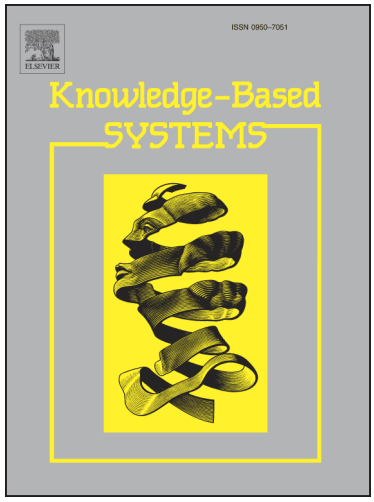


Evolving Connectionist Systems for Adaptive Learning and Knowledge Discovery: Trends and Directions

Nikola K. Kasabov

PII: S0950-7051(15)00004-0
DOI: http://dx.doi.org/10.1016/j.knosys.2014.12.032
Reference: KNOSYS 3041

To appear in: *Knowledge-Based Systems*

Received Date: 13 October 2014
Revised Date: 27 December 2014
Accepted Date: 30 December 2014

Please cite this article as: N.K. Kasabov, Evolving Connectionist Systems for Adaptive Learning and Knowledge Discovery: Trends and Directions, *Knowledge-Based Systems* (2015), doi: http://dx.doi.org/10.1016/j.knosys.2014.12.032

Revised version of the paper KNOSYS-D-14-01264

Evolving Connectionist Systems for Adaptive Learning and Knowledge Discovery: Trends and Directions

Nikola K. Kasabov, FIEEE, FRSNZ

Knowledge Engineering and Discovery Research Institute - KEDRI,
Auckland University of Technology, nkasabov@aut.ac.nz, www.kedri.info

Abstract

This paper follows the 25 years of development of methods and systems for knowledge-based neural network systems and more specifically the recent evolving connectionist systems (ECOS). ECOS combine the adaptive/evolving learning ability of neural networks and the approximate reasoning and linguistically meaningful explanation features of symbolic representation, such as fuzzy rules. This review paper presents the classical now hybrid expert systems and evolving neuro-fuzzy systems, along with new developments in spiking neural networks, neurogenetic systems, and quantum inspired systems, all discussed from the point of few of their adaptability, model interpretability and knowledge discovery. The paper discusses new directions for the integration of principles from neural networks, fuzzy systems, bio- and neuroinformatics, and nature in general.

Keywords: Knowledge-based systems; Neuro-fuzzy systems**;** Evolving Connectionist Systems; Evolving Spiking Neural Networks; Computational Neurogenetic Systems; Quantum inspired spiking neural networks; Spatio-temporal pattern recognition.

1. Hybrid connectionist systems

The human brain uniquely combines low level neuronal learning in the neurons and the connections between them and higher level rule abstraction leading to adaptive learning and abstract concept formation. This is the ultimate inspiration for the development of hybrid connectionist systems where specially constructed artificial neural networks (NN) are trained on data so that after training abstract knowledge representation can be derived that explains the data and can be further interpreted as a knowledge-based system. Examples of such hybrid connectionist systems are connectionist-based expert systems (Kasabov, 1991), neuro-fuzzy systems (Yamakawa, Uchino, et al, 1992), evolving connectionist systems (Kasabov, 1998), evolving fuzzy systems (Angelov, 2002).

In the past 50 years several seminal works in the areas of neural networks (Amari, 1967; Amari, 1990), fuzzy systems (Zadeh, 1965) and rule-based expert systems (Knowledge-Based Systems Journal, 1988) opened a new field of information science - the creation of new types of hybrid systems that combine the learning ability of neural networks, at a lower level of information processing, and the reasoning and explanation ability of rule-based systems, at the higher level. The first hybrid connectionist expert systems combined NN and propositional type of rules – either production rules, implemented in CLIPS (Kasabov, 1991, 94-96; Kasabov and Shishkov, 1993), or first order logic rules implemented in PROLOG (Kasabov, 1996).

Combining NN and fuzzy rule based systems is illustrated on a simple example in figure 1. A NN module is trained to predict the value of a stock index based on the current day and the day before prices. At a higher level a fuzzy reasoning module combines the predicted value by the NN module with a macro-economic variable (Good or Bad state of the economy) and a variable representing the political situation in the country (stable or unstable) using the following types of fuzzy rules (Kasabov, 1996):

IF <the predicted by the NN module stock value is high> AND <the economic situation is good> AND <the political situation is Stable) THEN <buy stock> (1)

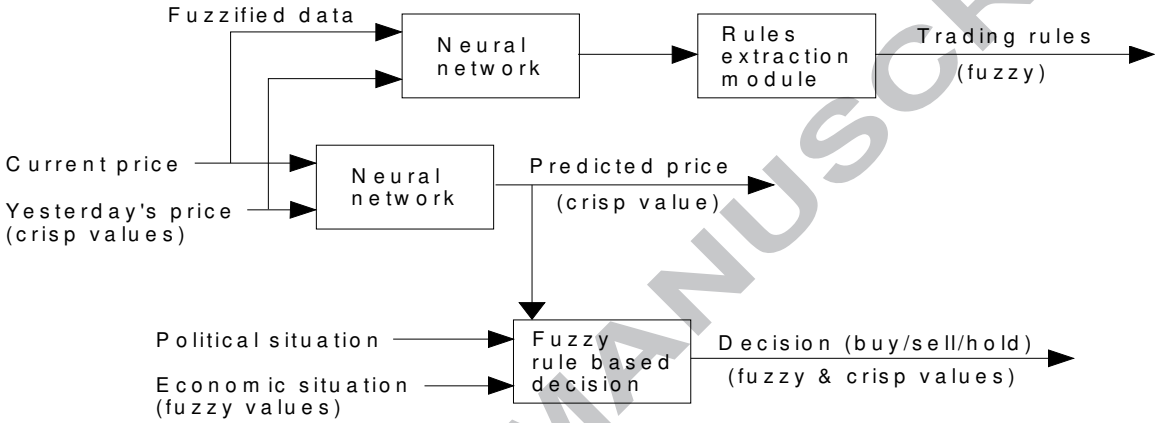


Figure 1: A hybrid NN-fuzzy rule-based expert system for financial decision support (from Kasabov, 1996)

A traditional Multi-Layer Perceptron (MLP) NN is used to implement the NN module in fig.1 and a fuzzy inference system is used to implement the Fuzzy rule-based decision module. In addition to these two module, a specifically constructed fuzzy neural network module is used to be trained on data and to extract fuzzy trading rules. Fuzzy neural networks and their further development as evolving connectionist systems are presented next in the paper.

2. Fuzzy neurons and fuzzy neural networks. Evolving connectionist systems

A low-level integration of fuzzy rules into a single neuron model and larger neural network structures, tightly coupling learning and fuzzy reasoning rules into connectionists structures, was initiated by Professor Takeshi Yamakawa and other Japanese scientists and promoted at a series of IIZUKA conferences in Japan (Yamakawa, Uchino, et al, 1992). Since then, many models of fuzzy neural networks were developed based on these principles (Furuhashi, Hasegawa, et al, 1993; Kasabov, 1996; Kasabov, Kim, et al, 1997).

The evolving neuro-fuzzy systems developed these ideas further, where instead of training a fixed connectionist structure, the structure and its functionality were evolving from incoming data, often in an on-line, one-pass learning mode. This is the case with the evolving connectionist systems, ECOS (Kasabov, 1998; 2001; 2003; 2007; Kasabov, Kim, et al, 1997; Kasabov and Song, 2002).

ECOS are modular connectionist based systems that evolve their structure and functionality in a continuous, self-organised, on-line, adaptive, interactive way from incoming data (Kasabov, 1998). They can process both data and knowledge in a supervised and/or unsupervised way. 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. They can learn incrementally single data records or chunks of data and also incrementally change their input features. ECOS further develops some connectionist information processing principles already introduced in classical NN models, such as: SOM, RBF, FuzyARTMap, Growing neural gas, neuro-fuzzy systems, RAN.

ECOS perform *adaptive* l*ocal learning* - neurons are allocated as centres of data clusters and the system creates local models in these clusters. The clustering used in ECOS is on-line, one-pass, evolving clustering, which is in contrast to the traditional fuzzy clustering methods that use pre-defined number of clusters and many iterations (Bezdek, ed., 1987; Yager and Filey, 1994).

The following are the main principles of ECOS as stated in (Kasabov, 1998):

(1) Fast learning from large amount of data, e.g. using 'one-pass' training, starting with little prior knowledge;
(2) Adaptation in a real time and in an on-line mode where new data is accommodated as it comes based on local learning;
(3) 'Open', evolving structure, where new input variables (relevant to the task), new outputs (e.g. classes), new connections and neurons are added/evolved 'on the fly';
(4) Both data learning and knowledge representation is facilitated in a comprehensive and flexible way, e.g. supervised learning, unsupervised learning, evolving clustering, 'sleep' learning, forgetting/pruning, fuzzy rule insertion and extraction;
(5) Active interaction with other ECOSs and with the environment in a multi-modal fashion;
(6) Representing both space and time in their different scales, e.g.: clusters of data, short- and long-term memory, age of data, forgetting, etc.;
(7) System's self-evaluation in terms of behaviour, global error and success and related knowledge representation.

In 1998 Walter Freeman, who attended the ICONIP conference then, commented on the proposed ECOS concepts: "...Through the 'chemicals' and let the system grow …".

The development of ECOS, as a trend in neural networks and computational intelligence that started in 1998 (Kasabov, 1998) continued as many improved or new computational *methods* that use the ECOS principles have been developed along many *applications.*

Here the concepts of ECOS are illustrated on two implementations: Evolving Fuzzy Neural Network, EFuNN (Kasabov, 2001) and Dynamic Evolving Neuro-Fuzzy Inference Systems, DENFIS (Kasabov and Song, 2002). Examples of EFuNN and DENFIS are shown in figures 2 and figure 3 respectively. In ECOS, 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), or in both the input and output space (this is

the case in the EFuNN models). Samples (examples) that have a distance to an existing neuronal 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 fuzzy model/function, e.g.:

IF <data is in a fuzzy cluster Ci > THEN <the model is Fi>, (2)

where Fi can be a fuzzy membership function (EFuNN, Kasabov, 2001), a linear or regression function (figure 4) or a NN model (Kasabov and Song, 2002; Kasabov, 2003; 2007).

Figure 2 shows an example of EFuNN model that consists of 5 layers: input neurons accept real- value inputs; fuzzy input layer produces fuzzy membership values between 0 and 1 according to the membership degree of a corresponding input value to one of several (in this case just two) fuzzy membership functions (e.g. Small and Large values); evolving rule (case) neuronal nodes that represent evolving clusters of data with their connection weights representing the coordinates of these clusters; fuzzy output neurons, calculating the membership degrees of the output value that corresponds to the input vector, to output membership functions (in this case only 3 are shown); output neuron that represents the real output value that correspond to the input vector.
Adaptive (evolving) learning in EFuNN is performed through supervised clustering of the input data vectors so that input vectors that are similar and have similar putput values are clustered in one cluster represented by the same rule node.

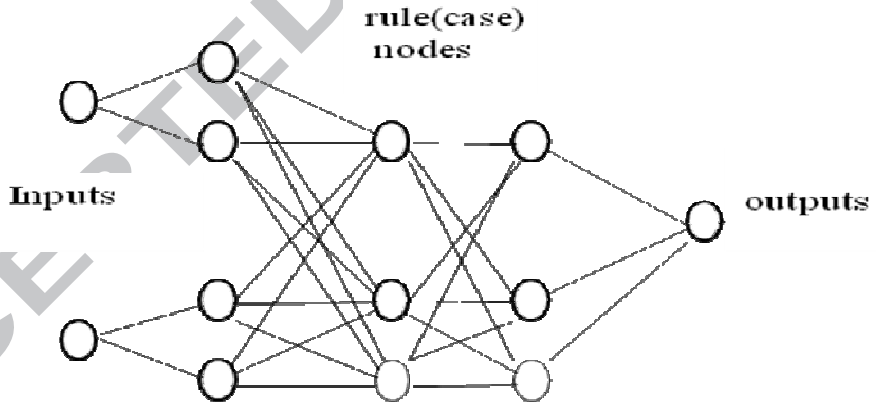


Figure 2: An example of EFuNN model that consists of 5 layers (see the text for explanation)

Figure 3 shows an example of DENFIS for a real problem application (Kasabov, 2007). Five input variables are used that represent data and a real value output is associated with each example (sample, vector). The DENFIS learning consist of unsupervised clustering (clusters are created based only on the similarity of the input vectors) and the estimation of local output functions that approximate the data from each cluster.

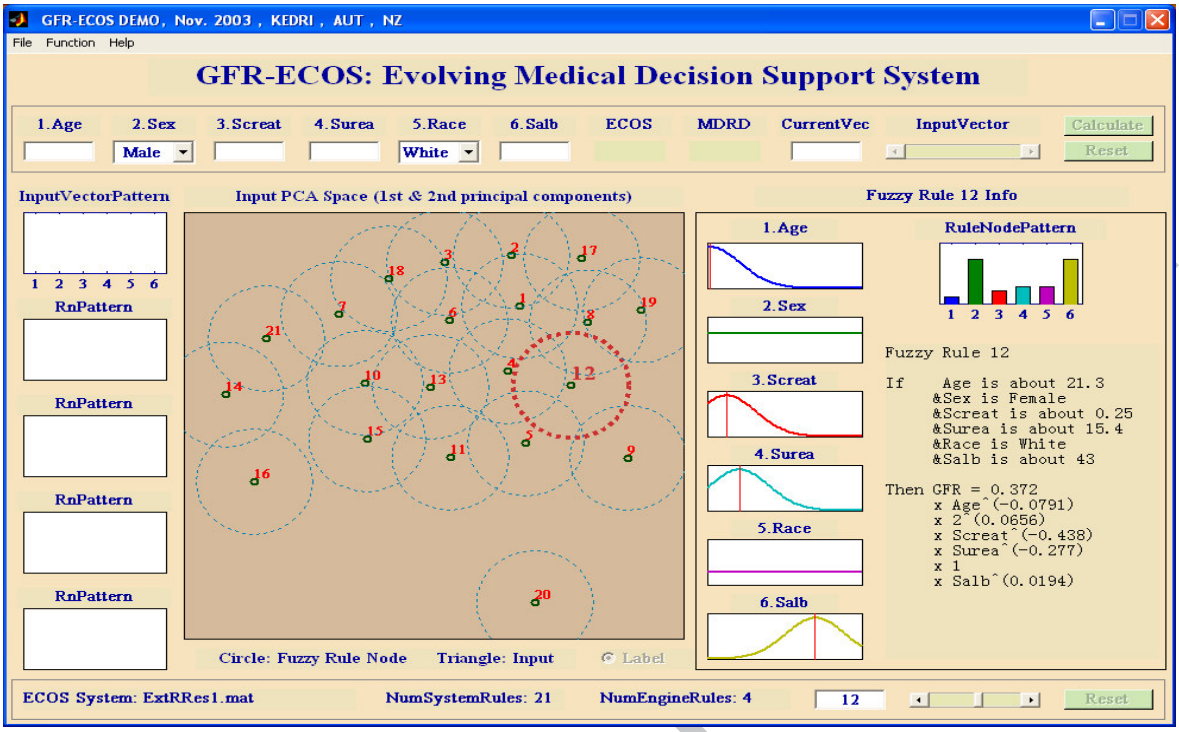


Figure 3: An example of DENFIS model (from Kasabov, 2007) with 6 input variables for a practical application of estimating the GFR parameter of the renal function of a patient.

The clusters shown in Fig.3 are defined as fuzzy clusters and the output functions are logistic regressions. As an example, a fuzzy rule is shown that approximates the data in cluster 12 (defined as a fuzzy cluster where Age is about 21.3 etc.) and a logistic regression function is evolved to calculate the GFR of any new patient whose input data vector will fall into cluster 12:

Fuzzy Rule 12: IF (Age is About 21.3 AND Sex is Female AND …)
THEN (GFR= 0.372 . $2^{0.0656}$ SCreat $^{-0.438}$…) (see fig.3) (3)

When tested on Australia and New Zealand GFR data the GFR-ECOS from Fig.3 was 15-20% more accurate than the traditional global regression function used as a golden rule in clinical practice - the MDRD formula (Kasabov, 2007). But the most important feature of the ECOS-based system was its local knowledge discovery (the local fuzzy rules) that makes it possible to explain differences between different groups of patients based on the used input variables.

A special direction of ECOS was transductive reasoning and personalised modelling. Instead of building a set of local models (e.g. prototypes) to cover the whole problem space and then use these models to classify/predict any new input vector, in transductive modelling for every new input vector a new model is created based on selected nearest neighbour vectors from the available data. Such ECOS models are Neuro-Fuzzy Inference model, NFI, and Transductive, weighted neuro-fuzzy inference model, TWNFI (Song and Kasabov, 2006). In TWNFI for every new input vector the neighbourhood of closets data vectors is optimised using both the

distance between the new vector and the neighbouring ones and the weighted importance of the input variables, so that the error of the model is minimised. As one application, in (Kasabov and Hu, 2010) a method for personalised modelling is proposed so that the features, the neighbourhood and the model type are all optimised together with the use of an evolutionary computation algorithm. The method consists of the following steps as also illustrated in fig.4.

- Define a global data set of personalized records (module 1) related to the record of a new input vector **x** (module 2) for which a personalized model Mx is developed for classification or prediction purposes;
- Select (module 3) the most important variables Vx for the person represented by the data vector ***x;***
- Select a neighbourhood Dx (module 4) of the Kx closest samples to **x**;
- Rank the importance of the features in Vx (module 5);
- Create a personalised model Mx (module 6) for this person to predict an outcome, calculate the output for the person for the problem and evaluate the accuracy of the suggested outcome, which step includes an iterative application of the previous steps.
- Create and evaluate a personal profile (module 7) of the person in regard to possible outcomes. If necessary design personal improvement scenarios, consisting of suggested changes in the values of the persons' variables as a concert taking into account their ranking to improve the outcome.

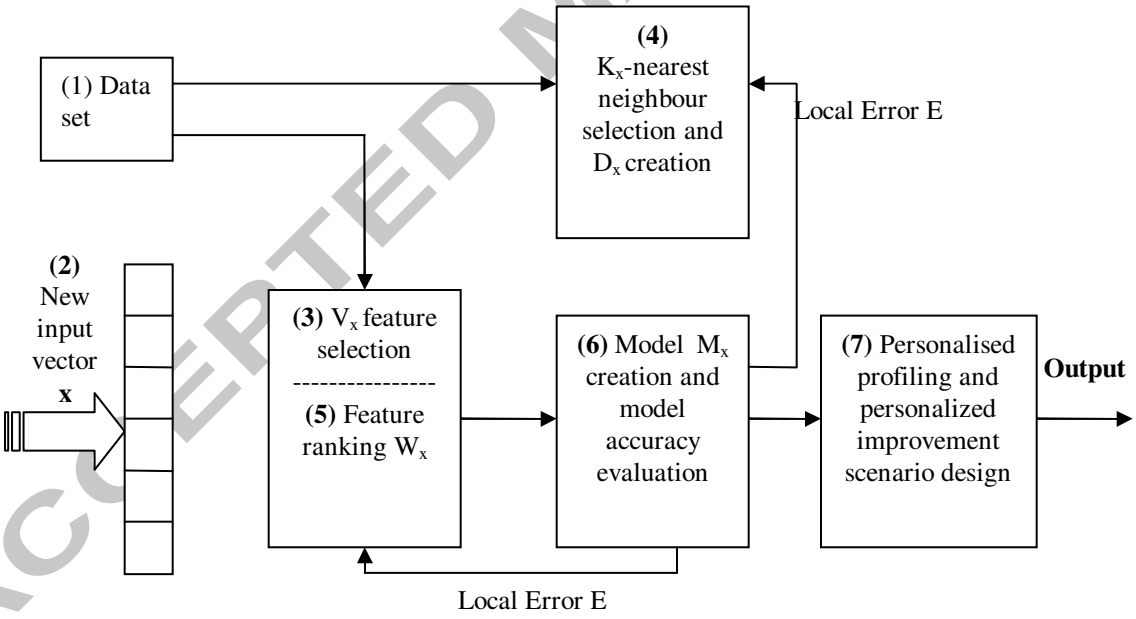


Fig. 4. A block-diagram of a personalised modelling system (from Kasabov and Hu, 2010)

The personalised profile created in module 7 can be a fuzzy rule that is derived from the personalised model Mx and a personalised ranking of the importance of the variables for the person (**x**) under consideration. That would allow for a better understanding of person x conditions, a better prediction of a possible outcome and for a personalised treatment design.

Examples of using this methodology are many, among them: cancer diagnosis based on gene expression data; diabetes prognosis; criminal behaviour prognosis; predicting establishment of harmful species based on climate data; stock index prediction; etc. (Kasabov, 2007)
Several ECOS models, including EFuNN, its simplified version ECF, DENFIS, TWNFI and others are implemented in a publicly available simulator NeuCom (www.theneucom.com; www.kedri.aut.ac.nz).

While the classical ECOS use a simple McCulloch and Pitts model of a neuron, where data is represented as scalars, the further developed evolving spiking neural network (eSNN) architectures (see next section) use a spiking neuron model, while applying the same or similar ECOS principles. eSNN use data represented as temporal sequences of spikes in a similar mode as information is represented and processed in the brain. An example was a personalised modelling system using the same functionality as in fig.4, but an evolving spiking neural network model Mx. One application of this method was to predict a personalised risk of stroke occurrence just few days ahead of a possible stroke event with an accuracy of 95% which is much higher when compared with the accuracy of prediction when using traditional NN and neuro-fuzzy models (Kasabov et al, 2014). The main reason is that the data used is a temporal climate data and it is well established now that temporal and spatio-temporal data are more efficiently processed in eSNN (http://ncs.ethz.ch/projects/EvoSpike/. The next section discusses the main principles and the contemporary trends in eSNN and how they process temporal and spatio-temporal data.

3. Current Trends in ECOS: Evolving Spiking Neural Networks (eSNN)

A single biological neuron and the associated synapses is a complex information processing machine that involves short term information processing, long term information storage, and evolutionary information stored as genes in the nucleus of the neuron. A spiking neuron model assumes input information represented as trains of spikes over time. When sufficient input information is accumulated in the membrane of the neuron and the neuron's post synaptic potential exceeds a threshold, the neuron emits a spike at its axon (figure 5). Some of the-state-of-the-art models of a spiking neuron include: early models by Hodgkin and Huxley (Hodgkin and Huxley, 1952); more recent models by Maas, Gerstner, Kistler, Izhikevich and others, e.g.: Spike Response Models (SRM); Integrate-and-Fire Model (IFM) (figure 5); Izhikevich models (Izhikevich, 2004); adaptive IFM; probabilistic neurogenetic model (Kasabov, 2010).

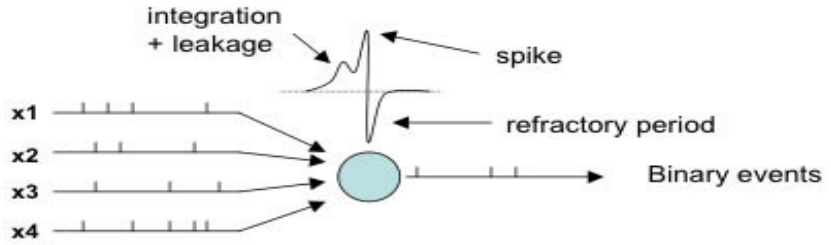


Figure 5. The schema of a LIFM

Based on the ECOS principles, an evolving spiking neural network architecture (eSNN) was proposed (Kasabov, 2007; Wysoski et al, 2010). It was initially designed as a visual pattern recognition system. The first eSNNs were based on the Thorpe's neural model (Thorpe,S., Delorme, A., et al., 2001), 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.

An eSNN evolves its structure and functionality in an on-line manner, from incoming information. For every new input data vector, a new output neuron is dynamically allocated and connected to the input neurons (feature neurons). The neuron's connections are established using the RO rule for the output neuron to recognise this vector (frame, static pattern) or a similar one as a positive example. The weight vectors of the output neurons represent centres of clusters in the problem space and can be represented as fuzzy rules (Soltic and Kasabov, 2010).

In some implementations neurons with similar weight vectors are merged based on Euclidean distance between them. That makes it possible to achieve a very fast learning (only one pass may be sufficient), both in a supervised and in an unsupervised mode (Kasabov, 2007). When in an unsupervised mode, the evolved neurons represent a learned pattern (or a prototype of patterns). The neurons can be labelled and grouped according to their belonging to the same class if the model performs a classification task in a supervised mode of learning – an example is shown in fig.6.

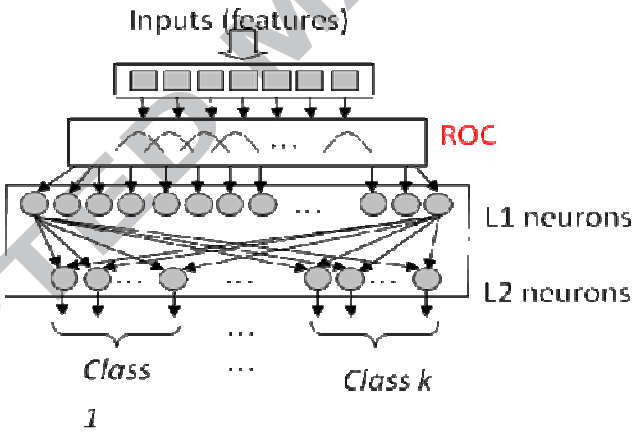


Fig.6. Example of an eSNN for classification using population RO coding of inputs (Soltic and Kasabov, 2010). Each input is connected to several feature neurons representing overlapping Gaussian receptive fields and producing spikes according to how much the current input variable value belongs to the receptive field: the higher the membership degree – the earlier a spike is generated and forwarded to the output neurons for learning or recall. A pool of output neurons representing different input vectors or prototypes is evolved for each class. This model was used for odour recognition. In unsupervised learning, the output neurons are not labelled and not organised as class pools.

During a *learning phase,* for each M-dimensional training input pattern (sample, example, vector) P_i a new output neuron i is created and its connection weights $w_{j,i}$ (j=1,2,...,M) to the input (feature) neurons are calculated based on the order of the incoming spikes on the corresponding synapses using the RO learning rule :

$$w_{j,i} = \alpha . mod^{order(j,i)} \quad (4)$$

where: α is a learning parameter (in a partial case it is equal to 1)*; mod* is a modulation factor, that defines how important the order of the first spike is; $w_{j,i}$ is the synaptic weight between a pre-synaptic neuron *j* and the postsynaptic neuron *i*; *order(j,i)* represents the order (the rank) of the *first* spike at synapse *j,i* ranked among all spikes arriving from all synapses to the neuron *i; order(j,i)* has a value 0 for the first spike to neuron *i* and increases according to the input spike order at other synapses.

While the input training pattern (example) is presented (all input spikes on different synapses, encoding the input vector are presented within a time window of T time units), the spiking threshold Th_i of the neuron *i* is defined to make this neuron spike when this or a similar pattern (example) is presented again in the recall mode. The threshold is calculated as a fraction (C) of the total PSP_i (denoted as PSP_{imax}) accumulated during the presentation of the input pattern:

$$PSP_{imax} = \sum_{(j)} mod^{order(j,i)} \quad (5)$$

$$Th_i = C. PSP_{imax} \quad (6)$$

If the weight vector of the evolved and trained new neuron is similar to the one of an already trained neuron (in a supervised learning mode for classification this is a neuron from the same class pool), i.e. their similarity is above a certain threshold *Sim,* the new neuron will be merged with the most similar one, averaging the connection weights and the threshold of the two neurons (Kasabov, 2007; Wysoski et al, 2010). Otherwise, the new neuron will be added to the set of output neurons (or the corresponding class pool of neurons when a supervised learning for classification is performed). The similarity between the newly created neuron and a training neuron is computed as the inverse of the Euclidean distance between weight matrices of the two neurons. The merged neuron has weighted average weights and thresholds of the merging neurons.

While an individual output neuron represents a single input pattern, merged neurons represent clusters of patterns or prototypes in a transformed spatial – RO space. These clusters can be represented as fuzzy rules (Soltic and Kasabov, 2010) that can be used to discover new knowledge about the problem under consideration.

The eSNN learning is adaptive, incremental, theoretically – 'lifelong', so that the system can learn new patterns through creating new output neurons, connecting them to the input neurons, and possibly merging the most similar ones. The eSNN implement the 7 ECOS principles from section 1.

During the *recall phase*, when a new input vector is presented and encoded as input spikes, the spiking pattern is submitted to all created neurons during the learning phase. An output spike is generated by neuron *i* at a time *l* if the $PSP_i(l)$ becomes higher than its threshold Th_i. After the first neuron spikes, the PSP of all neurons are set to initial value (e.g. 0) to prepare the system for the next pattern for recall or learning.

The postsynaptic potential $PSP_i(l)$ of a neuron i at time *l* is calculated as:

$$PSP_i(l) = \sum \quad \sum e_j(t). mod^{order(j,i)} \quad (7)$$

$t=0,1,2,...,l$ (j)

where: $e_j(t)=1$ if there is a *first* spike at time *t* on synapse j; *order (j,i)* is the rank order of the first spike at synapse *j* among all spikes to neuron *i* for this recall pattern.

The parameter *C*, used to calculate the threshold of a neuron *i,* makes it possible for the neuron *i* to emit an output spike before the presentation of the whole learned pattern (lasting T time units) as the neuron was initially trained to respond. As a partial case C=1. This is an important property of eSNN that can be utilized to train an eSNN on whole temporal input patterns and to recall the eSNN on a partial input pattern, e.g. only initial input data, so that the eSNN can predict an outcome earlier.

The recall procedure can be performed using different recall algorithms implying different methods of comparing input patterns for recall with already learned patterns in the output neurons:

(a) The first one is described above. Spikes of the new input pattern are propagated as they arrive to all trained output neurons and the first one that spikes (its PSP is greater that its threshold) defines the output. The assumption is that the neuron that best matches the input pattern will spike earlier based purely on the PSP (membrane potential). This type of eSNN is denoted as eSNNm.

(b) The second one implies a creation of a new output neuron for each recall pattern, in the same way as the output neurons were created during the learning phase, and then – comparing the connection weight vector of the new one to the already existing neurons using Euclidean distance. The closest output neuron in terms of synaptic connection weights is the 'winner'. This method uses the principle of transductive reasoning and nearest neighbour classification in the connection weight space. It compares spatially distributed synaptic weight vectors of a new neuron that captures a new input pattern and existing ones. We will denote this model as eSNNs.

The main advantage of the eSNN when compared with other supervised or unsupervised SNN models is that it is computationally inexpensive and boosts the importance of the order in which input spikes arrive, thus making the eSNN suitable for on-line learning with a range of applications. For a comprehensive study of eSNN see (Wysoski et al, 2010) and for a comprehensive review - (Schliebs and Kasabov, 2013).

Different eSNN models were further developed, including:

- Reservoir-based eSNN for spatio- and spectro-temporal pattern recognition shown in figure 7 (Schliebs and Kasabov, 2013) (for principles of reservoir computing – see Verstraeten, Schrauwen, et al, 2007);
- Dynamic eSNN (deSNN) - an architecture that uses both rank-order and time-based learning methods (Song, Miller, et al, 2000) to account for spatio-temporal learning (Kasabov et al, 2013);
- Specialised architectures for EEG modelling and moving object recognition systems (Kasabov et al, 2013;);

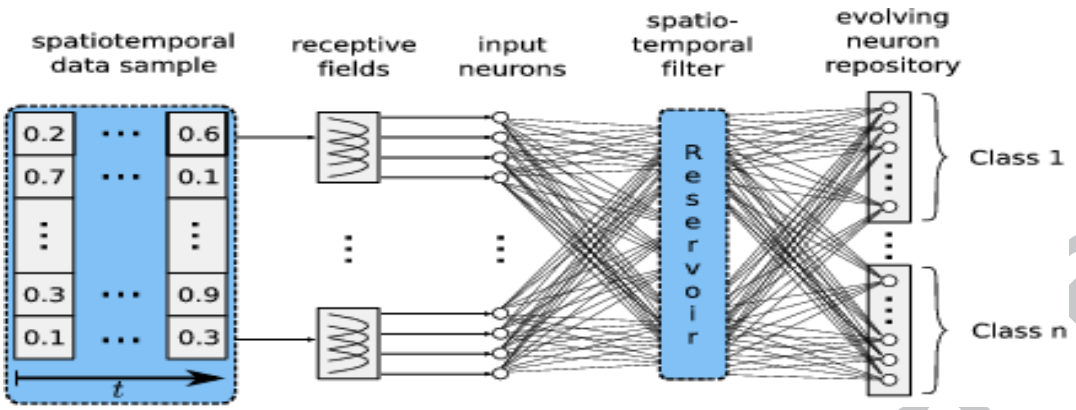
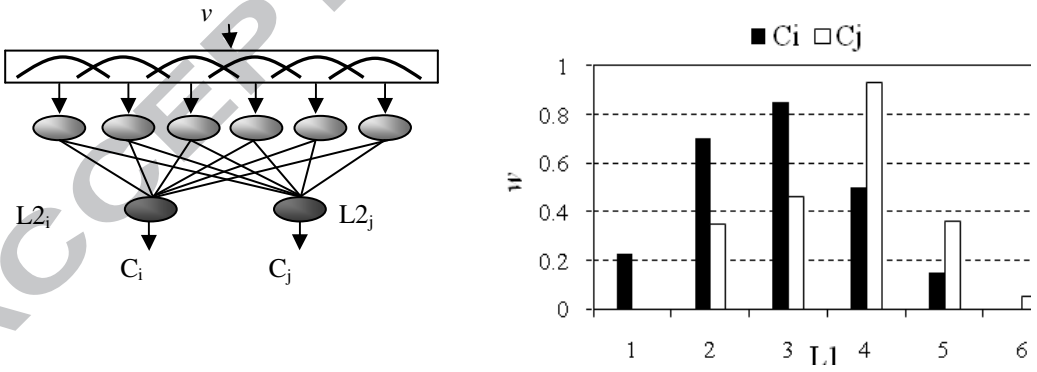


Figure 7: A reservoir-based eSNN. It consists of: input transformation module, transforming real data sequences into spike sequences using receptive fields; reservoir module, usually 3-dimensional, for learning whole spatio-temporal patterns from input data; output classification module, where output spiking neurons are evolved for every learned input pattern along with their assigned class when classification tasks are considered (the neurons from the same class may be merged based on their similarity).

Extracting fuzzy rules from an eSNN would make the eSNN not only efficient learning models, but also knowledge-based models. A method was proposed (Soltic and Kasabov, 2010) and illustrated in figures 8a and 8b. Based on the connection weights (W) between the receptive field layer (L1) and the class output neuron layer (L2), the following fuzzy rules are extracted:

IF(input variable v is SMALL)THEN class C_i ; (8)
IF(v is LARGE)THEN class C_j



(a) (b)

Figure 8: (a): A simple structure of an eSNN for 2-class classification based on one input variable using 6 receptive fields to convert the input values into spike trains; (b): The connection weights of the connections to class Ci and Cj output neurons respectively are interpreted as fuzzy rules.

4. A Current Trend and a Future Direction: Evovlving Computational Neuro-Genetic Models (eCNGM)

A neurogenetic model of a neuron is proposed and studied in (Benuskova and Kasabov, 2007; Kasabov, 2010). It utilises information about how some proteins and genes affect the spiking activities of a neuron such as *fast excitation, fast inhibition, slow excitation, and slow inhibition*. An important part of the model is a dynamic gene/protein regulatory network (GRN) model of the dynamic interactions between genes/proteins over time that affect the spiking activity of the neuron – figure 9.

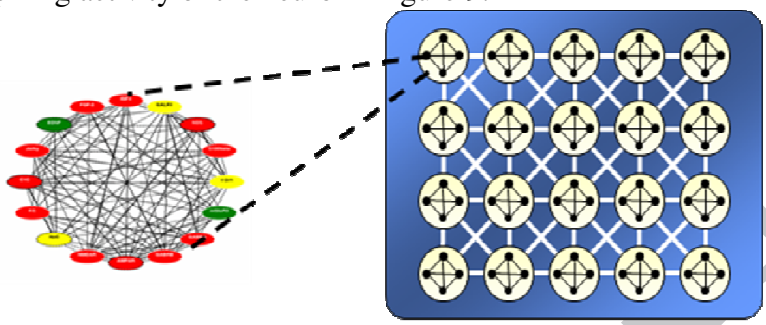


Figure 9: A schematic diagram of an eCNGM, consisting of a GRN as part of a eSNN (Benuskova and Kasabov, 2007).

New types of neuro-genetic fuzzy rules can be extracted from such CNGM in the form of:

IF < GRN is represented by a function F> AND < input is Small> (9)
THEN <Class C>

eCNGM can be used in modelling and prediction the progression of brain degenerative diseases, such as memory loss and Alzheimer's Disease (AD) (Kasabov and Capecci, 2014). eCNGM need to be further developed in terms of both theory and applications.

5. A Current Trend and a Future Direction: Quantum Inspired eSNN (QeSNN)

QeSNNs use the principle of superposition of states to represent and optimize features (input variables) and gene parameters of an eSNN model (Kasabov, 2007). They are optimized through quantum inspired genetic algorithm (Defoin-Platel, Schliebs, et al, 2009) or QiPSO. Features or genes are represented as qu-bits in a superposition of 1 (selected), with a probability α, and 0 (not selected) with a probability β. When the model has to be calculated, the quantum bits 'collapse' in a state of either 1 or 0. Fuzzy rules in QeSNN look like:

IF < GRN is represented by a function F with a quantum probability p > (10)
AND < input is Small with a quantum probability q>
AND <the model parameters are S with quantum probability s>
THEN <Class C, with probability r>

A block diagram of a QeSNN is shown in Fig.10.

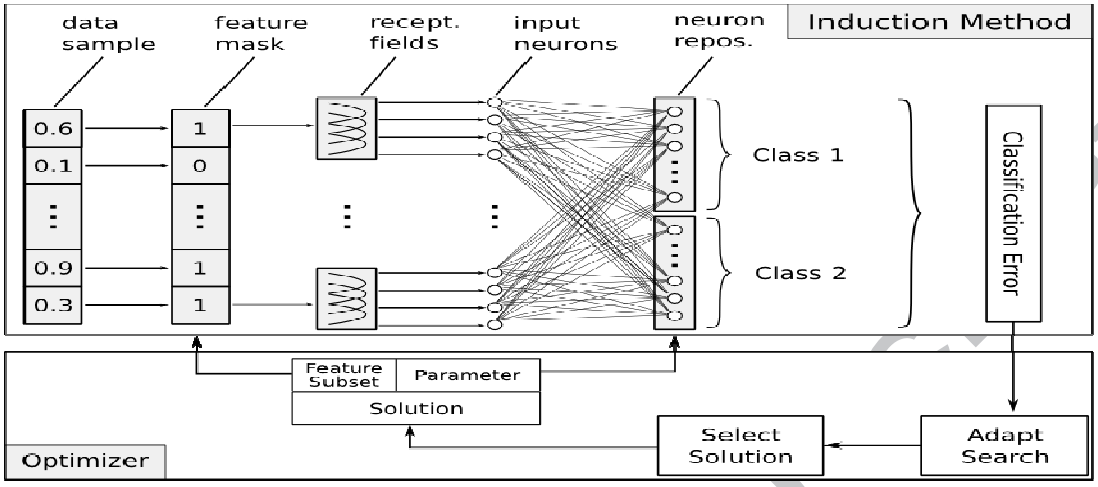


Fig.10. A block diagram of a QeSNN (from S.Schliebs, M.Defoin-Platel and N.Kasabov, 2009).

QeSNN need to be further developed in terms of both theory and applications.

6. A Current Trend and a Future Direction: The NeuCube eSNN Spatio-Temporal Data Machine

The latest development in the direction of eSNN and neurogenetic systems was proposed as a new architecture of a virtual spatio-temporal data machine called NeuCube (Kasabov, 2014). It was initially proposed for spatio-temporal brain data modelling, but then it was used for climate data modelling, stroke occurrence prediction and other applications.

The NeuCube framework is depicted in Fig.11. It consists of the following modules:

- Input information encoding module;
- 3D SNN module (the Cube);
- Output module;
- Gene regulatory network (GRN) module (Optional).
- Parameter optimisation module.

The input module transforms input data into trains of spikes. Spatio-temporal data (such as EEG, fMRI, climate) is entered into the main module – the 3D SNN cube (SNNc). Different types of data can used. This data is entered into *pre-designated spatially located* areas of the SNNc that correspond to the spatial location in the origin where data was collected (if there is such).

Learning in the SNN is performed in two stages:

- Unsupervised training, where spatio-temporal data is entered into relevant areas of the SNNc over time. Unsupervised learning is performed to modify the initially set connection weights. The SNNr will learn to activate same groups of spiking neurons when similar input stimuli are presented, also known as a *polychronization* effect (Izhikevich, 2004).

- Supervised training of the spiking neurons in the output module, where the same data that was used for unsupervised training is now propagated again through the trained SNN and the output neurons are trained to classify the spatio-temporal spiking pattern of the SNNc into pre-defined classes (or output spike sequences). As a special case, all neurons from the SNN are connected to every output neuron. Feedback connections from output neurons to neurons in the SNN can be created for reinforcement learning. Different SNN methods can be used to learn and classify spiking patterns from the SNNc, including the deSNN (Kasabov, Dhoble, et al, 2013) and SPAN models (Kasabov, 2014). The latter is suitable for generating motor control spike trains in response to certain patterns of activity of the Cube.

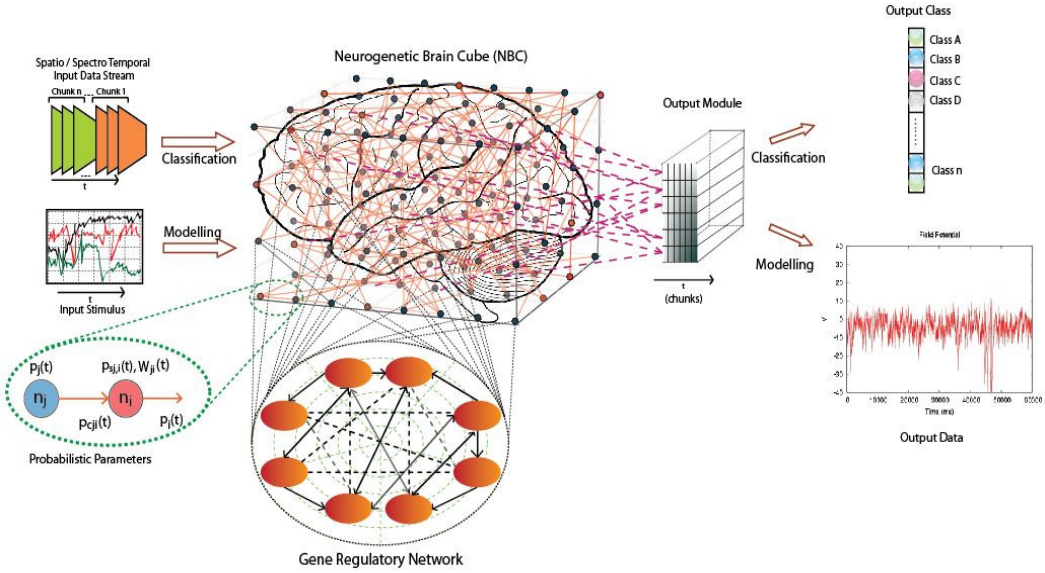


Fig. 11. A block diagram of the NeuCube architecture of a virtual spatio-temporal data machine (STDM), initially proposed for spatio-temporal brain data (from Kasabov, 2014).

Memory in the NeuroCube architecture is represented as a combination of the three types of memory described below, which are mutually interacting:

- Short-term memory, represented as changes of the PSP and temporary changes of synaptic efficacy;
- Long-term memory, represented as a stable establishment of synaptic efficacy – LTP and LTD;
- Genetic memory, represented as a genetic code.

In NeuCube similar activation patterns (called 'polychronous waves') can be generated in the SNNc with recurrent connections to represent short term memory. When using STDP learning, connection weights change to form LTP or LTD, which constitute long-term memory. Results of the use of the NeuCube suggest that the NeuCube architecture can be explored for learning long spatio-temporal patterns and to be used as associative memory. Once data is learned, the SNNc retains the connections as a long-term memory. Since the SNNc learns functional pathways of spiking activities represented as structural pathways of connections, when only a small initial part of input data is entered the SNNc will 'synfire' and 'chain-fire' *learned connection pathways* to reproduce *learned functional pathways.* Thus a NeuCube can be used as an associative memory and as a predictive system for event prediction when only some initial new input data is presented.

There are some challenging questions that need to be further explored, for example:

- What is the capacity of a NeuCube in terms of both spatial and temporal characteristics of the data?
- How much noise can be tolerated?
- How do we model transitions between spatio-temporal states triggered by external stimuli?
- Can we make the SNN learn and model mental states represented as brain data and their transitions?
- How do we model for example the transition between a SNNc state that represents a brain state of 'mild depression' to a state that represents 'happiness'; from a state of 'stroke' to a state of 'full cognitive recovery', from a state of 'clinical depression' to a state of 'suicidal mode'; from a state of 'anger' to a state of 'attack'?

These are some of the questions that need to be addressed as a future work.

7. Conclusion

This paper presents an overview of trends and directions of evolving connectionist systems (ECOS). The main goal of ECOS is to facilitate the creation of computational models and systems for adaptive learning and knowledge discovery from complex data. ECOS principles are derived from the integration of principles from neural networks, fuzzy systems, evolutionary computation, quantum computing and brain information processing. ECOS applications are manifold, but perhaps most welcome in the medical, environmental and health sciences, where the diagnostic phenomena are chaotic in nature and the data sets are massive and often incomplete. Massive (so called 'big') data sets with the characteristics just described need to be analyzed, virtually in real time, for prognoses to be made and solutions to the issues sought at a level of urgency. In this sense, evolving connectionist systems for adaptive learning and knowledge discovery can make a great contribution to the methodologies employed by the emerging trans-disciplinary, integrative, systemic and problem-solving science.

Acknowledgement

The work on this paper is supported by the Knowledge Engineering and Discovery Research Institute (KEDRI, http://www.kedri.aut.ac.nz). I was helped with the organization of this paper by Joyce D'Mello. More papers, data and software systems can be found at: http://www.kedri.aut.ac.nz, and: http://ncs.ethz/projects/evospike/.

References

1. Amari S (1967) A theory of adaptive pattern classifiers. IEEE Transactions on Electronic Computers, 16: 299-307

2. Amari, S. 1990. Mathematical Foundations of Neurocomputing. Proc. IEEEE, 78, 1143-63.

3. Angelov, P. (2002) Evolving Rule-based Models: A Tool for Design of Flexible Adaptive Systems. Heidelberg, Germany: Springer-Verlag.
4. Benuskova, L, and N.Kasabov, Computational neuro-genetic modelling, Springer, New York, 2007.
5. Bezdek, J. ed. (1987) Analysis of Fuzzy Information, vols. 1,2,3, CRC Press, Boca Raton, Florida.
6. Defoin-Platel, M., S.Schliebs, N.Kasabov, Quantum-inspired Evolutionary Algorithm: A multi-model EDA, IEEE Trans. Evolutionary Computation, vol.13, No.6, Dec.2009, 1218-1232
7. Knowledge-Based Systems Journal, Expert Systems, vol.1, issue 2, 1988, p.127.
8. Furuhashi, T., Hasegawa, T., Horikawa S., Uchikawa, Y. (1993) An Adaptive Fuzzy Controller Using Fuzzy Neural Networks, in: Proceedings of Fifth IFSA World Congress, pp.769-772.
9. Futschik, M. and Kasabov, N. (2002) Fuzzy clustering in gene expression data analysis, Proc. of the World Congress of Computational Intelligence WCCI'2002, Hawaii, May, 2002, IEEE Press
10. Gerstner, W. (1995) Time structure of the activity of neural network models, Phys. Rev 51: 738-758.
11. Hebb, D. (1949). The Organization of Behavior. New York, John Wiley and Sons.
12. Hodgkin, A. L. and A. F. Huxley (1952) A quantitative description of membrane current and its application to conduction and excitation in nerve. Journal of Physiology, 117: 500-544.
13. Hopfield, J., Pattern recognition computation using action potential timing for stimulus representation. Nature, 376:33–36, 1995.
14. Izhikevich, E. M. (2004) Which model to use for cortical spiking neurons? IEEE TNN, 15(5): 1063-1070.
15. Kasabov, N. and Shishkov, S. A connectionist production system with partial match and its use for approximate reasoning. Connection Science 5(3/4): 275-305 (1993)
16. Kasabov, N. Incorporating neural networks into production systems and a practical approach towards the realisation of fuzzy expert systems. Computer Science and Informatics 21(2): 26-34 (1991)
17. Kasabov, N. Hybrid Connectionist Fuzzy Production Systems - Towards Building Comprehensive AI. Intelligent Automation and Soft Computing 1(4): 351-360 (1995)
18. Kasabov, N. Connectionist fuzzy production systems. LNCS/AI, 847:114-128 (1994)
19. Kasabov, N. Hybrid connectionist production systems. Journal of Systems Engineering 3(1): 15-21 (1993)
20. Kasabov, N. Foundations of Neural Networks, Fuzzy Systems and Knowledge Engineering. Cambridge, Massachussets, MIT Press (1996) 550p
21. Kasabov N (1998) Evolving Fuzzy Neural Networks - Algorithms, Applications and Biological Motivation. In: Yamakawa, T. and G.Matsumoto (eds) Methodologies for the conception, design and application of soft computing, World Scientific, pp 271 – 274
22. Kasabov N (2001) Evolving fuzzy neural networks for on-line supervised/unsupervised, knowledge–based learning. IEEE Trans. SMC/ B, Cybernetics 31(6): 902-918.
23. Kasabov N (2003, 2007) Evolving connectionist systems, Springer Verlag, London, New York, Heidelberg (first edition 2003; second edition 2007)

24. Kasabov N, Kim J S, Watts M and Gray, A. (1997) FuNN/2- A Fuzzy Neural Network Architecture for adaptive learning and Knowledge Acquisition, Information Sciences - Applications, 101(3-4): 155-175
25. Kasabov N and Song Q (2002) DENFIS: Dynamic, evolving neural-fuzzy inference systems and its application for time-series prediction. IEEE Trans. on Fuzzy Systems, vol. 10:144 – 154
26. Kasabov, N., and Y. Hu (2010) Integrated optimisation method for personalised modelling and case study applications, Int. Journal of Functional Informatics and Personalised Medicine, vol.3,No.3,236-256.
27. Soltic, S. and Kasabov, N. (2010), "Knowledge extraction from evolving spiking neural networks with rank order population coding ," *International Journal of Neural Systems*, 20:6, pp. 437-445.
28. Kasabov, N. (2014) NeuCube: A Spiking Neural Network Architecture for Mapping, Learning and Understanding of Spatio-Temporal Brain Data, Neural Networks (2014), v.52,62-76, http://dx.doi.org/10.1016/j.neunet.2014.01.006
29. Kasabov, N., Liang, L., Krishnamurthi, R., Feigin, V., Othman, M., Hou, Z.,. Parmar, P. (2014). Evolving Spiking Neural Networks for Personalised Modelling of Spatio-Temporal Data and Early Prediction of Events: A Case Study on Stroke. Neurocomputing, vol.134, 269-279, 2014.
30. Kasabov, N. and E. Capecci (2014) Spiking neural network methodology for modelling, classification and understanding of EEG spatio-temporal data measuring cognitive processes, Information Sciences, 294, 565-575, 2015.
31. Kasabov, N., Dhoble, K., Nuntalid, N., & Indiveri, G. (2013). Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition. Neural Networks, 41, 188-201.
32. Song, S., K. Miller, L. Abbott et al., Competitive Hebbian learning through spike-timing-dependent synaptic plasticity, Nature Neuroscience, vol. 3, pp. 919–926, 2000.
33. Song, Q. and Kasabov, N. TWNFI- a transductive neuro-fuzzy inference system with weighted data normalisation for personalised modelling, Neural Networks, 19(10), 2006, 1591-1596.
34. Thorpe,S., Delorme, A., et al. (2001). Spike-based strategies for rapid processing. Neural Networks, 14(6-7),715-25.
35. Verstraeten, D., B. Schrauwen, M. D'Haene, and D. Stroobandt, An experimental unification of reservoir computing methods, Neural Networks, 20(3):391 – 403, 2007.
36. Watts, M. (2009) A Decade of Kasabov's Evolving Connectionist Systems: A Review, IEEE Trans. Systems, Man and Cybernetics- Part C: Applications and Reviews, vol.39, no.3, 253-269.
37. Widiputra, H., Pears, R., & Kasabov, N. (2011). Multiple time-series prediction through multiple time-series relationships profiling and clustered recurring trends. Springer LNAI 6635 (part 2), 161-172.
38. Wysoski, S., L. Benuskova, N. Kasabov, Evolving spiking neural networks for audiovisual information processing, Neural Networks, vol 23, 7, pp 819-835, 2010.
39. Yager R.R., Filev, D., Generation of fuzzy rules by mountain clustering, J. of Intelligent and Fuzzy Systems, 2, 209-19, 1994.

40. Yamakawa, T., E. Uchino, T. Miki and H.Kusanagi (1992) A Neo Fuzzy Neuron and Its Application to System Identification and Prediction of the System behaviour. Proceedings of the 2nd International Conference on Fuzzy Logic & Neural Networks, Iizuka, Japan, 477-483.
41. Zadeh L, (1965) Fuzzy Sets. Information and Control, vol.8, 338-353.
42. Zadeh LA (1988) Fuzzy Logic. IEEE Computer 21: 83 – 93.