Evolving spiking neural networks: A Survey

Schliebs, S; Kasabov, N

Abstract: This paper provides a comprehensive literature survey on the evolving Spiking Neural Network (eSNN) architecture since its introduction in 2006 as a further extension of the ECoS paradigm introduced by Kasabov in 1998. We summarize the functioning of the method, discuss several of its extensions and present a number of applications in which the eSNN method was employed. We focus especially on some proposed extensions that allow the processing of spatio-temporal data and for feature and parameter optimisation of eSNN models to achieve better accuracy on classification/prediction problems and to facilitate new knowledge discovery. Finally, some open problems are discussed and future directions highlighted.

DOI: https://doi.org/10.1007/s12530-013-9074-9

Evolving spiking neural network – A survey

Stefan Schliebs · **Nikola Kasabov**

Received: date / Accepted: date

Abstract This paper provides a comprehensive literature survey on the evolution of the evolving Spiking Neural Network (eSNN) architecture since its introduction in 2006 as a further extension of the ECOS paradigm introduced by Kasabov in 1998. We summarize the functioning of the method, discuss several of its extensions and present a number of applications in which the eSNN method was employed. We focus especially on some proposed extensions that allow the processing of spatio-temporal data and for feature and parameter optimisation of eSNN models to achieve better accuracy on classification/prediction problems and to facilitate new knowledge discovery. Finally, some open problems are discusses and future directions highlighted.

Keywords Evolving Spiking Neural Network · QiSNN · eSNN-DQiPSO · reSNN · deSNN

1 Introduction

Evolving connectionist systems (ECoS) are modular connectionist-based systems that evolve their structure and functionality in a continuous, self-organised, on-line, adaptive, interactive way from incoming information [28–32, 34, 41, 52, 53]. They can process both data and knowledge in a supervised and/or unsupervised way. They can learn incrementally single data items or chunks of data and also incrementally change their input features [52, 53]. Elements of ECOS have been proposed as part of the classical NN models, such as SOM, RBF, FuzyARTMap, Growing neural gas, neuro-fuzzy systems, RAN (see [3, 7, 16, 19, 43, 46, 55]. Other ECOS models, along with their applications, have been reported in [34, 1, 2, 23, 70, 80]. Here we will briefly illustrate the concepts of ECOS on two implementations, EFuNN [30] and DENFIS [41] (see also [28, 29, 31, 32, 34, 52, 53].

ECOS learn local models from data through clustering of the data and associating a local output function for each cluster represented in a connectionist structure. Clusters of data are created based on similarity between data samples either in the input space (this is the case in some of the ECoS models, *e.g.* the dynamic neuro-fuzzy inference system DENFIS [32, 34, 41], or in both the input and output space (this is the case *e.g.* in the EFuNN models [30–32, 34]. Samples that have a distance to an existing node (cluster center, rule node) less than a certain threshold are allocated to the same cluster. Samples that do not fit into existing clusters, form new clusters. Cluster centers are continuously adjusted according to new data samples, and new clusters are created incrementally.

ECOS learn from data and automatically create or update a local output function for each cluster, the function being represented in the connection weights, thus creating local models. Different functionalities of ECOS have been introduced and studied in [30–32, 34, 41] such as: on-line or off-line neuron aggregation and pruning; "sleep"-learning; fuzzy rule extraction and rule adaptation (see also [33]); using SVM as local models [44]; evolutionary optimisation of features and parameters of ECOS [33]; and others. Applications of ECOS span across domain areas [32, 34], *e.g.* bioinformatics (see also [35]); speech and image processing; multimodal audio-visual information processing; ecological modelling; robot control (see also [24]); personalised modelling [69, 40]; neuroinformatics and brain study. Software environment Neucom has been developed to include some

Stefan Schliebs
Auckland University of Technology
Tel.: +64-9-921 8427
E-mail: sschlieb@aut.ac.nz

Nikola Kasabov
KEDRI, Auckland University of Technology, New Zealand,
Institute for Neuroinformatics, ETH/UZH Zurich, Switzerland
E-mail: nkasabov@aut.ac.nz

of the ECOS methods[1]. A detailed survey on ECOS can be found in [80].

While the classical ECOS use a simple McCulloch and Pitts model of a neuron, the further developed evolving spiking neural network (eSNN) architecture uses a spiking neuron model, but same or similar ECOS principles and applications are applicable. How this is done is revealed in the next sections of the paper.

2 Evolving Spiking Neural Network

Based on the ECoS methodology, an evolving spiking neural network architecture (eSNN) was proposed in [88, 34] which was initially designed as a visual pattern recognition system. Other studies have utilised eSNN as a general classification method, *e.g.* in the context of classifying water and wine samples [68]. The first eSNNs were based on the Thorpe's neural model [76], in which the importance of early spikes (after the onset of a certain stimulus) is boosted called rank-order coding and learning. Synaptic plasticity is employed by a fast supervised one-pass learning algorithm that is explained as part of this section. Following eSNN architectures used both rank-order and time-based learning methods to account for spatio-temporal data [39, 51].

In the next sections, Thorpe's neural model and the spike encoding principle used in eSNN are presented, followed by the description of the one-pass learning method and the overall functioning of the eSNN method. Finally, a variety of applications based on the eSNN architecture is reviewed and summarised. A number of open problems that have not been solved in the existing literature, have motivated this study and are outlined at the end of this review.

2.1 Neural model

A simplified LIF model was formally proposed in [76]. However, the general idea of the model can be traced back to publications as early as 1990, *cf.* [71]. This model lacks the post-synaptic potential leakage. The spike response of a neuron depends only on the arrival time of pre-synaptic spikes. The importance of early spikes is boosted and affects the post-synaptic potential more strongly than later spikes. This concept is very interesting due to the fact that the brain is able to compute even complex tasks quickly and reliably. For example, the human brain requires for the processing of visual data only approximately 150ms [72, 74], see also a similar study on rapid visual categorisation of natural and artificial objects [79]. Since it is known that this type of computation is partly sequential and several parts of the brain involving millions of neurons participate in the computation,

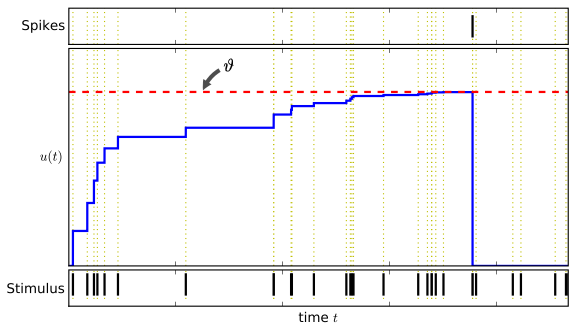

Fig. 1: Evolution of the post-synaptic potential (PSP) of the Thorpe neuronal model for a given input stimulus. If the potential reaches threshold ϑ, a spike is triggered and the PSP is set to 0 for the rest of the simulation, even if the neuron is still stimulated by incoming spike trains.

it has been argued in [75] and [72] that each neuron has time and energy to emit only very few spikes that can actually contribute to the processing of the input. As a consequence, few spikes per neuron are biologically sufficient to solve a highly complex recognition task in real time.

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

$$u_i(t) = \begin{cases} 0 & \text{if fired} \\ \sum\limits_{j|f(j)<t} w_{ji}\, m_i^{order(j)} & \text{else} \end{cases} \quad (1)$$

where w_{ji} is the weight of a pre-synaptic neuron j, $f(j)$ is the firing time of j, and $0 < m_i < 1$ is a parameter of the model, namely the modulation factor. Function $order(j)$ represents the rank of the spike emitted by neuron j. For example, a rank $order(j) = 0$ would be assigned if neuron j is the first among all pre-synaptic neurons of i that emits a spike. In a similar fashion, the spikes of all pre-synaptic neurons are ranked and then used in the computation of u_i. A neuron i fires a spike when its potential reaches a certain threshold ϑ. After emitting a spike, the potential resets to $u_i = 0$. Each neuron is allowed to emit only a single spike at most. The threshold $\vartheta = c\, u_{max}$ is set to a fraction $0 < c < 1$ of the maximum potential u_{max} reachable for a neuron. Figure 1 presents the change of the post-synaptic potential for the Thorpe neural model if a series of input spikes stimulates the neuron through different synapses.

These simplifications allow a very fast real-time simulation of large networks. Due to its low computational costs this model was mainly used for studying image and speech recognition methods involving thousands of connected neurons (*cf. e.g.* [14] and [77]). Many studies have investigated the Thorpe model, *e.g.* for face recognition [78] and [12].

[1] `http://www.theneucom.com`

2.2 Neural encoding

In order to classify real-valued data sets, each data sample, *i.e.* a vector of real-valued elements, is mapped into a sequence of spikes using a certain neural encoding technique. In the context of eSNN, the so-called rank order population encoding is employed, but other encoding may be suitable as well.

Rank order population encoding is an extension of the rank order encoding introduced in [76]. It allows the mapping of vectors of real-valued elements into a sequence of spikes. An implementation based on arrays of receptive fields is firstly described in [6]. Receptive fields allow the encoding of continuous values by using a collection of neurons with overlapping sensitivity profiles. Each input variable is encoded independently by a group of M one-dimensional receptive fields. For a variable n an interval $[I^n_{min}, I^n_{max}]$ is defined. The Gaussian receptive field of neuron i is given by its centre μ_i

$$\mu_i = I^n_{min} + \frac{2i-3}{2} \cdot \frac{I^n_{max} - I^n_{min}}{M-2} \qquad (2)$$

and width σ:

$$\sigma = \frac{1}{\beta} \cdot \frac{I^n_{max} - I^n_{min}}{M-2} \qquad (3)$$

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

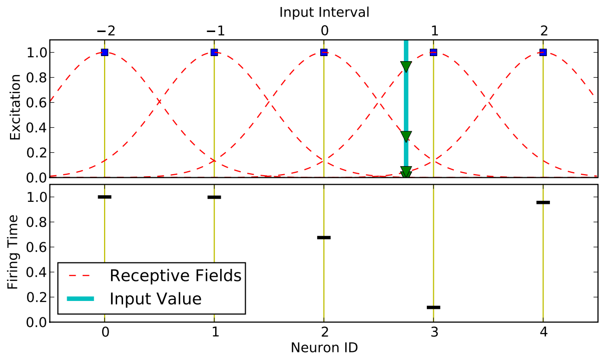

Fig. 2: Population encoding based on Gaussian receptive fields. For an input value $v = 0.75$ (thick straight line in top figure) the intersection points with each Gaussian is computed (triangles), which are in turn translated into spike time delays (lower left figure).

More information on rank order coding strategies can be found in [54] and the accompanying article [12]. Very interesting is also the review on rapid spike-based processing strategies in the context of image recognition presented in [73], where most work on the Thorpe neural model and rank order coding is summarised. Rank order coding was also explored for speech recognition problems [47] and is a core part of the eSNN architecture.

2.3 Learning

The topology of eSNN is strictly feed-forward and may be organised in several layers. Weight modification only occurs on the connections between the neurons of the output layer and the neurons of either hidden layer or the input layer.

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

Algorithm 1 Training an evolving spiking neural network (eSNN)

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

The learning algorithm successively creates a repository of trained output neurons during the presentation of training samples. For each class label $l \in L$ an individual repository is evolved. The procedure is described in detail in Algorithm 1. For each training sample i with class label $l \in L$ a new output neuron is created and fully connected to the previous layer of neurons resulting in a real-valued weight vector $w^{(i)}$, with $w_j^{(i)} \in \mathbb{R}$ denoting the connection between the pre-synaptic neuron j and the created neuron i. In the next step, the input spikes are propagated through the network and the value of weight $w_j^{(i)}$ is computed according to the *order* of spike transmission through a synapse j, *cf.* line 3 in Algorithm 1:

$$w_j^{(i)} = (m_l)^{order(j)}, \quad \forall\, j \mid j \text{ pre-synaptic neuron of } i \quad (4)$$

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

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

$$\vartheta^{(i)} = c_l u_{max}^{(i)} \quad (5)$$
$$u_{max}^{(i)} = \sum_j w_j^{(i)} (m_l)^{order(j)} \quad (6)$$

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

The weight vector of the trained neuron is then compared to the ones of neurons that are already stored neurons in the repository, *cf.* line 6 in Algorithm 1. If the minimal Euclidean distance between the weight vectors of the neuron i and an existing neuron k is smaller than a specified similarity threshold s_l, the two neurons are considered too “similar” and both the firing thresholds and the weight vectors are merged according to:

$$w_j^{(k)} \leftarrow \frac{w_j^{(i)} + N w_j^{(k)}}{1+N}, \forall j \mid j \text{ pre-synaptic neuron of } i \quad (7)$$
$$\vartheta^{(k)} \leftarrow \frac{\vartheta^{(i)} + N \vartheta^{(k)}}{1+N} \quad (8)$$

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

Figure 3 depicts the eSNN architecture. Due to the incremental evolution of output neurons, it is possible to accumulate knowledge as it becomes available. Hence, a trained network is able to learn new data without the need of re-training on already learnt samples as it is in all ECOS models.

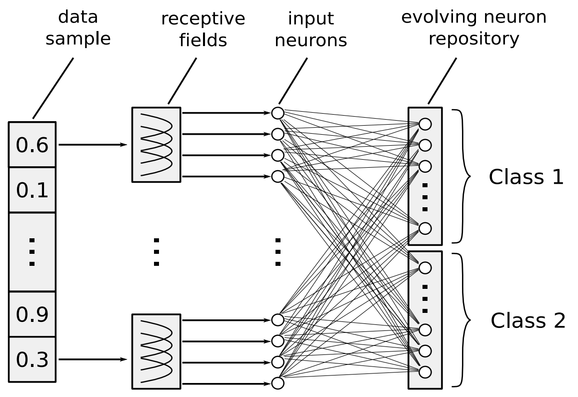

Fig. 3: Schematic illustration of the evolving spiking neural network architecture (eSNN). Real-valued vector elements are mapped into the time domain using rank order population encoding based on Gaussian receptive fields. As a consequence of this transformation input neurons emit spikes at pre-defined firing times, invoking the one-pass learning algorithm of the eSNN. The learning iteratively creates repositories of output neurons, one repository for each class. Here a two-class problem is presented. Due to the evolving nature of the network, it is possible to accumulate knowledge as it becomes available, without the requirement of re-training with already learnt samples, which is one of the ECOS principles.

2.4 Feature selection and parameter optimization

A number of studies have extended the eSNN architecture in which a mechanism of automatically optimizing the features of the given dataset along with most of the eSNN parameters was proposed. In the next sections, we survey the Quantum-inspired eSNN framework QiSNN [59] and the Dynamic Quantum-inspired Particle Swarm Optimized eSNN method eSNN-DQiPSO [21].

2.4.1 QiSNN

With encouraging results eSNN was presented in the context of a feature subset selection (FSS) problem [59]. In this work, a binary optimization algorithm, namely the Versatile Quantum-inspired Evolutionary Algorithm (vQEA) [9], was combined with eSNN. Through implementing quantum principles, vQEA evolves in parallel a number of independent probability vectors, that may interact at certain intervals with each other, forming a multi-model Estimation of Distribution Algorithm (EDA) [10]. Following the wrapper approach [42], vQEA was used to identify relevant feature subsets and simultaneously evolve an optimal eSNN parameter setting. Due to the quantum metaphor employed in vQEA, the architecture was referred to as the Quantum-inspired SNN (QiSNN) framework. Applied to benchmark data, the QiSNN-based feature selection reported excellent classification results and an accurate detection of relevant information in the data set [59–63].

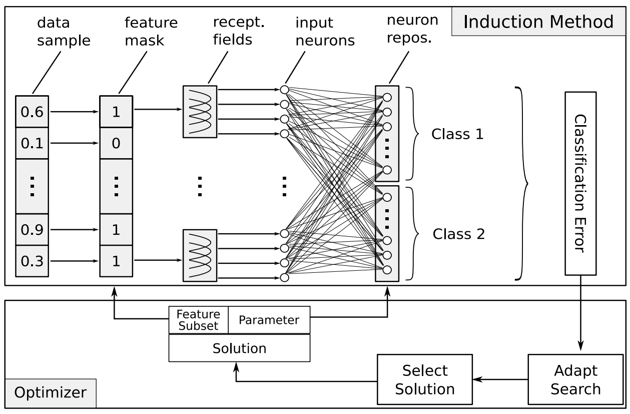

Fig. 4: The QiSNN framework of eSNN with tightly coupled feature selection and parameter optimization, integrated with the data. As a first step a feature subset is selected from a real-valued data sample using a bit string acting as a feature mask, where a “0” (“1”) in this mask indicates (non-)selected features of the data vector. Selected vector elements are then mapped into the time domain using a number of Gaussian receptive fields. Based on this transformation input neurons of a eSNN emit spikes at predefined firing times, invoking the one-pass learning algorithm of the eSNN. The learning iteratively creates repositories of output neurons, one repository for each class. Here a two-class problem is presented. Based on a set of training samples the eSNN is trained and its quality is determined based on the classification accuracy on a set of testing samples. The classification accuracy is then used as the fitness criterion of the optimization method. Based on the fitness the search strategy is adapted and a new solution is proposed. The solution includes two parts: A binary feature mask and a set of real-valued parameters for eSNN. The whole process iterates until a termination criterion is met, *i.e.* a satisfying classification accuracy is reached or a the maximum number of iterations is exhausted.

The QiSNN framework is shown in Figure 4. The upper part of the diagram represents the eSNN classification method. Note the binary mask in the second step of the process. This mask along with a specific configuration of neural and learning parameters is passed to eSNN from the optimization method depicted in the lower part of the figure. The binary mask describes the features to be selected from a real-valued input data vector. Then, the selected features are transformed into a train of spikes using the rank order population encoding technique, see Section 2.2 for details. Following the one-pass learning procedure, the connection weights of eSNN are trained according to the given parameter set. The learning process includes the presentation of all training samples. After the learning, the classification accuracy is determined on a set of test samples. This accuracy provides a quality measure of the feature subset and the used parameter configuration. This quality feedback is passed to the employed optimization algorithm. Based on the quality, the optimizer adapts the search strategy and passes new feature subsets and configurations to eSNN for evaluation. The whole process iterates until a termination criterion is met, *i.e.* a satisfying classification accuracy is reached or a the maximum number of iterations is exhausted.

The automatic parameter adaptation of QiSNN is a highly desirable feature since it promotes the straightforward application of the method to other problem domains. In section 4, we will present some of the applications in which QiSNN was employed for knowledge discovery.

For the simultaneous optimization of the network parameters and the feature subset, a heterogeneous search space has to be explored. A binary search space encodes the (non-)selection of specific features and a continuous search space represents the parameter space of eSNN. Therefore, the binary nature of vQEA requires the conversion of bit strings into real values which are then mapped into eSNN parameters. The use of a binary optimiser for a real-valued search space appears inappropriate for a number of reasons.

First, for the mapping of bit strings into a real value additional computational resources are necessary. Furthermore, a granularity is introduced into a continuous search interval. Even worse, neighbouring solutions in the continuous domain might not be neighbours in their binary representation. Thus, exploring the local neighbourhood of a real-valued solution may require the optimiser to flip many bits in the solution’s binary representation encouraging premature convergence and promoting the well-known phenomenon of hitch-hiking. For these reasons vQEA was extended towards continuous search spaces in [61] resulting in a novel Estimation of Distribution Algorithm (EDA) called the hierarchical multi-model EDA (hMM-EDA). hMM-EDA uses a continuous representation based on Gaussian distributions to optimize the parameter space of eSNN. The study presented in [63] describes the algorithm in detail and compares its performance and its characteristics to related methods.

Similar to QiSNN, the combination of hMM-EDA and eSNN forms an integrated feature and parameter optimization framework based on the eSNN classification method. The continuous representation in hMM-EDA is used to optimize the parameter space of eSNN, while the binary representation explores the feature space of the given data set. The enhanced QiSNN was shown to converge faster, more consistently and more reliably than the original QiSNN [61, 63, 58].

In [60] presented some detailed experimental analysis of the behavior and functioning of QiSNN. The analysis focused on the process of the simultaneous optimization of features subsets and eSNN parameters and their interaction and mutual influences. The study also investigate the role of the neural encoding and its impact on the classification characteristics of QiSNN. Recommendations for configuring eSNN parameters that are not included in the evolutionary optimization process were given.

2.4.2 eSNN-DQiPSO

An approach very similar to QiSNN was presented in [20]. Instead of using hMM-EDA as employed in QiSNN, a Quantum-inspired Particle Swarm Optimizer (QiPSO) was investigated to perform both feature selection and parameter optimization of eSNN. Since QiPSO was designed as a binary optimization method, the algorithm reported optimization results comparable to the ones obtained by the original QiSNN that employed the binary optimizer vQEA. A future study should attempt a formal comparison of these two QiSNN variants.

Later a heterogeneous version of QiPSO was proposed in [21,22], namely the Dynamic QiPSO, that was able to explore both continuous and binary search spaces simultaneously. The method essentially combined QiPSO with a traditional PSO to explore the two search spaces in parallel. Additionally, some specialized particles were introduced in order to increase the diversity of the particle population thus counteracting premature convergence of the algorithm.

The same study also proposed another interesting variation of the eSNN classifier. Following ideas of a probabilistic neural model introduced in [36], the probability of presence and absence of connections between input and output neurons of eSNN were also subjected to the evolutionary optimization process. The resulting probabilistic eSNN-DQiPSO was studied on the two-spiral problem and reported superior classification performance compared to the deterministic eSNN.

2.5 Rule extraction from eSNN

In ECOS models each cluster of training data is represented by a cluster center captured in a (hidden) neuron called rule node, and a local function to approximate the data in the cluster. The cluster information and the local function are represented as fuzzy rules of the form:

IF x_1 is HIGH, i.e. data is in cluster C_i
THEN the function is F_i

where $x_1, \ldots$ are input variables. Similarly, fuzzy rules can be extracted from a trained eSNN as suggested in [67], where the output function is a class label.

3 Spatio-temporal pattern recognition

Many of today's data volumes are continuously updated adding an additional time dimension to the data sets. The classification of spatio-temporal patterns is a great challenge for data mining methods. Many data vectors are sequentially presented to a classification algorithm which in turn learns the mapping of this sequence to a given class label. In its original form, the first eSNN model does not allow the classification of spatio-temporal data.

3.1 eSNN with dynamic synapses (deSNN)

Several new models of eSNN have been recently proposed to deal with spatio and spectro-temporal pattern recognition (SSTPR), from a single neuron layer model, to reservoir-based models. In [39,51] dynamic synapses are added to the LIF model, where in addition to the rank-order learning based on the order of incoming spikes, the synaptic weights change in an unsupervised mode based on the time of the following incoming spikes to the post-synaptic neuron [17]. This model is called dynamic eSNN (deSNN). It is efficient in both address event representation (AER) and frame-based representation. In the former, only changes in the input information are presented as incoming spikes, asynchronously (*e.g.* [39]). In the latter whole frames of input data are transferred into spikes one by one as a temporal sequence (*e.g.* [51]).

When the temporal input patterns are longer (*e.g.* lasting for seconds) a single neuron layer structure may not be efficient in learning these sequences. In this case larger neuronal "reservoirs" would be needed.

3.2 Reservoir-based eSNN (reSNN)

A reservoir extension of the standard LIF eSNN was proposed in [66] that enables the method to process spatio-temporal information. The general idea is to add an additional layer to the network architecture that transforms the spatio-temporal input pattern into a single high-dimensional network state. The mapping of this intermediate state into a desired class label can then be learned by the one-pass learning algorithm of eSNN. The study presented an initial experimental analysis demonstrating the feasibility of this extension in principle and left a more comprehensive analysis for future studies. The idea was further developed and investigated in [65][50] resulting in the reservoir based eSNN (reSNN) illustrated in Figure 5. Compared to the original eSNN, the novel parts in the architecture are indicated by the highlighted boxes.

We outline the working of the method by explaining the diagram from left to right. Spatio-temporal data patterns are

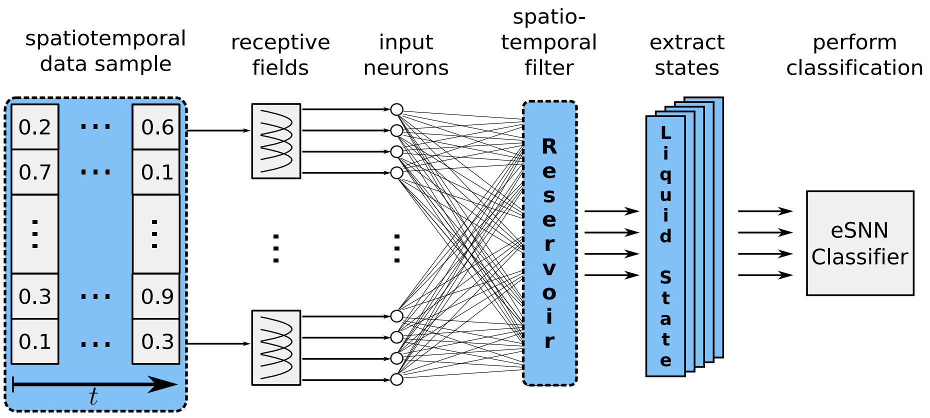

Fig. 5: Architecture of the reservoir-based eSNN capable of processing spatio-temporal data. The colored (dashed) boxes indicate novel parts in the original eSNN architecture.

presented to the reSNN system in form of an ordered sequence of real-valued data vectors. In the first step, each real-value of a data vector is transformed into a spike train using a population encoding already described in section 2.2. As a result of the encoding, input neurons spike at predefined times according to the presented data vectors. The input spike trains are then fed into a spatio-temporal filter which accumulates the temporal information of all input signals into a single high-dimensional intermediate liquid state. The filter is implemented in form of a liquid or a reservoir [48], *i.e.* a recurrent SNN, for which the eSNN acts as a readout function. Since the intermediate liquid state has no temporal dimension, the one-pass learning algorithm of eSNN is able to learn the mapping of this state into a desired class label.

The study in [65] investigated the classification performance of reSNN on the real-world sign-language dataset LIBRAS. reSNN reported very promising classification accuracies which were at least comparable to results obtained by traditional neural network approaches.

The reservoir is constructed of Leaky Integrate-and-Fire (LIF) neurons with exponential synaptic currents. This neural model is based on the idea of an electrical circuit containing a capacitor with capacitance C and a resistor with a resistance R, where both C and R are assumed to be constant. The dynamics of a neuron i are then described by the following differential equations:

$$\tau_m \frac{du_i}{dt} = -u_i(t) + R\, I_i^{\text{syn}}(t) \tag{9}$$

$$\tau_s \frac{dI_i^{\text{syn}}}{dt} = -I_i^{\text{syn}}(t) \tag{10}$$

The constant $\tau_m = RC$ is called the membrane time constant of the neuron. Whenever the membrane potential u_i crosses a threshold ϑ from below, the neuron fires a spike and its potential is reset to a reset potential u_r. We use an exponential synaptic current I_i^{syn} for a neuron i modeled by Eq. 10 with τ_s being a synaptic time constant.

In our experiments we construct a liquid having a small-world inter-connectivity pattern as described in [48]. A recurrent SNN is generated by aligning 100 neurons in a three-dimensional grid of size $4 \times 5 \times 5$. Two neurons A and B in this grid are connected with a connection probability

$$P(A,B) = C \times e^{\frac{-d(A,B)}{\lambda^2}} \tag{11}$$

where $d(A,B)$ denotes the Euclidean distance between two neurons and λ corresponds to the density of connections which was set to $\lambda = 2$ in all simulations. Parameter C depends on the type of the neurons. We discriminate into excitatory (ex) and inhibitory (inh) neurons resulting in the following parameters for C: $C_{ex-ex} = 0.3$, $C_{ex-inh} = 0.2$, $C_{inh-ex} = 0.5$ and $C_{inh-inh} = 0.1$. The network contained 80% excitatory and 20% inhibitory neurons. The connections weights were randomly selected by a uniform distribution and scaled in the interval $[-8, 8]$nA. The neural parameters were set to $\tau_m = 30$ms, $\tau_s = 10$ms, $\vartheta = 5$mV, $u_r = 0$mV. Furthermore, a refractory period of 5ms and a synaptic transmission delay of 1ms was used.

Using this configuration, the recorded liquid states did not exhibit the undesired behavior of over-stratification and pathological synchrony – effects that are common for randomly generated liquids [50]. For the simulation of the reservoir we used the SNN simulator Brian [18].

4 Applications

The eSNN architectures described above have been used in a variety of applications that are briefly summarised here.

4.1 Visual pattern recognition

Among the earliest application of the eSNN is the visual pattern recognition system presented in [88] which extends the work of [11, 13] by including the on-line learning technique described above. In [88] and [83] the method was studied on an image data set consisting of 400 faces of 40 different persons. The task here was to predict the class labels of presented images correctly. The system was trained on a subset of the data and then tested on the remaining samples of the data. Classification results were similar to [11, 13] with the additional advantages of the novel on-line learning method.

In a later study another processing layer was added to the system which allows efficient multi-view visual pattern recognition [86]. The additional layer accumulates information over several different views of an image in order to reach a final decision about the associated class label of the frames. Thus, it is possible to perform an efficient on-line person authentication through the presentation of a short video clip to the system, although the audio information was ignored in this study.

The main principle of this image recognition method is briefly outlined here. The neural network is composed of four layers of Thorpe LIF neurons, each of them grouping a set of neurons into several two-dimensional maps, so-called neural maps. Information in this network is propagated in a feed-forward manner, *i.e.* no recurrent connections exist. An input frame in form of a grey-scale image is fed into the first neural layer (L_1), each pixel of the image corresponding to one neuron in a neural map of L_1. Several neural maps may exist in this layer. The map consists of "On" and "Off" neurons that are responsible for the enhancement of the high contrast parts of the image. Each map is configured differently and thus is sensitive to different grey scales in the image. The output of this layer is transformed into the spike domain using rank order encoding as described in [76]. As a consequence of this encoding, pixels with higher contrast are prioritised in the neural processing.

The second layer, denoted L_2, consists of orientation maps. Each map is selective for different directions, *e.g.* $0°, 45°, \ldots, 315°$, and is implemented by appropriately parametrised Gabor functions. It is noted that the first two layers are passive filters that are not subject to any learning process. In the third layer, L_3, the learning occurs using the one-pass learning method described in section 2.3. Here neural maps are created and merged according to the rules of the learning algorithm. Finally, the fourth layer, L_4, consists of a single neuron for each output class, which accumulates opinions about the class label of a certain sequence of input frames. The weights between L_3 and L_4 are fixed to a constant value, usually 1, and are not subject to learning. The first L_4 neuron that is activated by the presented stimuli determines the classification result for the input. After the activation of an L_4 neuron the system stops.

Experimental evidence about the suitability of this pattern recognition system is provided in [86] along with a comparison to other typical classification methods.

4.2 Auditory pattern recognition

A similar network, but in an entirely different context, was investigated in [84], where a text-independent speaker authentication system is presented. The classification task in this work consisted of the correct labelling of audio streams presented to the system.

Speech signals are split into temporal frames, each containing a signal segment over a short time period. The frames are first pre-processed using the Mel Frequency Cepstrum Coefficients (MFCC) [56] and then used to invoke the eSNN. The MFCC frame is transformed into the spike domain using rank order encoding [76] and the resulting stimulus is propagated to the first layer of neurons. This layer, denoted L_1, contains two neural ensembles representing the speaker and the background model, respectively. While the former model is trained on the voice of a certain speaker, the latter one is trained on the background noise of the audio stream. This system also collects opinions about the class label of the presented sequence of input frames, which is implemented by the second layer of the network. Layer L_2 consists of only two neurons, each of which accumulates information about whether a given frame corresponds to a certain speaker or to the background noise. Whenever an L_2 neuron is activated, the simulation of the network stops and the classification output is presented.

4.3 Audio-visual pattern recognition

The two recognition systems presented above were successfully combined, forming an audio-visual pattern recognition method. Both systems are trained individually, but their output is propagated to an additional supra-modal layer. The supra-modal layer integrates incoming sensory information from individual modalities and cross-modal connections enable the influence of one modality upon the other. A detailed discussion of this system along with experimental evidence is given in [85] and in the PhD dissertation of Simei Wysoski in [82]. A later study suggests that this system might be suitable not only for audio and visual pattern recognition problems, but also for more general brain-like multimodal information processing tasks [81, 87].

4.4 Taste recognition

Another application of eSNN being discussed here investigates the use of a SNN for taste recognition in a gustatory model. The classification performance of eSNN was experimentally explored [68, 67] based on water and wine samples collected from [8] and [57]. The topology of the model consists of two layers. The first layer receives an input stimulus obtained from the mapping of a real-valued sensory data sample into spike trains using a rank order population encoding, *cf.* section 2.2. The weights from the first neural layer are subject to training according to the already discussed one-pass learning method. Finally, the output of the second neural layer determines the class label of the presented input stimulus.

The method was investigated in a number of scenarios, where the size of the data sets and the number of class labels was varied. Generally, eSNN reported promising results on both large and small data sets, which has motivated an FPGA hardware implementation of the system [89].

4.5 Ecological modeling

The QiSNN framework as presented in section 2.4 was used in a case study on ecological modeling [62]. Meteorological data, such as monthly and seasonal temperature, rain fall and soil moisture recordings for different geographical sites, were compiled from published results, and each global site was labelled according to the presence or absence of the Mediterranean fruit-fly (a serious fruit pest). The study aimed towards the identification of important features relevant for predicting the presence/absence of this insect species. From 68 features, QiSNN identified 14 features to be of particular importance. These features were analyzed by an ecological expert and they compared well to results found in related studies. The clear potential for further improvement of classification accuracy with model refinement, as well as automatic optimisation of parameters, suggested QiSNN as an useful approach for the analysis and modelling of complex, noisy ecological data.

The same data was later revisited [61] and the experiments were repeated using the enhanced version of QiSNN (see section 2.4 for details). Compared to the original study, the number of features could be further decreased (down to nine relevant features) without affecting the overall classification accuracy. Related wrapper-based feature selection methods employing the same optimizer, *i.e.* vQEA, but a different classification algorithm, *e.g.* the Naïve Bayesian Classifier, identified significantly more features to be relevant on the same dataset.

4.6 Sign language recognition

The reSNN method discussed in section 3 is suitable to address spatio-temporal pattern recognition problems. In [65], this method was investigated using a real-world data set called LIBRAS [15]. LIBRAS is the acronym for **LI**ngua **BRA**sileira de **S**inais, which is the official Brazilian sign language. The data contains 15 hand movements (signs) to be learned and classified. The movements were obtained from recorded video of four different people performing the movements in two sessions. In total 360 videos have been recorded, each video showing one movement lasting for about seven seconds. From the videos 45 frames uniformly distributed over the seven seconds have then been extracted. In each frame, the centroid pixels of the hand are used to determine the movement. All samples have been organized in ten sub-datasets, each representing a different classification scenario. More comprehensive details about the dataset can be found in [15]. The data can be obtained from the UCI machine learning repository.

As shown in [65], the classification performance of reSNN depends strongly on the employed readout method extracting the reservoir or liquid state at pre-defined time intervals. Using an analog readout technique, a good overall testing classification accuracy of around 82% was reported on this dataset. Additional optimization of the eSNN parameters and the features of the data using the DQiPSO algorithm further increased this accuracy to around 89% which compares very favourably to the results reported in [15] where a traditional Multilayer Perceptron was used.

4.7 Object movement recognition

In [39], the deSNN model was applied for moving object recognition using AER taken from a silicon retina camera [45]. The case study data created aimed at investigating the efficiency of the method for collision avoidance prediction when fast objects are moving towards the observer.

4.8 EEG spatio/spectro temporal pattern recognition

In [51], several eSNN architectures were investigated for the recognition of four EEG patterns, recorded when a subject is listening a sound, seeing an image, both, or no stimulus is presented. The best accuracy was obtained with the use of the deSNN when compared with the reSNN.

5 Summary and future directions

In this paper, we have surveyed the recent developments on the eSNN classification method. We reviewed algorithmic details on the learning algorithm, on the neural model and on the neural encoding. Numerous models were discussed and many applications and case studies in which the eSNN was successfully employed were presented.

The two models deSNN and reSNN, designed towards the processing of spatio-temporal real world data, appear especially interesting. Future studies will focus on the tighter integration of reservoir based techniques with the "evolving philosophy" of eSNN. Especially the automatic configuration of the reservoir represents a highly desirable goal. The large number of parameters in the reservoir is clearly a drawback of the presented reSNN despite the promising results that have been obtained with this method. Initial studies aiming for a better controlled reservoir have suggested some simple mechanisms adjusting the overall neural activity of recurrent SNN based on the strength of the input stimuli [64]. Some significantly improved separation capabilities of these adaptive reservoirs were reported based on synthetic datasets. Future studies are planned in which these findings are applied to the EEG based prediction of epileptic seizures.

Other directions involve the neurogenetic modeling approaches firstly introduced in [38, 4, 5]. The idea is to adapt the parameters of a SNN using an additional gene regulatory network which has to be optimized itself in order to achieve a desired network behavior of the SNN. Other studies have recently investigated this direction in the context of robotic controllers [49], cognitive systems [34] and Alzheimer disease models [27]. A combination of the fast evolving one-pass learning of eSNN with the automatic parameter adaptation of a genetically controlled recurrent SNN could open the path for numerous relevant real-world applications including neuromorphic computation systems [25, 26, 37].

References

1. Angelov, P., Filev, D., Kasabov, N.: Guest editorial evolving fuzzy systems – preface to the special section. Fuzzy Systems, IEEE Transactions on **16**(6), 1390–1392 (2008). DOI 10.1109/TFUZZ.2008.2006743
2. Angelov, P., Filev, D., Kasabov, N. (eds.): Evolving intelligent systems: methodology and applications. Wiley (2010)
3. Arbib, M. (ed.): The Handbook of Brain Theory and Neural Networks, 2nd edn. MIT Press, Cambridge, MA (2003)
4. Benuskova, L., Jain, V., Wysoski, S.G., Kasabov, N.: Computational neurogenetic modeling: a pathway to new discoveries in genetic neuroscience. Intl. Journal of Neural Systems **16**(3), 215–227 (2006)
5. Benuskova, L., Kasabov, N.: Computational Neurogenetic Modelling. Springer, NY (2007)
6. Bohte, S.M., Kok, J.N., Poutré, J.A.L.: Error-backpropagation in temporally encoded networks of spiking neurons. Neurocomputing **48**(1-4), 17–37 (2002)
7. Carpenter, G., Grossberg, S., Markuzon, N., Reynolds, J., Rosen, D.: Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analogue multi-dimensional maps. IEEE Trans. of Neural Networks **3**(5), 698–713 (1991)
8. de Sousa, H.C., Riul Jr., A.: Using MLP networks to classify red wines and water readings of an electronic tongue. Neural Networks, Brazilian Symposium on **0**, 13 (2002). DOI 10.1109/SBRN.2002.1181428
9. Defoin-Platel, M., Schliebs, S., Kasabov, N.: A versatile quantum-inspired evolutionary algorithm. In: IEEE Congress on Evolutionary Computation, CEC'07, pp. 423–430. IEEE Press, Singapore (2007)
10. Defoin-Platel, M., Schliebs, S., Kasabov, N.: Quantum-inspired evolutionary algorithm: A multimodel EDA. Evolutionary Computation, IEEE Transactions on **13**(6), 1218–1232 (2009). DOI 10.1109/TEVC.2008.2003010
11. Delorme, A., Gautrais, J., VanRullen, R., Thorpe, S.: SpikeNET: A simulator for modeling large networks of integrate and fire neurons (1999). URL `citeseer.ist.psu.edu/delorme99spikenet.html`
12. Delorme, A., Perrinet, L., Thorpe, S.J.: Networks of integrate-and-fire neurons using rank order coding B: Spike timing dependent plasticity and emergence of orientation selectivity. Neurocomputing **38-40**, 539–545 (2001)
13. Delorme, A., Thorpe, S.J.: Face identification using one spike per neuron: resistance to image degradations. Neural Networks **14**(6-7), 795–803 (2001)
14. Delorme, A., Thorpe, S.J.: SpikeNET: an event-driven simulation package for modelling large networks of spiking neurons. Network: Computation in Neural Systems **14**, 613–627 (2003). DOI 10.1088/0954-898X/14/4/301
15. Dias, D., Madeo, R., Rocha, T., Biscaro, H., Peres, S.: Hand movement recognition for brazilian sign language: A study using distance-based neural networks. In: Neural Networks, 2009. IJCNN 2009. International Joint Conference on, pp. 697–704 (2009). DOI 10.1109/IJCNN.2009.5178917
16. Fritzke, B.: A growing neural gas network learns topologies. In: G. Tesauro, D.S. Touretzky, T.K. Leen (eds.) Advances in Neural Information Processing Systems 7, pp. 625–632. MIT Press, Cambridge MA (1995)
17. Fusi, S., Annunziato, M., Badoni, D., Salamon, A., Amit, D.J.: Spike-driven synaptic plasticity: Theory, simulation, vlsi implementation. Neural Computation **12**(10), 2227–2258 (2000)
18. Goodman, D., Brette, R.: Brian: a simulator for spiking neural networks in python. BMC Neuroscience **9**(Suppl 1), P92 (2008). DOI 10.1186/1471-2202-9-S1-P92
19. Grossberg, S.: Studies of the mind and brain. Reidel, Boston, USA (1982)
20. Hamed, H., Kasabov, N., Shamsuddin, S.: Integrated feature selection and parameter optimisation for evolving spiking neural networks using quantum inspired particle swarm optimisation. In: International Conference on Soft Computing and Pattern Recognition, pp. 695–698. IEEE Press (2009)
21. Hamed, H., Kasabov, N., Shamsuddin, S.: Probabilistic evolving spiking neural network optimization using dynamic quantum-inspired particle swarm optimization. Australian Journal of Intelligent Information Processing Systems **11**(1), 23–28 (2010)
22. Hamed, H.N.A., Shamsuddin, S.M., Kasabov, N.: Quantum-inspired particle swarm optimization for feature selection and parameter optimization in evolving spiking neural networks for classification tasks. In: Numerical Analysis and Scientific Computing, pp. 133–148. InTech, AUT (2011)
23. Hisada, M., Ozawa, S., Zhang, K., Kasabov, N.: Incremental linear discriminant analysis for evolving feature spaces in multitask pattern recognition problems. Evolving Systems **1**, 17–27 (2010). DOI 10.1007/s12530-010-9000-3
24. Huang, L., Song, Q., Kasabov, N.: Evolving connectionist system based role allocation for robotic soccer. International Journal of Advanced Robotic Systems **5**, 59–62 (2008)
25. Indiveri, G., Linares-Barranco, B., Julia Hamilton, T., Van Schaik, A., Etienne-Cummings, R., Delbruck, T., Liu, S.C., Dudek, P.,

Häfliger, P., Renaud, S., Schemmel, J., Cauwenberghs, G., Arthur, J., Hynna, K., Folowosele, F., Saighi, S., Serrano-Gotarredona, T., Wijekoon, J., Wang, Y., Boahen, K.: Neuromorphic silicon neuron circuits. Frontiers in neuroscience pp. 1–23 (2011). DOI 10.3389/fnins.2011.00073
26. Johnston, S.P., Prasad, G., Maguire, L., McGinnity, T.M.: An fpga hardware/software co-design towards evolvable spiking neural networks for robotics applications. International Journal of Neural Systems **20**(6), 447–461 (2010)
27. Kasabov, K., Schliebs, R., Kojima, H.: Probabilistic computational neurogenetic modeling: From cognitive systems to alzheimer's disease. Autonomous Mental Development, IEEE Transactions on **3**(4), 300–311 (2011). DOI 10.1109/TAMD.2011.2159839
28. Kasabov, N.: ECOS: Evolving connectionist systems and the ECO learning paradigm. In: S. Usui, T. Omori (eds.) The Fifth International Conference on Neural Information Processing, ICONIP'98, pp. 1232–1235. IOA Press, Kitakyushu, Japan (1998)
29. Kasabov, N.: Evolving fuzzy neural networks-algorithms, applications and biological motivation. In: T. Yamakawa, G. Matsumoto (eds.) Methodologies for the Conception Design and Application of Soft Computing, pp. 271–274. World Scientific, Singapore (1998)
30. Kasabov, N.: Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **31**(6), 902 –918 (2001). DOI 10.1109/3477.969494
31. Kasabov, N.: On-line learning, reasoning, rule extraction and aggregation in locally optimized evolving fuzzy neural networks. Neurocomputing **41**(14), 25 – 45 (2001). DOI 10.1016/S0925-2312(00)00346-5
32. Kasabov, N.: Evolving Connectionist Systems. Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines. Perspectives in Neural Computing. Springer-Verlag, London (2002)
33. Kasabov, N.: Adaptation and interaction in dynamical systems: Modelling and rule discovery through evolving connectionist systems. Applied Soft Computing **6**(3), 307–322 (2006). DOI 10.1016/j.asoc.2005.01.006
34. Kasabov, N.: Evolving Connectionist Systems: The Knowledge Engineering Approach, second edn. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2007)
35. Kasabov, N.: Adaptive modeling and discovery in bioinformatics: The evolving connectionist approach. International Journal of Intelligent Systems **23**(5), 545–555 (2008). DOI 10.1002/int.20282
36. Kasabov, N.: To spike or not to spike: A probabilistic spiking neuron model. Neural Networks **23**(1), 16–19 (2010). DOI http://dx.doi.org/10.1016/j.neunet.2009.08.010
37. Kasabov, N.: Evolving, probabilistic spiking neural networks and neurogenetic systems for spatio-and spectro-temporal data modelling and pattern recognition. Natural Intelligence: The INNS Magazine **1**(2), pp.23–37 (2012)
38. Kasabov, N., Benuskova, L., Wysoski, S.: A computational neurogenetic model of a spiking neuron. In: Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on, vol. 1, pp. 446–451 (2005). DOI 10.1109/IJCNN.2005.1555872
39. Kasabov, N., Dhoble, K., Nuntalid, N., Indiveri, G.: On-line spatio- and spectro-temporal pattern recognition with evolving spiking neural networks utilising integrated rank oder- and spike-time learning. Neural Networks (2012). Under review
40. Kasabov, N., Hu, Y.: Integrated optimisation method for personalised modelling and case studies for medical decision support. International Journal of Functional Informatics and Personalised Medicine **3**(3), 236–256 (2010). DOI 10.1504/IJFIPM.2010.039123
41. Kasabov, N., Song, Q.: DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction. IEEE Transactions on Fuzzy Systems **10**(2), 144–154 (2002). DOI 10.1109/91.995117
42. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence **97**(1-2), 273 – 324 (1997). DOI DOI: 10.1016/S0004-3702(97)00043-X
43. Kohonen, T.: Self-Organizing Maps, second edn. Springer, Verlag (1997)
44. Komijani, M., Lucas, C., Araabi, B., Kalhor, A.: Introducing evolving takagisugeno method based on local least squares support vector machine models. Evolving Systems **3**, 81–93 (2012). DOI 10.1007/s12530-011-9043-0
45. Lichtsteiner, P., Delbruck, T.: A 64x64 aer logarithmic temporal derivative silicon retina. In: Research in Microelectronics and Electronics, vol. 2, pp. 202–205 (2005). DOI 10.1109/RME.2005.1542972
46. Lin, C.T., Lee, C.S.G.: Neuro Fuzzy Systems. Prentice Hall (1996)
47. Loiselle, S., Rouat, J., Pressnitzer, D., Thorpe, S.: Exploration of rank order coding with spiking neural networks for speech recognition. In: IEEE International Joint Conference on Neural Networks, IJCNN '05, vol. 4, pp. 2076–2080 (2005). DOI 10.1109/IJCNN.2005.1556220
48. Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural Computation **14**(11), 2531–2560 (2002). DOI 10.1162/089976602760407955
49. Meng, Y., Jin, Y., Yin, J., Conforth, M.: Human activity detection using spiking neural networks regulated by a gene regulatory network. In: The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1–6 (2010). DOI 10.1109/IJCNN.2010.5596340
50. Norton, D., Ventura, D.: Preparing more effective liquid state machines using hebbian learning. In: International Joint Conference on Neural Networks, IJCNN 2006, pp. 4243–4248. IEEE, Vancouver, BC (2006). DOI 10.1109/IJCNN.2006.246996
51. Nuntalid, N., Kasabov, N.: Eeg spatio/spectro temporal pattern recognition with evolving probabilistic spiking neural networks,. IEEE Trans. Neural Networks (2012). Under review
52. Ozawa, S., Pang, S., Kasabov, N.: Incremental learning of chunk data for online pattern classification systems. Neural Networks, IEEE Transactions on **19**(6), 1061 –1074 (2008). DOI 10.1109/TNN.2007.2000059
53. Ozawa, S., Toh, S.L., Abe, S., Pang, S., Kasabov, N.: Incremental learning of feature space and classifier for face recognition. Neural Networks **18**(56), 575 – 584 (2005). DOI 10.1016/j.neunet.2005.06.016
54. Perrinet, L., Delorme, A., Samuelides, M., Thorpe, S.J.: Networks of integrate-and-fire neuron using rank order coding A: How to implement spike time dependent Hebbian plasticity. Neurocomputing **38-40**, 817–822 (2001)
55. Platt, J.: A resource-allocating network for function interpolation. Neural Comput. **3**(2), 213–225 (1991). DOI 10.1162/neco.1991.3.2.213
56. Rabiner, L., Juang, B.H.: Fundamentals of speech recognition. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1993)
57. Riul, A., de Sousa, H.C., Malmegrim, R.R., dos Santos, D.S., Carvalho, A.C.P.L.F., Fonseca, F.J., Oliveira, O.N., Mattoso, L.H.C.: Wine classification by taste sensors made from ultra-thin films and using neural networks. Sensors and Actuators B: Chemical **98**(1), 77 – 82 (2004). DOI DOI:10.1016/j.snb.2003.09.025
58. Schliebs, S.: Heterogeneous probabilistic models for optimisation and modelling of evolving spiking neural networks. Ph.D. thesis, Auckland University of Technology (2010). http://hdl.handle.net/10292/963
59. Schliebs, S., Defoin-Platel, M., Kasabov, N.: Integrated feature and parameter optimization for an evolving spiking neural network. In: M. Köppen, N.K. Kasabov, G.G. Coghill (eds.) Advances in Neuro-Information Processing, 15th International Con-

ference, *Lecture Notes in Computer Science*, vol. 5506, pp. 1229–1236. Springer, Heidelberg, Germany (2009). DOI 10.1007/978-3-642-02490-0_149
60. Schliebs, S., Defoin-Platel, M., Kasabov, N.: Analyzing the dynamics of the simultaneous feature and parameter optimization of an evolving spiking neural network. In: The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2010). DOI 10.1109/IJCNN.2010.5596548
61. Schliebs, S., Defoin-Platel, M., Worner, S., Kasabov, N.: Integrated feature and parameter optimization for an evolving spiking neural network: Exploring heterogeneous probabilistic models. Neural Networks **22**(5-6), 623 – 632 (2009). DOI 10.1016/j.neunet.2009.06.038
62. Schliebs, S., Defoin-Platel, M., Worner, S., Kasabov, N.: Quantum-inspired feature and parameter optimisation of evolving spiking neural networks with a case study from ecological modeling. In: International Joint Conference on Neural Networks, IEEE - INNS - ENNS, vol. 0, pp. 2833–2840. IEEE Computer Society, Los Alamitos, CA, USA (2009). DOI 10.1109/IJCNN.2009.5179049
63. Schliebs, S., Defoin-Platel, M.D., Kasabov, N.: On the probabilistic optimization of spiking neural networks. International Journal of Neural Systems **20**(6), 481–500 (2010)
64. Schliebs, S., Fiasché, M., Kasabov, N.: Constructing robust liquid state machines to process highly variable data streams. In: International Conference on Neural Networks (ICANN'12), LNCS, pp. 1–8. Springer, Heidelberg, Lausanne, Switzerland (2012). Under review
65. Schliebs, S., Hamed, H.N.A., Kasabov, N.: A reservoir-based evolving spiking neural network for on-line spatio-temporal pattern learning and recognition. In: 18th International Conference on Neural Information Processing, no. 7063 in LNCS, pp. 160–168. Springer, Heidelberg, Shanghai, China (2011)
66. Schliebs, S., Nuntalid, N., Kasabov, N.: Towards spatio-temporal pattern recognition using evolving spiking neural networks. In: K. Wong, B. Mendis, A. Bouzerdoum (eds.) Neural Information Processing. Theory and Algorithms, *Lecture Notes in Computer Science*, vol. 6443, pp. 163–170. Springer Berlin / Heidelberg (2010). DOI 10.1007/978-3-642-17537-4_21
67. Soltic, S., Kasabov, N.: Knowledge extraction from evolving spiking neural networks with rank order population coding. International Journal of Neural Systems **20**(6), 437–445 (2010). DOI 10.1142/S012906571000253X
68. Soltic, S., Wysoski, S., Kasabov, N.: Evolving spiking neural networks for taste recognition. In: IEEE World Congress on Computational Intelligence (WCCI), Hong Kong, pp. 2091–2097 (2008). DOI 10.1109/IJCNN.2008.4634085
69. Song, Q., Kasabov, N.: TWNFI – a transductive neuro-fuzzy inference system with weighted data normalization for personalized modeling. Neural Networks **19**(10), 1591–1596 (2006). DOI 10.1016/j.neunet.2006.05.028
70. Swope, J.: Artdecos, adaptive evolving connectionist model and application to heart rate variability. Evolving Systems **3**, 95–109 (2012). DOI 10.1007/s12530-012-9049-2
71. Thorpe, S.J.: Spike arrival times: A highly efficient coding scheme for neural networks. In: R. Eckmiller, G. Hartmann, G. Hauske (eds.) Parallel Processing in Neural Systems, International Conference on, pp. 91–94. North-Holland: Elsevier (1990)
72. Thorpe, S.J.: How can the human visual system process a natural scene in under 150ms? On the role of asynchronous spike propagation. In: ESANN. D-Facto public (1997)
73. Thorpe, S.J., Delorme, A., van Rullen, R.: Spike-based strategies for rapid processing. Neural Networks **14**(6-7), 715–725 (2001)
74. Thorpe, S.J., Fize, D., Marlot, C.: Speed of processing in the human visual system. Nature **381**, 520–522 (1996). DOI 10.1038/381520a0
75. Thorpe, S.J., Gautrais, J.: Rapid visual processing using spike asynchrony. In: Advances in Neural Information Processing Systems 9, NIPS, pp. 901–907. MIT Press, Denver, CO, USA (1996)
76. Thorpe, S.J., Gautrais, J.: Rank order coding. In: CNS '97: Proceedings of the 6th annual conference on Computational neuroscience: trends in research, 1998, pp. 113–118. Plenum Press, New York, NY, USA (1998)
77. Thorpe, S.J., Guyonneau, R., Guilbaud, N., Allegraud, J.M., VanRullen, R.: SpikeNet: real-time visual processing with one spike per neuron. Neurocomputing **58-60**, 857 – 864 (2004). DOI 10.1016/j.neucom.2004.01.138
78. Van Rullen, R., Gautrais, J., Delorme, A., Thorpe, S.: Face processing using one spike per neurone. Biosystems **48**(1-3), 229–239 (1998)
79. Van Rullen, R., Thorpe, S.J.: Rate coding versus temporal order coding: What the retinal ganglion cells tell the visual cortex. Neural Comput. **13**(6), 1255–1283 (2001). DOI 10.1162/08997660152002852
80. Watts, M.: A decade of Kasabov's evolving connectionist systems: A review. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on **39**(3), 253–269 (2009). DOI 10.1109/TSMCC.2008.2012254
81. Wysoski, S., Benuskova, L., Kasabov, N.: Brain-like evolving spiking neural networks for multimodal information processing. In: A. Hanazawa, T. Miki, K. Horio (eds.) Brain-Inspired Information Technology, *Studies in Computational Intelligence*, vol. 266, pp. 15–27. Springer Berlin / Heidelberg (2010). DOI 10.1007/978-3-642-04025-2_3
82. Wysoski, S.G.: Evolving spiking neural networks for adaptive audiovisual pattern recognition. Ph.D. thesis, Auckland University of Technology (2008). Http://hdl.handle.net/10292/390
83. Wysoski, S.G., Benuskova, L., Kasabov, N.: On-line learning with structural adaptation in a network of spiking neurons for visual pattern recognition. In: Artificial Neural Networks ICANN 2006, pp. 61–70. Springer, Berlin / Heidelberg (2006). DOI 10.1007/11840817_7
84. Wysoski, S.G., Benuskova, L., Kasabov, N.: Text-independent speaker authentication with spiking neural networks. In: ICANN (2), vol. 4669/2007, pp. 758–767. Springer, Berlin / Heidelberg (2007). DOI 10.1007/978-3-540-74695-9_78
85. Wysoski, S.G., Benuskova, L., Kasabov, N.: Adaptive spiking neural networks for audiovisual pattern recognition. In: Neural Information Processing: 14th International Conference, ICONIP 2007, pp. 406–415. Springer-Verlag, Berlin, Heidelberg (2008). DOI 10.1007/978-3-540-69162-4_42
86. Wysoski, S.G., Benuskova, L., Kasabov, N.: Fast and adaptive network of spiking neurons for multi-view visual pattern recognition. Neurocomputing **71**(13-15), 2563 – 2575 (2008). DOI 10.1016/j.neucom.2007.12.038
87. Wysoski, S.G., Benuskova, L., Kasabov, N.: Evolving spiking neural networks for audiovisual information processing. Neural Networks **23**(7), 819–835 (2010). DOI 10.1016/j.neunet.2010.04.009
88. Wysoski, S.G., Benuskova, L., Kasabov, N.K.: Adaptive learning procedure for a network of spiking neurons and visual pattern recognition. In: Advanced Concepts for Intelligent Vision Systems, pp. 1133–1142. Springer, Berlin / Heidelberg (2006). DOI 10.1007/11864349_103
89. Zuppicich, A., Soltic, S.: FPGA implementation of an evolving spiking neural network. In: M. Köppen, N.K. Kasabov, G.G. Coghill (eds.) Advances in Neuro-Information Processing, 15th International Conference, *Lecture Notes in Computer Science*, vol. 5506, pp. 1129–1136. Springer, Heidelberg, Germany (2009). DOI 10.1007/978-3-642-02490-0_137