Evolving Connectionist and Fuzzy-Connectionist Systems for On-line Adaptive Decision Making and Control

Nikola Kasabov

Department of Information Science
University of Otago, P.O Box 56, Dunedin, New Zealand
Phone: +64 3 479 8319, fax: +64 3 479 8311
nkasabov@otago.ac.nz

Keywords: evolving neural networks; fuzzy neural networks; intelligent information systems; on-line adaptive control; on-line decision making.

Abstract

The paper contains a discussion material and preliminary experimental results on a new approach to building on-line, adaptive decision making and control systems. This approach is called evolving connectionist systems (ECOS). ECOS evolve through incremental, on-line learning. They can accommodate any new input data, including new features, new classes, etc. New connections and new neurons are created during operation. The ECOS framework is illustrated here on a particular type of evolving neural networks - evolving fuzzy neural networks. ECOS are three to six orders of magnitude faster than the multilayer perceptrons, or fuzzy neural networks, trained with the backpropagation algorithm or with a genetic algorithm. ECOS are appropriate techniques to use for creating on-line, real-time, adaptive intelligent systems. This is illustrated on a case study problem of on-line, wastewater time-series flow prediction and cont rol. Possible real world applications of this approach are discussed.

1. Introduction: On-line, Adaptive Decision Making and Control

The complexity and the dynamics of many real-world problems, especially in engineering and manufacturing, require using sophisticated methods and tools for building on-line, adaptive decision making and control systems (OLADECS). Such systems should be able to 'grow' as they work. They should be able to build-up their knowledge and refine the model through interaction with the environment.

There are different methods and tools, based on soft computing that have been developed for decision making and control so far. They include: traditional statistical and probabilistic methods; neural networks [1,2,3,4,14]; fuzzy systems

[22]; genetic algorithms and evolutionary computing [20]; fuzzy neural networks and hybrid systems [5,6,9,12,13,16,21]. The traditional approach to building decision making and control systems assumes that a model is preliminary designed based on existing knowledge and/or data. The above methods and the resulting systems are usually concerned with the precision of the control or decision making model and only few of them have the ability to partially adapt during operation. But in many cases existing knowledge or data are either not sufficient to build an adequate and precise model of the process, or they change during the system's operation due to changes in the environment. In this case OLADECS are needed.

The following is a list of requirements to the contemporary OLADECS:

- to learn fast in an incremental, on-line mode;
- to accommodate both data and a priory knowledge (e.g., existing rules)
- to produce explanation about their decision on (e.g., extract rules);
- to adjust to changes in the operating environment, introducing new variables and features if needed
- if past data is available, to learn quickly from such data.

In this section a general framework of OLADECS, that addresses all the above requirements, is presented. Its realisation with the use of evolving connectionist systems (ECOS) and evolving fuzzy neural networks EFuNNs in particular is illustrated on a case study problem after the principles of ECOS and EFuNNs are introduced in the following sections.

A block diagram of OLADECS is given in fig.1. It consists of the following blocks:

- Pre-processing (filtering) block: data is processed (filtered) in this block (e.g. checking for consistency; feature extraction, calculating moving averages, selecting time-lags for a time-series).
- Neural network (NN), multi-modular learning block: it consists of many NNs that are continuously trained with data (both old, historical data, and new incoming data)
- Rule-based block for final decision - this block takes the produced by the NN outputs and applies expert rules. The rules take into account some other variables as inputs for which there might not be data available.
- Adaptation block - this block compares the output of the system with the desired-, or the real data, obtained over certain periods of time. The error is used to adjust/adapt the NN modules in a continuous mode.
- Rule extraction, explanation block - this block uses both extracted from the NN modules rules and rules from the final DM block to explain: (1) *what* the system currently 'knows' about the problem it is solving; (2) *why* a particular decision for a concrete input vector has been made.

The framework allows: feature selection, modelling, final multivariable decision-making, rule extraction and explanation and adaptation to changes in the operating environment and to new incoming data.

There are some neural network methods and techniques for adaptive learning developed so far. They include: incremental learning [3]; on-line learning [7]; real-time learning; life-long learning; growing neural networks [4]; neural networks trained with forgetting and pruning [8,15,19]. None of them though address all the listed above requirements to OLADECS.

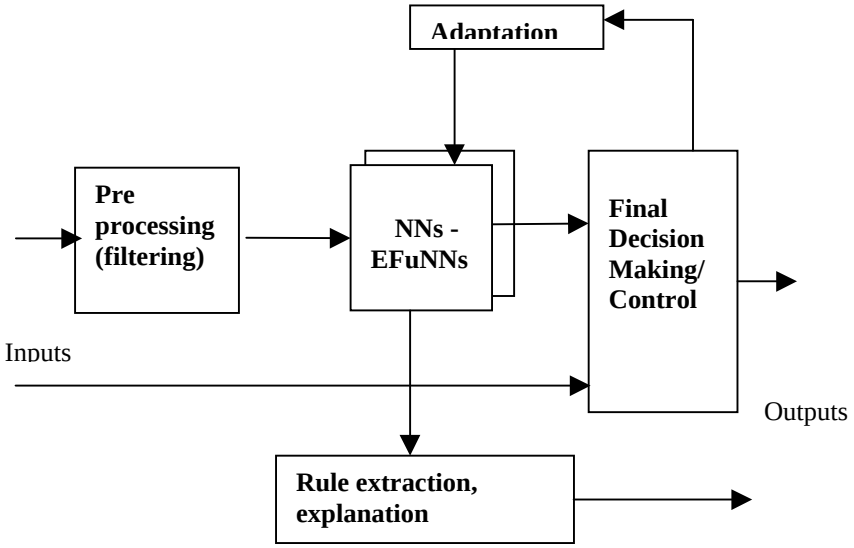

Figure 1: A block diagram of OLADECS

In the next sections the principles of ECOS and EFuNNs are presented and then applied to a case study OLADECS.

2. ECOS - Evolving Connectionist Systems

ECOS are systems that evolve in time trough interaction with the environment, i.e. an ECOS adjusts its structure with a reference to the environment (fig.2).

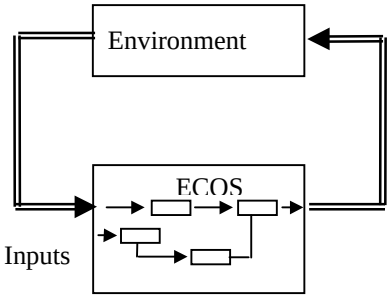

Fig.2. ECOS evolve through an interaction with the environment

A block diagram of the ECOS framework is given in [10]. ECOS are multi-level, multi-modular structures where many modules are connected with inter- and intra-connections. The evolving connectionist system does not have to have a 'clear' multi-layer structure. It has a modular 'open' structure.

Initially an ECOS contains nodes (neurons) with very little connections between them pre-defined through *prior* knowledge or 'genetic' information. These connections mainly connect modules of the initial connectionist structure. An initial set of rules can be inserted in this structure. Gradually, through self-organisation, the system becomes more and more 'wired'. The network stores different patterns (exemplars) from the training examples. A node is created and designated to represent an individual example if it is significantly different from the previous ones (with a level of differentiation set through dynamically changing parameters. The functioning of ECOS is based on the following general principles [10]:

(1) Input patterns are presented one by one, in a pattern mode, having not necessarily the same input feature sets. After each input example is presented, the ECOS either associates this example with an already existing rule (case) node, or creates a new one. A NN module, or a neuron is created when needed at any time of the functioning of the whole system. After the presentation of each new input example, the system is able to react properly on both new and old examples.

(2) The representation module evolves in two phases. In phase one input vector **x** is passed through the representation module and the case nodes become activated based on the similarity between the input vector and their input connection weights. If there is no node activated above a certain *sensitivity threshold (Sthr)* a new rule neuron (*rn*) is created and its input weights are set equal to the values of the input vector **x** and the output weights - to the desired output vector. In phase two, activation from either the winning case neuron ("one-out of-n" mode), or from all case neurons with activation above an *activation threshold (Athr)* ("many-of-on" mode) is passed to the next level of neurons. Evolving can be achieved in both supervised and unsupervised modes. In a supervised mode the final decision which class (e.g., phoneme) the current vector **x** belongs to, is made in the higher-level decision module that may activate an adaptation process. Then the connections of the representation nodes to the class output nodes, and to the input nodes are updated with the use of *learning rate coefficients lr1 and lr2,* correspondingly. If the class activated is not the desired one, then a new case node is created. The feedback from the higher level decision module goes also to the feature selection and filtering part. New features may be involved in the current adaptation and evolving phase. In an unsupervised mode a new case node is created if there is no existing case node or existing output node that are activated above *Sthr* and an *output threshold Othr* respectively. The parameters *Sthr, lr1, lr2, Errthr, Athr and Othr* can change dynamically during learning.

(3) Along with growing, an ECOS has a pruning rule defined. It allows for removing neurons and their corresponding connections that are not actively involved in the functioning of the ECOS thus making space for new input patterns. Pruning is based on local information kept in the neurons. Each neuron in ECOS keeps a 'track' of its 'age', its average activation over the whole life span, the error

it contributes to, and the density (in Euclidean sense) of the surrounding area of neurons. Pruning is performed through the fuzzy rule:

IF case node (j) is OLD, and average activation of (j) is LOW, and the density of the neighbouring area of neurons is HIGH or MODERATE, and the sum of the incoming or outgoing connection weights is LOW, THEN the probability of pruning node (j) is HIGH.

(4) The case neurons are spatially organised and each neuron has its relative spatial dimensions in regards to the rest of the neurons based on their reaction to the input patterns. If a new neuron is created when the input vector **x** was presented, the new neuron is allocated close to the neuron which had the highest activation to the input vector **x**.
(5) There are two global modes of learning in ECOS:
(a) *Active learning mode* - learning is performed when a stimulus (input pattern) is presented and kept active.
(b) *Eco learning mode* : learning is performed when there is no input pattern presented at the input of the ECOS. In this case the process of further elaboration of the connections in ECOS is done in a passive learning phase, when existing connections that store previously 'seen' input patterns are used as eco training examples. The connection weights that represent stored input patterns are now used as compressed input patterns for training other modules in ECOS. This type of learning with the use of 'echo' data is called here eco training. There are two types of eco training that are applicable to classification tasks: (1) cascade eco training; (2) sleep eco training. In *cascade eco training* a new NN module is created when a new class data is presented. The module is trained on the positive examples of this class, plus the negative examples of the following different class data, and on the negative examples of previously stored patterns in previously created modules. In the sleep eco training, modules are created with positive examples only when data is presented. Then the modules are trained on the stored in the other modules patterns as negative examples.
(6) ECOS provide explanation information extracted from the structure of the NNs. Each case (rule) node is interpreted as an IF-THEN rule as it is the case in the FuNN fuzzy neural networks [12,13].
(7) ECOS are biologically inspired. Some biological motivations for evolving systems are given in [17,18].
(8) The ECOS framework can be applied to different types of NNs, neurons, activation functions, etc. One realisation that uses the ECOS framework is the evolving fuzzy neural networks EFuNNs and the EFuNN algorithm as given in [11].

3. Fuzzy Neural Networks FuNNs and Evolving Fuzzy Neural Networks EFuNNs

3.1. The FuNN Architecture and its Functionality

Fuzzy neural networks are neural networks that realise a set of fuzzy rules and a fuzzy inference machine in a connectionist way [5,6,9,12,13,16,21]. FuNN is a fuzzy neural network introduced in [12] and then developed as FuNN/2 in [13]. It is a connectionist feed-forward architecture with five layers of neurons and four layers of connections. The first layer of neurons receives the input information. The second layer calculates the fuzzy membership degrees to which the input values belong to predefined fuzzy membership functions, e.g. small, medium, large. The third layer of neurons represents associations between the input and the output variables, fuzzy rules. The forth layer calculates the degrees to which output membership functions are matched by the input data and the fifth layer does defuzzification and calculates values for the output variables. A FuNN has both the features of a neural network and a fuzzy inference machine. A simple FuNN structure is shown in Figure 2. The number of neurons in each of the layers can potentially change during operation through growing or shrinking. The number of connections is also modifiable through learning with forgetting, zeroing, pruning and other operations [12].

The membership functions, used in FuNN to represent fuzzy values, are of triangular type, the centres of the triangles being attached as weights to the corresponding connections. The membership functions can be modified through learning.

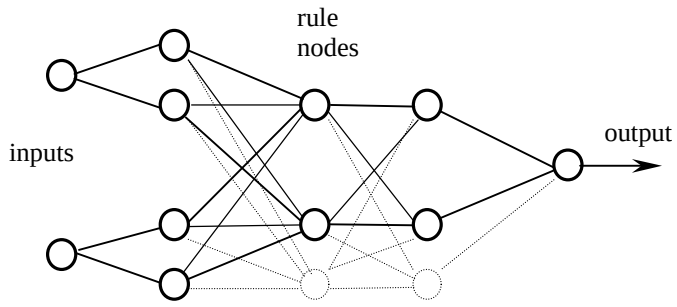

Figure 3: A FuNN structure of 2 inputs (input variables), 2 fuzzy linguistic terms for each variable (2 membership functions). The number of the rule nodes is fixed in a FuNN structure and is evolving in the EFuNN structure.

Several training algorithms have been developed for FuNN [12,13]:
(a) A modified back-propagation (BP) algorithm that does not change the input and the output connections representing the membership functions.

(b) A modified BP algorithm that utilises structural learning with forgetting, i.e. a small forgetting ingredient, e.g. 10^{-5}, is used when the connection weights are updated (see [8]).
(c) A modified BP algorithm that updates both the inner connection layers and the membership layers. This is possible when the derivatives are calculated separately for the two parts of the triangular membership functions. These are also the non-monotonic activation functions of the neurons in the condition element layer.
(d) A genetic algorithm for training [20].
(e) A combination of any of the methods above used in different time intervals as part of a single training procedure.

Several algorithms for rule extraction from FuNNs have been developed and applied [12,13]. One of them represents each rule node of a trained FuNN as an IF-THEN fuzzy rule.

FuNNs have several advantages when compared with the traditional connectionist systems or with the fuzzy systems:
(a) They are both statistical and knowledge engineering tools.
(b) They are robust to catastrophic forgetting, i.e. when they are further trained on new data, they keep a reasonable memory of the old data.
(c) They interpolate and extrapolate well in regions where data is sparse.
(d) They accept both real input data and fuzzy input data represented as singletons (centres of gravity of the input membership functions))

3.2. EFuNNs - Evolving FuNNs

EFuNNs are FuNN structures that evolve according to the ECOS principles. All nodes in an EFuNN are created during learning. The nodes representing membership functions (fuzzy label neurons) can be modified during learning. As in FuNN, each input variable is represented here by a group of spatially arranged neurons to represent different fuzzy domain areas of this variable. For example three neurons can be used to represent "small", "medium" and "large" fuzzy values of a variable. Different membership functions can be attached to these neurons (triangular, Gaussian, etc.). New neurons evolve in this layer if for a given input vector the corresponding variable value does not belong to any of the existing membership functions to a membership degree greater than a membership threshold. A new fuzzy label neuron, or an input variable neuron, can be created during the adaptation phase of an EFuNN.

The EFuNN algorithm for evolving EFuNNs is presented in [11]. A new rule node rn is created and its input and output connection weights are set as follows: W1(rn)=EX; W2(rn) = TE, where TE is the fuzzy output vector for the current example EX. In case of "one-of-n" EFuNNs, the maximum activation of a rule node is propagated to the next level. Saturated linear functions are used as activation functions of the fuzzy output neurons. In case of "many-of-n" mode, all the activation values of rule (case) nodes that are above an activation threshold *Athr* are propagated.

4. Applying ECOS and EFuNNs to Building OLADECS: A Case Study Problem

A case study problem is taken here to illustrate the potential of the ECOS and the EFuNNs. The problem is to predict a waste water flow coming from three pumps into a sewage plant (see [12] for a description of the problem and the WWW http://divcom.otago.ac.nz:800/com/infosci/KEL/home.htm for the data set). The flow is measured every hour. It is important to be able to predict the flow as the collecting tank has a limited capacity (in this case it is 650 cubic meters) and a sudden overflow will cause bacteria, that clean the water, to be thrown away. As there is very little data available before the control system is installed and put in operation, the control system has to be adaptive and learn the dynamics of the flow as it operates. This is a task for an OLADECS.

Here one EFuNN that has 4 inputs, F(t), F(t-1), MA12h(t), MA24h(t), and one output F(t+1) is evolved from the time series data. The resulting EFuNN after 500 data points, has 397 rule nodes (Sthr=0.9; Errthr=0.05; lr=0; no pruning). The MSSE over a test set of the last 100 data points is 0.0068 (normalised data is used) - see fig.4.

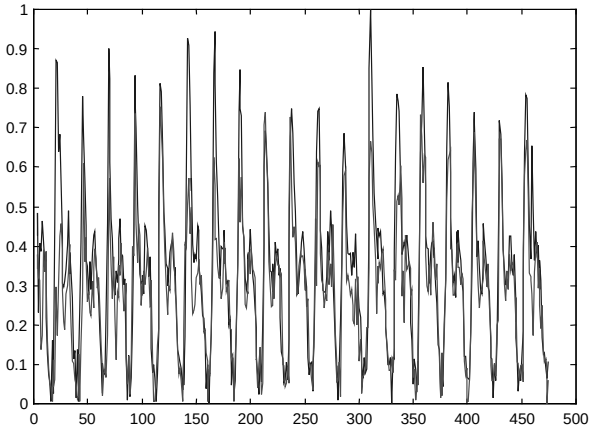

Figure 4: AnEFuNNs is evolved from the flow F(t), the moving average 12 hours MA12h(t) and the moving average 24 hours MA24h(t) data. Here the real versus the predicted one day ahead values of the flow are plotted.

It is seen from fig.4 that in the beginning the EFuNN could not generalise well on the next hour flow value, but after learning (accommodating) about 400 data points, it produces a generalisation that is much better than the generalisation on the same test data when a MLP is used that had 5 inputs, two hidden layers with 6 nodes in each of them and one output, trained with the BP algorithm for 20,000 epochs (MSSE=0.044, see [12]). The EFuNN used 4 orders of magnitude less time for training per example at average than the MLP.

5. Conclusions and Directions for Further Research

This paper applies a framework ECOS for evolving connectionist systems, and an evolving fuzzy neural network EFuNN to building on-line adaptive decision making and control systems (OLADECS). A case study, real problem has been used to illustrate the potential of this approach. Several applications of ECOS and EFuNNs, some of them already shown, will be explored in the future: adaptive speech recognition [11]; OLADECS for financial applications; OLADECS for mobile robots; OLADECS for multi-modal information retrieval.

References

1. Amari, S. and Kasabov, N. eds (1997) Brain-like computing and intelligent information systems, Springer Verlag
2. Arbib, M. (ed) (1995) *The Handbook of Brain Theory and Neural Networks.* The MIT Press
3. Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., Rosen, D.B., FuzzyARTMAP: A neural network architecture for incremental supervised learning of analog multi-dimensional maps, IEEE Trans. on Neural Networks,vol.3, No.5 (1991), 698-713
4. Fritzke, B., Growing cell structures - a self- organising network for unsupervised and supervised learning, Neural Networks, vol.7, No.9 (1994) 1441-1460.
5. Hashiyama, T., Furuhashi, T., Uchikawa, Y.(1992) A Decision Making Model Using a Fuzzy Neural Network, in: *Proceedings of the 2nd International Conference on Fuzzy Logic & Neural Networks*, Iizuka, Japan, 1057-1060.
6. Hauptmann, W., Heesche, K. (1995) A Neural Net Topology for Bidirectional Fuzzy-Neuro Transformation, in *: Proceedings of the FUZZ-IEEE/IFES* , Yokohama, Japan, 1511-1518.
7. Heskes, T.M., Kappen, B. (1993) On-line learning processes in artificial neural networks, in: Math. foundations of neural networks, Elsevier, Amsterdam, 199-233
8. Ishikawa, M. (1996) "Structural Learning with Forgetting", Neural Networks 9, 501-521.
9. Jang, R. (1993) ANFIS: adaptive network-based fuzzy inference system, *IEEE Trans. on Syst.,Man, Cybernetics*, 23(3), May-June 1993, 665-685
10. Kasabov, N. ECOS: A framework for evolving connectionist systems and the eco learning paradigm,Proc. of ICONIP'98, Kitakyushu, Oct. 1998
11. Kasabov, N. Evolving Fuzzy Neural Networks - Algorithms, Applications and Biological Motivation, in Proc. of Iizuka'98, Iizuka, Japan, Oct.1998
12. Kasabov, N .(1996) *Foundations of Neural Networks, Fuzzy Systems and Knowledge Engineering*, The MIT Press, CA, MA.

13. Kasabov, N., Kim J S, Watts, M., Gray, A (1997) FuNN/2- A Fuzzy Neural Network Architecture for Adaptive Learning and Knowledge Acquisition, *Information Sciences - Applications*, 1997, 3(2)
14. Kohonen, T. (1990) The Self- Organizing Map. Proceedings of the IEEE, vol.78, N-9, pp.1464-1497.
15. Le Cun, Y., J.S. Denker and S.A. Solla (1990) "Optimal Brain Damage", in: Touretzky,ed.,*Advances in Neural Inform. Proc. Systems*, Morgan Kaufmann, 2, 598-605.
16. Lin, C.T. and C.S. G. Lee, "Neuro Fuzzy Systems", Prentice Hall (1996).
17. McClelland, J., B.L. McNaughton, and R.C. Reilly (1994) "Why there are Complementary Learning Systems in the Hippocampus and Neocortx: Insights from the Successes and Failures of Connectionist Models of Learning and Memeory", CMU Technical Report PDP.CNS.94.1, March
18. Quartz, S.R., and Senowski, T.J. The neural basis of cognitive development: a constructivist manifesto, Behavioral and Brain Science, in print
19. Reed, R. (1993) "Pruning algorithms - a survey", IEEE Trans. Neural Networks, 4 (5) 740-747.
20. Watts, M., and Kasabov, N. Genetic algorithms for the design of fuzzy neural networks, in Proc. of ICONIP'98, Kitakyushu, Oct. 1998
21. Yamakawa, T., H. Kusanagi, E. Uchino and T.Miki, (1993) "A new Effective Algorithm for Neo Fuzzy Neuron Model", in: Proceedings of Fifth IFSA World Congress, 1017-1020
22. Zadeh, L. 1965. Fuzzy Sets, *Information and Control*, vol.8, 338-353.