Evolutionary Computation for Dynamic Parameter Optimisation of Evolving Connectionist Systems for On-line Prediction of Time Series with Changing Dynamics

Nikola Kasabov[1], Qun Song[1] and Ikuko Nishikawa[2]
[1] Knowledge Engineering and Discovery Research Institute
Auckland University of Technology
Private Bag 92006, Auckland 1020, New Zealand
Emails: nkasabov@aut.ac.nz; qsong@aut.ac.nz;
[2] Computer Science Department, Ritsumeikan University, Japan

***Abstract*- The paper describes a method of using evolutionary computation technique for parameter optimisation of evolving connectionist systems (ECOS) that operate in an on-line, life-long learning mode. ECOS evolve their structure and functionality from an incoming stream of data in either a supervised-, or/and in an unsupervised mode. The algorithm is illustrated on a case study of predicting a chaotic time-series that changes its dynamics over time. With the on-line parameter optimisation of ECOS, a faster adaptation and a better prediction is achieved. The method is practically applicable for real time applications.**

I. INTRODUCTION – PARAMETER OPTMISATION OF ON-LINE LEARNING SYSTEMS

Evolving processes are difficult to model because some of their parameters may not be known a priori; unexpected perturbations or changes may happen at certain time of their development; they are not strictly predictable in a longer term. Thus, modelling of such processes is a challenging task with a lot of practical applications in life sciences and engineering.

One paradigm introduced for modelling evolving processes is called evolving connectionist systems (ECOS) [1-3]. ECOS evolve their structure and functionality over time from incoming data. ECOS need also to have evolving, adapting parameter values that would respond to changes in the modelled processes. In practice, evolving processes often change their dynamics, e.g. a stock index series move from one state (e.g. pseudo-periodic), to another (e.g. chaotic).

The paper introduces a method and an algorithm for on-line parameter optimisation of ECOS that is efficient especially when the dynamics of the modelled (predicted) time series change over time. This is a continuation of the developed GA-optimisation method for parameter optimisation of ECOS applied in an off-line learning mode on a classification task [4].

II. PRINCIPLES OF EVOLVING CONNECTIONIST SYSTEMS (ECOS)

Evolving connectionist systems (ECOS) are multi-modular, connectionist architectures that facilitate modelling of evolving processes and knowledge discovery [1-3]. An ECOS may consist of many evolving connectionist modules.

An ECOS is a neural network that operates continuously in time and adapts its structure and functionality through a continuous interaction with the environment and with other systems according to: (i) a set of parameters P that are subject to change during the system operation; (ii) an incoming continuous flow of information with unknown distribution; (iii) a goal (rationale) criteria (also subject to modification) that is applied to optimise the performance of the system over time.

The set of parameters P of an ECOS can be regarded as a chromosome of "genes" of the evolving system and evolutionary computation can be applied for their optimisation [4].

The evolving connectionist systems presented in [1-3] have the following specific characteristics:

1) They evolve in an open space, not necessarily of fixed dimensions.

2) They learn in on-line, incremental, fast learning - possibly through one pass of data propagation.

3) They learn in a life-long learning mode.

4) They learn as both individual systems, and as part of an evolutionary population of such systems.

5) They have evolving structures and use constructive learning.

6) They learn locally and locally partition the problem space, thus allowing for a fast adaptation and tracing the evolving processes over time.

7) They facilitate different kind of knowledge representation and extraction, mostly - memory based, statistical and symbolic knowledge.

The evolving connectionist models presented in [1-3] are knowledge-based models, facilitating Zadeh-Mamdani fuzzy rules (EFuNN, HyFIS), Takagi-Sugeno fuzzy rules (DENFIS), on-line fuzzy clustering (ECM).

One of the ECOS models is called evolving fuzzy neural network (EfuNN) [1-3] and used in this paper. A block diagram of EfuNN is shown in fig.1.

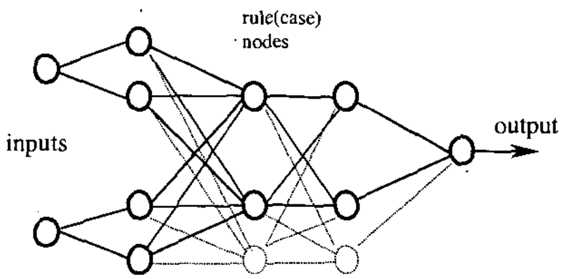

Figure 1. A block diagram of EfuNN

EfuNN consists of 5 layers – input layer, fuzzy input membership functions layer; rule (case) node layer; fuzzy output membership functions layer; and output layer.

There are two distinct phases of ECOS operation. In the first-, the learning phase, data vectors are fed into the system one by one with their known classes. The learning sequence of each iteration is described in fig.2.

Start training and EfuNN: Set initial values for the system parameters: number of membership functions; initial sensitivity thresholds (default Sj=0.9) also used as a radius of the receptive filed for the rule node R=1-Sj; error threshold E; aggregation parameter Nagg - number of consecutive examples after each aggregation is performed; pruning parameters OLD an Pr; a value for m (in m-of-n mode); maximum radius limit Rmax; thresholds T_1 and T_2 for rule extraction.

Set the first rule node r_0 to memorise the first example

(x,y): $W1(r_0)=x_f$, and $W2(r_0)=y_f$;

Loop over presentations of new input-output pairs (x,y):

{ Evaluate the local normalised fuzzy distance D between x_f and the existing rule node connections W1

Calculate the activation A1 of the rule node layer. Find the closest rule node r_k (or the closest m rule nodes in case of m-of-n mode) to the fuzzy input vector x_f for which $A1(r_k) >= S_k$ (sensitivity threshold for the node r_k),

if there is no such a node, create a new rule node for (x_f, y_f),

else

Find the activation of the fuzzy output layer A2=W2.A1(1-D(W1,x_f))) and the normalised output error Err= ‖y- y'‖ / Nout.

if Err > E, create a new rule node to accommodate the current example (x_f, y_f), *else*

Update W1 (r_k) and W2(r_k) according to (2) and (3) (in case of *m-of-n* system update all the *m* rule nodes with the highest A1 activation).

Apply *aggregation* procedure of rule nodes after each group of N_{agg} examples are presented

Update the values for the rule node r_k parameters S_k, R_k, Age(r_k), TA (r_k).

Prune rule nodes if necessary, as defined by pruning parameters.

Extract rules from the rule nodes }

Figure 2. The algorithm for training an EFuNN

The above described EfuNN has several parameters that need to be optimized according to the data set used. These are:

1) Maximum sensitivity threshold (radius of the receptive field - Rmax)

2) m-of-n value

3) Error threshold E.

An algorithm for this task is presented in the next section.

III. On-line Parameter Optimisation of Ecos Based on Evolutionary Computation

ECOS learn both in a supervised and unsupervised mode. There are two types of ECOS' parameters; the first one consists of parameters that change through the learning phase; the second one constitutes parameters that do not change through the learning process but defining it. For the EFuNN structure the parameters that change during learning are the rule nodes and their connection weight W1 and W2. The parameters that are predefined before learning are: 1. number of membership functions; 2.value for m-of-n parameter; 3.error threshold; 4.maximum receptive field; 5.rule extraction threshold; 6.numbers of examples for aggregation; and 7.pruning parameters. Here we focus on the optimisation of some of the second type of parameters, by the use of an evolutionary computation algorithm as an additional "loop" of the learning process.

Once the ECOS parameter values are fixed, ECOS learn to attain the best performance at the given set of parameter values, that is, at the given point in the parameter space. Given different parameter sets, each ECOS attains a different performance level with different connection weights learned. ECOS reaches the best performance at some optimal point in the area of the parameter space. Our aim is to let ECOS search for this optimal point in the parameter space along with their training on new data. The optimal point generally depends on the environment, i.e. on the input-output data that is presented. Therefore, once the environment changes, ECOS "needs" to search for a new optimal point in the parameter space.

The on-line search for the optimal parameter values at each point in time is done here through evolutionary computation (EC).EC comprises algorithms based on a heuristic approach [5-19]. EC enable an effective search to be applied to multiple individuals in parallel. The search converges after a number of iterations to an optimal or some sub-optimal point in the complex landscape of the parameter values defined by a fitness function. The diversity of multiple ECOS individuals makes it possible to keep not only the present optimal point, but also keep some candidates suitable for a future changed environment.

The framework of the proposed on-line parameter optimisation procedure is as follows:

1) We define certain time window during which each ECOS in a population of individuals learns via supervised and unsupervised algorithm with fixed parameter values.

2) At the end of the time window, each ECOS is evaluated by the root mean square error (RMSE) of the prediction for the data over the time window or on some other testing data set.

3) The obtained evaluation is used as a fitness value of the individual in a genetic algorithm (GA), or in another EC algorithm.

4) The GA operators of reproduction, crossover and mutation are applied to the genotype, which codes parameter values of each ECOS, to produce the next generation of population.

5) The newly created individuals (ECOS) inherit their rule node information from their parent. At the same time, a certain number of data from the time window are stored and used to evaluate the new ECOS individuals.

6) The new population of ECOS starts learning with fixed parameter values on the data composed of stored preceding data and some new data.

7) The learning process with fixed parameter values continues for the window width, and the GA loop is iterated every window width. The GA fitness is defined by the data in a current time window, which "moves" with 50 to 90% overlap.

For the successful performance of both the on-line learning and the on-line parameter optimisation, the characteristic time scale of the learning of individual ECOS should be shorter than one generation of GA, which should be shorter than the time scale of the environmental change.

In [1] a block diagram of a GA parameter optimisation procedure when ensembles (populations) are created from an ECOS and the best ECOS selected. This ECOS is used to generate the new ensemble of ECOS through parameter value perturbation.

IV. On-line Parameter Optimization by Evolutionary Computation for Chaotic Dynamic Time Series Prediction

In the present paper, on-line optimisation is applied to the time series prediction, which nature may change in a certain time scale. The on-line learning with parameter optimisation is demonstrated on the Mackey-Glass (MG) time series prediction task. The MG time series is generated with a time-delay differential equation as follows:

$$\frac{d\,x(t)}{d\,t} = \frac{0.2\,x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1\,x(t) \qquad (1)$$

where τ is a time delay system control parameter. This time series is well known and shows chaotic dynamics in some range of τ. Chaos is a dynamic system, which is seemingly irregular in a sense that there is no fixed point to converge. The prediction problem of the time series generated by (1) is as follows: given data vectors, $[x(t\text{-}18), x(t\text{-}12), x(t\text{-}6), x(t)]$ as inputs, to predict the future value $x(t+6)$, under the initial conditions $x(0) = 1.2$, $x(t) = 0$ for $t < 0$. The change of the environment for the EFuNN learning is given by the step-wise change of the τ value. The change of τ causes the change of the input-output mapping to be learned by EFuNN, which is attained by the different optimal parameter values.

In the following simulation, the MG equation given by (1) is used with $\tau = 17$ for the initial incoming data, after which it changes to 19. The chaotic nature and the change of its attractor is shown in Fig. 2a, which is a phase map ($x(t)$, $x(t+6)$) for $\tau = 17$ and 19 respectively.

The similar structures of the two-phase maps indicate that the attractors are not drastically changed by the change of τ from 17 to 19, still there are differences that may cause the ECOS system to fail to predict properly the time series with the changed parameter value. In the simulation, EFuNN first learns and optimises the parameters to predict the MG series with $\tau = 17$, and at the time point t = 600 the MG parameter τ changes to 19. Fig. 2b. shows the MG time series plotted at every integer time steps.

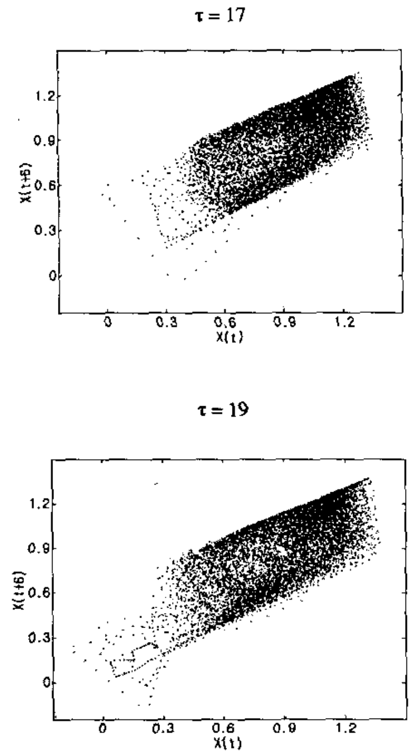

Fig. 2a. Phase maps of the Mackey-Glass data with $\tau = 17$ (and $\tau = 19$ respectively

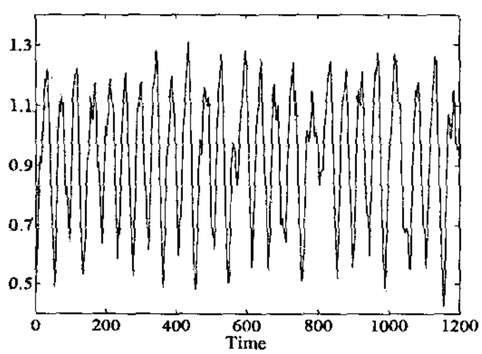

Fig. 2b. The Mackey-Glass MG time series data with $\tau = 17$ (time points from 0 till 599) and $\tau = 19$ (from time point t = 600 onwards)

he time window for the learning and for the evolutionary loop is set to 200 data points, with 90% overlap between consecutive windows. The size of the EFuNN population is 12. The optimized parameters by the GA are: 1. Number of the highest activated rule nodes used for both the training and the recall procedure (m-of-n, value range: 1 – 8); 2. Maximum value of the receptive field (value range: 0.75 – 0.95); 3. Error threshold (value range: 0.05 – 0.25). The first parameter is coded by 3-bit binary digits and the following two parameters are coded by 6-bit binary digits. The parallel loops of learning and parameter optimization are executed by the following procedure.

step1: These 3 parameter values are randomly selected from certain ranges to compose an initial population of 12 EFuNNs (TABEL 1). Take the first 200 data for the first learning phase of each of the 12 EFuNNs.

step2: Each EFuNN learns on the 200 data to create new rule nodes and update existing rule nodes under the predefined parameter values.

step3: After the learning phase on the first window of data, each individual EFuNN is evaluated by its RMSE used as a fitness function (see Table 1). Conventional GA operations of reproduction by a roulette selection based on this fitness value, and crossover and mutation, are applied to generate a new population.

step4: New individuals inherit the rule nodes from their parent individual.

step5: Take the next 20 data points and previous 180 data points to compose a new learning data set.

step6: Return to step2 and repeat steps 2-5 throughout the whole data set.

The simulation results are shown in Fig.3 and Fig.4, where solid lines represent the results or parameters for the EFuNN learning with GA parameter optimization and dashed lines are for the EFuNN with fixed parameters. Fig.3 shows the RMSE of two EFuNN learning procedures on the whole data set. The upper line (dashed) is RMSE of the best EFuNN from the population of 12 (the 6^{th}), which have the same parameter values as the EFuNNs in step1 of the preceding procedure but without parameter optimization, and the lower line (solid) is the RMSE of the EFuNN with GA on-line parameter optimization. It is obvious that the EFuNN with the on-line parameter optimization adapts faster to the changes in the dynamics of the time-series and its RMSE is much smaller.

Fig. 4. shows the number of rule nodes in the two EFuNNs, while Fig. 5., Fig. 6. and Fig. 7. show how the parameters m-of-n, Maxfield and Errthr change over time as a result of the GA optimization procedure.

As a result of the parameter optimisation after the time moment t = 600 there are more rule nodes created in the optimised EFuNN than in the EFuNN with fixed parameter values.

TABLE 1
INITIAL PARAMETERS FOR THE INITIAL 12 EFUNNS IN THE STATRTING POPULATION AND THE RMSES ERROR OBTAINED AFTER TRAINING THE EFUNNS ON THE FIRST WINDOW OF INCOMING DATA

Ind	Mofn	Maxfield	Errthr	RMSE
1	5	0.9405	0.0976	0.0936
2	4	0.9278	0.2024	0.1094
3	5	0.9151	0.1389	0.1264
4	2	0.9341	0.1992	0.1085
5	4	0.9373	0.2341	0.1293
6	7	0.7627	0.1198	0.0868
7	2	0.7786	0.0913	0.1036
8	1	0.804	0.0913	0.116
9	4	0.8389	0.2373	0.106
10	2	0.9183	0.1548	0.1181
11	6	0.9183	0.0532	0.0906
12	6	0.9151	0.1516	0.1032

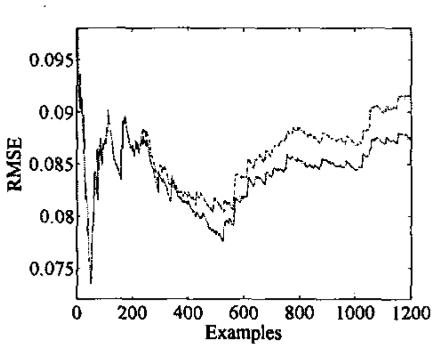

Fig. 3. The RMSE of EFuNN learning without (upper graph) and with (lower graph) on-line optimization. Note that at time point 600 the dynamics of the time series changed (from $\tau = 17$ to $\tau = 19$).

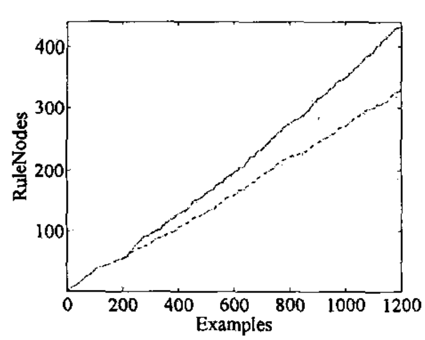

Fig. 4. The change in the number of rule nodes over time in the two EfuNN models (without optimisation – dotted line, and with optimisation – solid line)

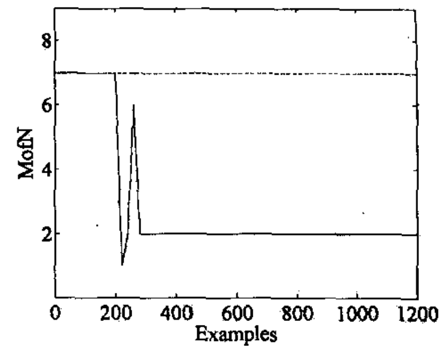

Fig. 5. The change of the values of the parameter "M-of-n" for the optimised EFuNN (for the other EFuNN this parameter has a fixed value of 7)

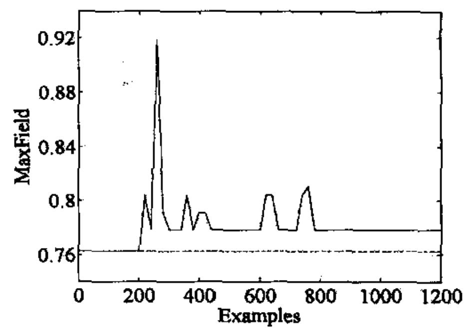

Fig. 6. The parameter "Maxfield" values in the two EfuNNs

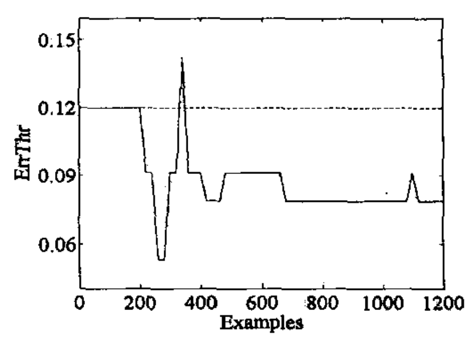

Fig. 7. The change of the parameter "Errthr" values over time for the optimised EFuNN.

It can be seen that the "m-of-n" parameter stabilises its optimal value to 2. This value will not only guarantee a better RMSE, but a faster recall procedure as only 2 rule node activation values will be calculated for each input vector, rather than 7 in the other case. Fig. 6 shows the change in the parameter values of the parameter "Maximum field" for the optimised EFuNN that stabilises around 0.78 while the fixed value is 0.76. Fig. 7. shows that the parameter Errthr stabilises at a value around 0.08. For not-optimised one the value is fixed at 0.12.

V. CONCLUSIONS

Further development of the method presented includes on-line optimisation of the set of input variables (feature optimisation and feature selection). The ECOS paradigm has a broad spectrum of applications in life sciences, especially in bio-informatics [20] and brain study [21] as the paradigm adopts principles from both areas. It is also a power tool for adaptive prediction, decision making and control. The optimisation algorithm presented in this paper for ECOS parameter optimisation makes the main characteristics of ECOS, such as adaptive learning and rule extraction, even more useful for a large scope of on-line applications. The proposed algorithm can be applied to other evolving connectionist models based on clustering, such as RBF (see for example the ZISC system [22]).

ACKNOWLEDGEMENT

The research presented in the paper is funded by the New Zealand Foundation for Research, Science and Technology under the grant: NERF/AUTX02-01.

REFERENCES

[1] N. Kasabov, Evolving connectionist systems: Methods and Applications in Bioinformatics, Brain study and intelligent machines, Springer Verlag, London, New York, Heidelberg, 2002.

[2] N. Kasabov, "Evolving fuzzy neural networks for on-line supervised/unsupervised, knowledge-based learning," IEEE Trans. SMC – part B, Cybernetics, vol.31, No.6, 902-918, December 2001.

[3] Kasabov, N. and Q. Song, "DENFIS: Dynamic, evolving neural-fuzzy inference systems and its application for time-series prediction," IEEE Trans. on Fuzzy Systems, vol.10, No.2, 144-154, April 2002.

[4] N. Kasabov and Q. Song, "GA-Optimisation of evolving connectionist systems for classification with a case study from bioinformatics," Proc. of ICONIP'2002, Singapore, November, 2002.

[5] X.Yao (1993) Evolutionary artificial neural networks, Int. Journal of Neural Systems, vol.4, No.3, 203-222

[6] X.Yao (1999) Evolving artificial neural networks, Proc. IEEE, 87 (9): 1423-1447, September 1999

[7] Fogel, D., Fogel, L. and Porto, V. (1990) Evolving neural networks, Biological Cybernetics, vol.63,487-493

[8] Linkens, D.A. Nyongesa, H.O. Genetic algorithms for fuzzy control.2. Online system development and application, Control Theory and Applications, IEE Proceedings, 177-185, Volume: 142, Issue: 3, May 1995

[9] Chia-Feng Juang, A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms, IEEE Transactions on Fuzzy Systems, 155-170, Volume: 10, Issue: 2, Apr 2002

[10] Mishra, S.; Dash, P.K.; Hota, P.K.; Tripathy, M.; Genetically optimized neuro-fuzzy IPFC for damping modal oscillations of power system, Power Systems, IEEE Transactions on , Volume: 17 Issue: 4 , Nov 2002, 1140 -1147

[11] Tippayachai, J. Ongsakul, W. Ngamroo, I., Parallel micro genetic algorithm for constrained economic dispatch, IEEE Transactions on Power Systems, 790-797, volume: 17, Issue: 3, Aug 2002

[12] da Silva, W.G. Acarnley, P.P. Finch, J.W. ; Application of genetic algorithms to the online tuning of electric drive speed controllers, IEEE Transactions on Industrial Electronics, 217-219, Volume: 47, Issue: 1, Feb 2000

[13] Rajapakse, A. Furuta, K. Kondo, S., Evolutionary learning of fuzzy logic controllers and their adaptation through perpetual evolution, , IEEE Transactions on Fuzzy Systems, 309-321, Volume: 10, Issue: 3, Jun 2002

[14] Baeck, T. *Evolutionary algorithm in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*, Oxford University Press, New York (1995)

[15] Holland, J. H. *Adaptation in natural and artificial systems*, The University of Michigan Press, Ann Arbor, MI (1975)

[16] Koza, J. R. *Genetic Programming*, MIT Press (1992)

[17] Lichtfuss, H. J. *Evolution eines Rohrkrummers*, Technische Universitat Berlin, Diplomarbeit (1965)

[18] Schwefel, H.-P. "Projekt MHK-Staustrahlrohr: Experimentelle Optimierung einer Zweiphasenduse" *Teil I. Technischer Bericht 11.034/68, 35*, AEG Forschungsinstitute, Berlin (October 1968)

[19] Goldberg, D. E. *Genetic Algorithms in Search, Optimization and machine Learning*, Addison-Wesley, Reading, MA (1989)

[20] P.Baldi, S.Brunak, Bioinfomatics – The Machine Learning Approach, The MIT Press, 2001

[21] M. Arbib (ed) The Handbook of Brain Theory and Neural Networks, 2003.

[22] ZISC Manual, Silicon Recognition, www.silirec.com, 2001.