, S; Del Giudice, P; "A vlsi recurrent network of integrate-and-fire neurons connected by plastic synapses with long-term memory," IEEE Trans on Neural Networks, vol. 14, no. 5, pp. 1297–1307, 2003.
36. Mead, C: "Neuromorphic electronic systems," Proc of the IEEE, vol. 78, no. 10, pp. 1629–1636, 1990.
37. Indiveri, G; Chicca, E; Douglas, R: "A VLSI array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity," IEEE Trans on Neural Networks, vol. 17, no. 1, pp. 211–221, 2006.
38. Vogelstein, R; Mallik, U; Vogelstein, J; Cauwenberghs, G: "Dynamically reconfigurable silicon array of spiking neurons with conductance-based synapses," IEEE Trans on Neural Networks, vol. 18, no. 1, pp. 253–265, 2007.
39. Vogelstein, R; Mallik, U; Cauwenberghs, G; Culurciello, E; Etienne-Cummings, R: "Saliency-driven image acuity modulation on a reconfigurable silicon array of spiking neurons," Advances in Neural Information Processing Systems, vol. 17, p. 14571464, 2005.
40. Mallik, U; Vogelstein, R; Culurciello, E; Etienne-Cummings, R; Cauwenberghs, G: "A real-time spike-domain sensory information processing system [image processing applications]," in Proc. IEEE International Symposium on Circuits and Systems ISCAS 2005, 2005,pp. 1919–1922 Vol. 3.
41. Schemmel, J; Meier, K; Mueller, E: "A new vlsi model of neural microcircuits including spike time dependent plasticity," in Proc. IEEE Int Joint Conf on Neural Networks, vol. 3, 2004, pp. 1711–1716 vol.3.
42. Schemmel, J; Bruderle, D; Meier, K; Ostendorf, B: "Modeling synaptic plasticity within networks of highly accelerated i&f neurons," in IEEE International Symposium on Circuits and Systems 2007, pp. 3367–3370.
43. Renaud-Le Masson, S; Le Masson, G; Alvado, L; Saghi, S; Tomas, J: "A neural simulation system based on biologically realistic electronic neurons," Information Sciences, vol. 161, no. 1-2, pp. 57–69, April 2004.
44. Zou, Q; Bornat, Y; Tomas, J; Renaud, S; Destexhe, A; "Real-time simulations of networks of hodgkin-huxley neurons using analog circuits," Neurocomputing, vol. 69, no. 10-12, pp. 1137 – 1140, 2006.
45. Wolf, W: FPGA Based System Design. Prentice Hall, 2004.
46. Gokhale M; Graham, P: Reconfigurable Computing: Accelerating Computation with Field-Programmable Gate Arrays. Springer, 2005.
47. Maguire, LP; McGinnity, TM; Glackin, B; Ghani, A. Belatreche, A; Harkin, J: "Challenges for large-scale implementations of spiking neural networks on fpgas," Neurocomput., vol. 71, no. 1-3, pp. 13–29, 2007.
48. Cassidy, A; Denham, S; Kanold, P; Andreou, A: "Fpga based silicon spiking neural array," in Proc. IEEE Biomedical Circuits and Systems Conference BIOCAS 2007, 2007, pp. 75–78.
49. Rossmann, M; Hesse, B; Goser, K; Buhlmeier, A; Manteuffel, G; "Implementation of a biologically inspired neuron-model in fpga," in Proceedings of the 5th International Conference on Microelectronics for Neural Networks and Fuzzy Systems, Feb 1996, pp. 322–329.
50. Maya, S; Reynoso, R; Torres, C; Arias-Estrada, M: "Compact spiking neural network implementation in fpga," in Procs of 10th Field-Programmable Logic and Applications (FPL). London, 2000, pp. 270–276.
51. Upegui, A; Pea-Reyes, CA; Sanchez, E: "An fpga platform for on-line topology exploration of spiking neural networks," Microprocessors and Microsystems, vol. 29, no. 5, pp. 211 – 223, 2005.
52. Roggen, D; Hofmann, S; Thoma, Y; Floreano, D: "Hardware spiking neural network with run-time reconfigurable connectivity in an autonomous robot," in Proc. NASA/DoD Conference on Evolvable Hardware, 2003, pp. 189–198.
53. Guerrero-Rivera, R; Morrison, A; Diesmann, M; Pearce, T; "Programmable logic construction kits for hyper-real-time neuronal modeling." Neural Comput, vol. 18, no. 11, pp. 2651–2679, Nov 2006.
54. Mak, T; Rachmuth, G; Lam, KP; Poon, C-S: "Field programmable gate array implementation of neuronal ion channel dynamics," in Proc. 2nd Int IEEE EMBS Conference on Neural Engineering, 2005, pp. 144–148.
55. Graas, E; Brown, E; Lee, R: "An fpga-based approach to high-speed simulation of conductance-based neuron models." Neuroinformatics, vol. 2, no. 4, pp. 417–436, 2004.
56. Hodgkin A; Huxley, A: "A quantitative description of membrane current and its application to conduction and excitation in nerve." J Physiol, vol. 117, no. 4, pp. 500–544, Aug 1952.
57. Booth V; Rinzel, J: "A minimal, compartmental model for a dendritic origin of bistability of motoneuron firing patterns." J Comput Neurosci, vol. 2, no. 4, pp. 299–312, Dec 1995.
58. Weinstein, R; Reid, M; Lee, R: "Methodology and design flow for assisted neural-model implementations in fpgas," IEEE Trans on Neural Systems and Rehabilitation Engineering, vol. 15, no. 1, pp. 83–93, 2007.

59. Butera, RJ; Rinzel, J; Smith, JC: "Models of respiratory rhythm generation in the pre-btzinger complex. i. bursting pacemaker neurons." J Neurophysiol, vol. 82, no. 1, pp. 382–397, Jul 1999.
60. Schrauwen, B; D'Haene, M; Verstraeten, D; Campenhout, J: "Compact hardware liquid state machines on fpga for real-time speech recognition." Neural Networks, vol. 21, no. 2-3, pp. 511–523, 2008. [Online].
61. Pearson, M; Gilhespy, I; Gurney, K; Melhuish, C; Mitchinson, B; Nibouche, M; Pipe "A real-time, fpga based, biologically plausible neural network processor," in Procs ICANN'05, Warsaw, September 2005.
62. Pearson, M; Gilhespy, I; Gurney, K; Melhuish, C; Mitchinson, B; Nibouche, M; Pipe, A: "Implementing spiking neural networks for real-time signal-processing and control applications: A model-validated fpga approach," IEEE Transactions on Neural Networks, vol. 18, no. 5, pp. 1472–1487, 2007
63. Ros, E; Ortigosa, E; Agis, R; Carrillo, R; Arnold, M: "Real-time computing platform for spiking neurons (rt-spike)," IEEE Transactions on Neural Networks, vol. 17, no. 4, pp. 1050–1063, 2006
64. Bohte, S.M., Kok, J.N. & Poutre, H.L. 2002, "Error-backprogation in Temporally Encoded Networks of spiking neurons", Neurocomputing, vol. 48, pp. 17-37.
65. Moore, S. 2002, Back-Propagation in Spiking Neural Networks MSc thesis, University of Bath.
66. Xin, J. & Embrechts, M., J. 2001, "Supervised Learning with Spiking Neuron Networks", In Proceedings IEEE International Joint Conference on Neural Networks, IJCNN'01Washington D.C., pp. 1772-1777.
67. Schrauwen, B. & Campenhout, J.,Van 2006, " Backpropagation for Population-Temporal Coded Spiking Neural Networks". 2006 IEEE International Joint Conference on Neural Networks, Vancouver, BC, Canada.
68. Tino, P. & Mills, A., J.S. 2005, "Learning Beyond Finite Memory in Recurrent Networks of Spiking Neurons" in Advances in Natural Computation- ICNC 2005, Lecture Notes in Computer Science, eds. L. Wang, K. Chen & Y.S. Ong, Springer-Verlag, , pp. 666-675.
69. Wu, QX, McGinnity, TM, Maguire, LP, Glackin, B & Belatreche, A 2006, "Learning under weight constraints in networks of temporal encoding spiking neurons", Neurocomputing, vol. 69, no. 16-18, pp. 1912-1922.
70. Silva, S.M.1. & Ruano, A.E.1. 2007, "Application of the Levenberg-Marquardt method to the training of spiking neural networks", 2006 IEEE Int Joint Conf on Neural Networks, Vancouver, BC, Canada, pp. 3978.
71. McKennoch, S.1., Liu1, D. & Bushnell, L.G.1. 2007, "Fast modifications of the SpikeProp algorithm", 2006 IEEE International Joint Conference on Neural Networks , Vancouver, BC, Canada, pp. 3970.
72. Xie, X. & Seung, H.S. 2004, "Learning in neural networks by reinforcement of irregular spiking", Physical Review E, vol. 69, no. 4, pp. 041909-1.
73. Booij, O.1. & Hieu tat Nguyen1 2005, "A gradient descent rule for spiking neurons emitting multiple spikes", Information Processing Letters, vol. 95, no. 6, pp. 552-8.
74. Sougne, J., P. 2001, "A learning algorithm for synfire chains" in Connectionist Models of Learning, Development and evolution, eds. R. M. French & J. P. Sougne, Springer Verlag, London, pp. 23-32.
75. Legenstein, R., Naeger, C. & Maass, W. 2005, "What can a Neuron Learn with Spike-Timing-Dependent Plasticity?", Neural Computation, vol. 17, pp. 2337-2382.
76. Strain, T.J., McDaid, L.J., Maguire, L.P. & McGinnity, T.M. 2006, "A supervised STDP based training algorithm with dynamic threshold neurons", International Joint Conference on Neural Networks 2006, IJCNN '06, Piscataway, NJ 08855-1331, United States, Vancouver, BC, Canada, pp. 3409.
77. Florian, R., V. 2005, "A reinforcement learning algorithm for spiking neural networks", Proceedings of the Seventh International Symposium on Symbolic and Numeric Algorithms for Scientific Computing.
78. Florian, R., V. 2007, "Reinforcement Learning Through Modulation of Spike-Timing-Dependent Synaptic Plasticity ", Neural Comp., vol. 19, pp. 1468-1502.
79. Seung, H.S. 2003, "Learning in Spiking Neural Networks by Reinforcement of Stochastic Synaptic Transmission", Neuron, vol. 40, no. 6, pp. 1063-1073.
80. De Queiroz, M.S., de Berrêdo, R.C. & de Pádua Braga, A. 2006, "Reinforcement learning of a simple control task using the spike response model", Neurocomputing, vol. 70, no. 1-3, pp. 14-20.
81. Alnajjar, F. & Murase, K. 2006, "Use-dependent synaptic connection modification in SNN generating autonomous behavior in a Khepera mobile robot", 2006 IEEE Conference on Robotics, Automation and Mechatronic, pp. 4018757.
82. Amin, H.H. & Fujii, R.H. 2005, "Sound classification and function approximation using spiking neural networks", Int Conf on Intelligent Computing, ICIC 2005 pp. 621.
83. Amin, H.H. & Fujii, R.H. 2005, "Learning algorithm for spiking neural networks", Proceedings, Part I Springer-Verlag, Changsha, China, pp. 456.
84. A Belatreche, LP Maguire, TM McGinnity, "Advances in Design and Application of Spiking Neural Networks" Springer Journal of Soft Computing - A Fusion of Foundations, Methodologies and Applications, pp. 239-248, Vol 11, No. 3, February 2007.
85. González-Nalda, P. & Cases, B. 2008, "Topos: Spiking neural networks for temporal pattern recognition in complex real sounds", Neurocomputing, vol. 71, no. 4-6, pp. 721-732.
86. Pavlidis, N.G.1., Tasoulis, O.K.1., Plagianakos, V.P.1., Nikiforidis, G. & Vrahatis, M.N. 2005, "Spiking neural network training using evolutionary algorithms", Proc IEEE Int Joint Conference on Neural Networks Montreal, Que., Canada, pp. 2190.

87. Johnston, S; Prasad, G., Maguire, LP & McGinnity, TM 2006, "A hybrid learning algorithm fusing STDP with GA based explicit delay learning for spiking neurons", 3rd Int IEEE Conference on Intelligent Systems, London, UK, pp. 621.
88. Barber, D. 2003, " Learning in spiking neural assemblies" in Advances in Neural Information Processing Systems 15 , eds. S. Thrun S. Becker & K. Obermayer, MIT press edn, Cambrige MA, , pp. 149-156.
89. Pfister, J., Barber, D. & Gerstner, W. 2003, "Optimal Hebbian Learning: A Probabilistic Point of View" in In ICANN/ICONIP 2003, volume 2714 of Lecture Notes in Computer Science, pp. 92-98.
90. Adibi, PL., Meybodi, MR. & Safabakhsh, R. 2005, "Unsupervised learning of synaptic delays based on learning automata in an RBF-like network of spiking neurons for data clustering", Neurocomputing, vol. 64, pp. 335-57.
91. Gerstner, W. & Kistler, W. 2002, "Mathematical formulations of Hebbian learning", Biological Cybernetics, vol. 87, pp. 404-415.
92. Kistler, W. 2002, "Spike-timing dependent synaptic plasticity: a phenomenological framework", Biological Cybernetics, vol. 87, pp. 416-427.
93. Bohte, S.M., Poutré, H.L. & Kok, J.N. 2002, "Unsupervised Clustering With Spiking Neurons By Sparse Temporal Coding and Multilayer RBF Networks", IEEE Trans. Neural Netw., vol. 13, no. 2, pp. 426-435.
94. Wade, J.J., McDaid, L.J., Santos, J.A., Sayers, H.M. 2008, “SWAT: An unsupervised SNN training algorithm for classification problems”, IEEE IJCNN 2008, pp.2648 - 2655
95. Wysoski, S., Gomes, Benuskova, L. & Kasabov, N. 2006, "On-line Learning with Structural Adaptation in a Network of Spiking Neurons for Visual Pattern Recognition", Artificial Neural Networks - ICANN 2006, 16th International Conference, pp. 61-70.
96. Soltic, S., Wysoski, S.G. & Kasabov, N.K. 2008 “Evolving spiking neural networks for taste recognition”, IEEE World Congress on Computational Intelligence, IJCNN 2008, pp.2091 – 2097.
97. Alnajjar, F., Bin Mohd Zin, I., Murase, K. 2008, “A Spiking Neural Network with dynamic memory for a real autonomous mobile robot in dynamic environment”, IEEE World Congress on Computational Intelligence, pp.2207 – 2213.
98. Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. Neural Networks, Elsevier, 10, 1659-1671.
99. Glackin, C; McDaid, L; Maguire, LP; Sayers, H: “Implementing Fuzzy Reasoning on a Spiking Neural Network,” Artificial Neural Networks: 18th International Conference, Prague, Czech Republic, Part II, 2008.
100. Izhikevich, E., 2004. Which model to use for cortical spiking neurons? IEEE Transactions on Neural Networks 15 (5), 1063–1070.