Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS): On-line learning and Application for Time-Series Prediction

Qun Song and 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
Qsong@infoscience.otago.ac.nz, nkasabov@otago.ac.nz

Key words: dynamic evolving fuzzy inference systems; on-line adaptive learning; clustering; time series prediction

Abstract. This paper introduces a new type of fuzzy inference systems, denoted as DENFIS (dynamic evolving neural-fuzzy system), for adaptive on-line learning, and its application for dynamic time series prediction. DENFIS evolve through incremental, hybrid (supervised/unsupervised), learning and accommodate new input data, including new features, new classes, etc. through local element tuning. New fuzzy rules are created and updated during the operation of the system. At each time moment the output of DENFIS is calculated through a fuzzy inference system based on *m*-most activated fuzzy rules which are dynamically chosen from a fuzzy rule set. An approach is proposed for a dynamic creation of a first-order Takagi-Sugeno type fuzzy rule set for the DENFIS model. The fuzzy rules can be inserted into DENFIS before, or during its learning process, and the rules can also be extracted from DENFIS during, or after its learning process. An evolving clustering method (ECM), which is employed in the DENFIS model, is also introduced. It is demonstrated that DENFIS can effectively learn complex temporal sequences in an adaptive way and outperform some existing models.

1. Introduction

The complexity and dynamics of real-world problems, especially in engineering and manufacturing, require sophisticated methods and tools for building on-line, adaptive intelligent systems. Such systems should be able to grow as they operate, to update their knowledge and refine the model through interaction with the environment, that is the systems should have an ability of on-line learning [1, 14, 11]. On-line learning is concerned with learning data as the system operates (usually in a real time) and the data might exist only for a short time.

Here we propose a model called dynamic evolving neural fuzzy inference system (DENFIS). DENFIS is similar to EFuNN (evolving fuzzy neural systems [12]) in some principles. It inherits and develops EFuNN's dynamic features that makes DENFIS suitable for on-line adaptive systems. The DENFIS model uses a local generalisation. It was developed with the idea that, depending on the position of the input vector in the input space, a fuzzy inference system for calculating the output value is formed dynamically bases on *m* fuzzy rules created after the learning phase.

This paper is organised as follows: Section 2 gives a description of the evolving clustering method (ECM) which is used in the DENFIS model for partitioning the input space. A comparison between ECM and some other clustering methods, such as EFuNN [12], fuzzy C-means [2], K-means [13] and subtractive clustering method [4], is also presented in this section. Section 3 introduces the DENFIS model, and in section 4, DENFIS model is applied to Mackay-Glass time series [3, 5] prediction problem. Results are compared with the results obtained with the use of resource-allocation network (RAN) [14], evolving fuzzy-neural networks (EFuNNs), and evolving self-organising maps (ESOM) [6]. Conclusions and directions for further research are presented in the final section.

The analysis of results indicates clearly the advantages of DENFIS when used especially for on-line learning applications. In addition to this, the evolving clustering method, ECM, can perform well as an on-line, self-organised generic clustering model. Rules can be inserted or extracted from the model that makes it useful for many knowledge engineering applications.

2. Evolving Clustering Method: ECM

Here, an evolving, on-line, maximum distance-based clustering method, called *evolving clustering method* (ECM), is proposed to implement a scatter partitioning of the input space for the purpose of creating fuzzy inference rules. This method is specially designed for DENFIS.

2.1 ECM: A Distance-based On-line Evolving Clustering Method

Without any optimisation, ECM is a fast, one-pass algorithm for a dynamic estimation of the number of clusters in a set of data and for finding their current centres in the input data space. It is a distance-based clustering method. In this method, cluster centres are represented as the evolved rule nodes in an evolving connectionist system [12]. In any cluster, the maximum distance, *MaxDist*, between an example point and the corresponding cluster centre, is less than a threshold value, *Dthr*, that has been set as a clustering parameter and would affect the number of clusters to be estimated.

In the clustering process, the data examples come from a data stream and this process starts with an empty set of clusters. When a new cluster is created, the cluster centre, $\boldsymbol{Cc}$, is defined and its cluster radius, Ru, is initially set to zero. With more examples presented one after another, some of the already created clusters may be updated through changing their centres' positions and increasing their cluster radiuses. Which cluster will be updated for a current input vector, and how much it will be changed, depends on the position of the vector in the input space. A cluster will not be updated any more when its cluster radius, Ru, reaches the value that is equal to the threshold value, $Dthr$.

Figure 2 shows a brief ECM clustering process in a 2-D space. The ECM algorithm is described below:

- Step 0: Create the first cluster C_1 by simply taking the position of the first example from the input data stream as the first cluster centre $\boldsymbol{Cc}_1$, and setting a value 0 for its cluster radius Ru_1 (Figure1. a).
- Step 1: If all examples of the data stream have been processed, the algorithm is finished. Else, the current example, $\boldsymbol{x}_i$, is taken and the distances*, between this example and all the n already created cluster centres $\boldsymbol{Cc}_j$, $D_{ij} = \| \boldsymbol{x}_i - \boldsymbol{Cc}_j \|$, $j = 1, 2, \dots, n$, are calculated.
- Step 2: If there is a cluster center (centers) $\boldsymbol{Cc}_j$, for $j = 1, 2, \dots, n$, so that the distance value, $D_{ij} = \| \boldsymbol{x}_i - \boldsymbol{Cc}_j \|$ is equal to, or less than, the radius Ru_j, it is assumed that the current example $\boldsymbol{x}_i$ belongs to a cluster C_m with the minimum of these distances:

  $D_{im} = \| \boldsymbol{x}_i - \boldsymbol{Cc}_m \| = min (\| \boldsymbol{x}_i - \boldsymbol{Cc}_j \|)$, where:

  $D_{ij} \le Ru_j, j = 1, 2, \dots, n,$

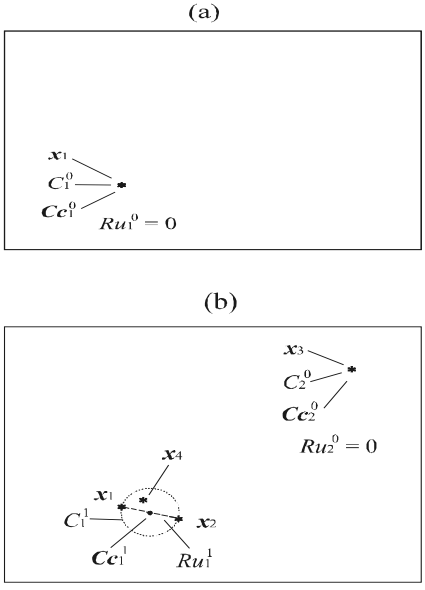
  In this case, neither a new cluster is created, nor any existing cluster is updated (the cases of $\boldsymbol{x}_4$ and $\boldsymbol{x}_6$ in Figure 1) and the algorithm returns to Step 1, else it goes to the next step.
- Step 3: Find a cluster C_a (with a centre $\boldsymbol{Cc}_a$ and a cluster radius Ru_a) from all n existing cluster centres through calculating the values $S_{ij} = D_{ij} + Ru_j$, $j = 1, 2, \dots, n$, and then select the cluster centre $\boldsymbol{Cc}_a$ with the minimum value S_{ia}:

  $S_{ia} = D_{ia} + Ru_a = min \{ S_{ij} \}, \ j = 1, 2, \dots, n.$
- Step 4: If S_{ia} is greater than $2\times Dthr$, the example $\boldsymbol{x}_i$ does not belong to any existing clusters. A new cluster is created in the same way as described in Step 0 (the cases of $\boldsymbol{x}_3$ and $\boldsymbol{x}_8$ in Figure 1), and the algorithm returns to Step 1.
- Step 5: If S_{ia} is not greater than $2\times Dthr$, the cluster C_a is updated by moving its centre, $\boldsymbol{Cc}_a$, and increasing the value of its radius, Ru_a. The updated radius Ru_a^{new} is set to be equal to $S_{ia} / 2$ and the new centre $\boldsymbol{Cc}_a^{new}$ is located on the line connecting the new input vector $\boldsymbol{x}_i$ and the cluster centre $\boldsymbol{Cc}_a$, so that the distance from the new centre $\boldsymbol{Cc}_a^{new}$ to the point $\boldsymbol{x}_i$ is equal to Ru_a^{new} (the cases of $\boldsymbol{x}_2$, $\boldsymbol{x}_5$, $\boldsymbol{x}_7$ *and* $\boldsymbol{x}_9$ in Figure 1). The algorithm returns to Step 1.

In this way, the maximum distance from any cluster centre to the farthest example that belongs to this cluster, is kept less than the threshold value, $Dthr$ though the algorithm does not keep any information of passed examples.

* In this paper, the distance, between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, is calculated as a *normalised Euclidean distance*, defined as follows:

$$\| \boldsymbol{x} - \boldsymbol{y} \| = \left(\sum_{i=1}^{q} | x_i - y_i |^2 \right)^{1/2} / q^{1/2}, \ \boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{R}^q. \qquad (1)$$



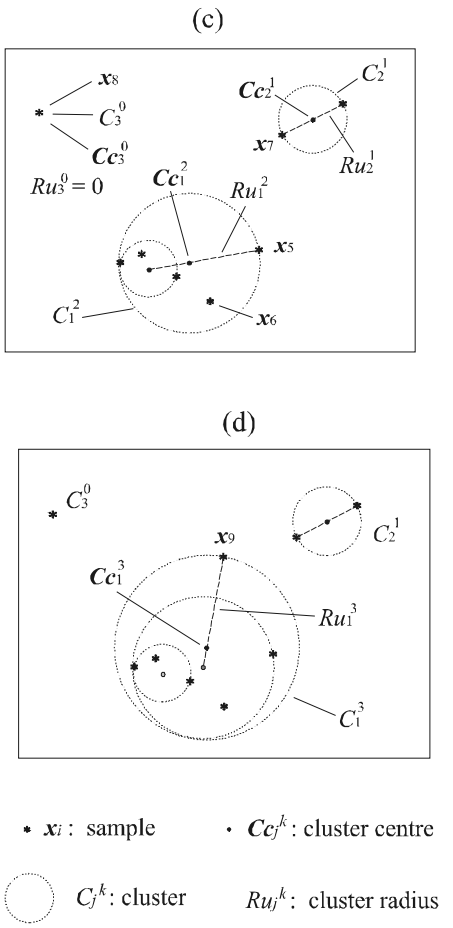


Figure 1. A brief clustering process using ECM with samples $\boldsymbol{x}_1$ to $\boldsymbol{x}_9$ in a 2-D space:

(a) The example $\boldsymbol{x}_1$ causes the ECM to create a new cluster C_1^0
(b) $\boldsymbol{x}_2$: update cluster $C_1^0 \rightarrow C_1^1$
$\boldsymbol{x}_3$: create a new cluster C_2^0
$\boldsymbol{x}_4$: do nothing
(c) $\boldsymbol{x}_5$: update cluster $C_1^1 \rightarrow C_1^2$
$\boldsymbol{x}_6$: do nothing
$\boldsymbol{x}_7$: update cluster $C_2^0 \rightarrow C_2^1$
$\boldsymbol{x}_8$: create a new cluster C_3^0
(d) x_9 : update cluster $C_1^2 \rightarrow C_1^3$

2.2. Applying ECM on a bench mark data set – on-line clustering of the gas-furnace data

The gas-furnace time series is a well-known bench-mark data set and has been frequently used by many researches in the area of neural networks and fuzzy system for control, prediction and adaptive learning [9]. It consists of 296 consecutive data pairs of methane at a time moment (t – 4), and the carbon dioxide CO_2 produced in a furnace at a time moment (t – 1) as input variables, with the produced CO_2 at the moment (t) as an output variable. In this case, the clustering simulation is implemented only on the input space. For comparing the performance of ECM and other traditional methods, the following five clustering methods are applied to the same data set :

(a) ECM, evolving clustering method (on-line, one-pass)
(b) EFuNN [12], evolving fuzzy-neural network clustering (on-line, one pass)
(c) SC [4], subtractive clustering (off-line, one pass)
(d) FCMC [2], fuzzy C-means clustering (off-line, multiple iterations)
(e) KMC [14], K-means clustering (off-line, multiple iterations)

Each of these methods partitions the data to a predefined number of clusters *NoC* = 15. The maximum distance, *MaxD*, between an example point and the corresponding cluster centre, and the value of the objection function *J* defined by Equation (2), are measured for comparison as shown in Table 1.

$$J = \sum_{j=1}^{15} \Big(\sum_{k,\ \mathbf{x}_k \in C_j} \| \boldsymbol{x}_k - \boldsymbol{Cc}_j \| \Big), \quad k = 1, 2, \ldots 296. \qquad (2)$$

We can see that the evolving clustering methods ECM obtains the minimum value of *MaxD*, that indicates that this method partitions the data set more uniformly than the other methods. Looking at the results from a different point of view, we can state that if all these clustering methods obtain the same value of *MaxD*, ECM wpould results in a less number of partitions. Considering that the ECM clustering is a 'one-pass' on-line process, the objection value *J* for the ECM simulation is acceptable as it is comparable with the *J* value for the other methods.

Table 1. Clustering results of the Gas-Furnace data set clustered into 15 clusters with the use of 5 different methods

Methods	*MaxD*	Objective value: *J*
ECM (on-line, one-pass)	0.1	12.9
EFuNN (on-line, onepass)	0.11	13.3
SC (off-line, one-pass)	0.15	11.5
FCM (off-line learning)	0.14	12.4
KM (off-line learning)	0.12	11.8

3. DENFIS: A Dynamic Evolving Neural-Fuzzy Inference System

3.1 General principles

The dynamic evolving neural-fuzzy inference system, DENFIS, uses Takagi-Sugeno type of fuzzy inference engine [15]. The inference engine in DENFIS is composed of m fuzzy rules indicated as follows:

if x_1 is R_{m1} and x_2 is R_{m2} and … and x_q is R_{mq},
then y is $f_m(x_1, x_2, \ldots, x_q)$

where "x_j is R_{ij}", $i = 1, 2, \ldots m$; $j = 1, 2, \ldots q$, are $m \times q$ fuzzy propositions as m antecedents for m fuzzy rules respectively; x_j, $j = 1, 2, \ldots, q$, are antecedent variables defined over universes of discourse X_j, $j = 1, 2, \ldots, q$, and R_{ij}, $i = 1, 2, \ldots m$; $j = 1, 2, \ldots, q$, are fuzzy sets defined by their fuzzy membership functions μ_{Aij}: $X_j \rightarrow [0, 1]$, $i = 1, 2, \ldots m$; $j = 1, 2, \ldots, q$. In the consequent parts, y is a consequent variable, and crisp linear functions f_i, $i = 1, 2, \ldots m$, are employed.

In the DENFIS model, all fuzzy membership functions are triangular type functions defined by three parameters as given by the following equation:

$$\mu(x) = mf(x, a, b, c) = \max(\min((x-a)/(b-a),(c-x)/c-b)),0). \quad (3)$$

where b is the value of the cluster centre on the x dimension, $a = b - d \times Dthr$, and $c = b + d \times Dthr$, $d = 1.2 \sim 2$. The threshold value, $Dthr$, is a clustering parameter.

For an input vector $\boldsymbol{x}^0 = [x_1^0\ x_2^0\ \ldots\ x_q^0]$, the result of inference, y^0 (the output of the system) is calculated as the weighted average of each rule's output:

$$y^0 = \frac{\sum_{i=1}^{m} w_i\ f_i(x_1^0, x_2^0, \ldots, x_q^0)}{\sum_{i=1}^{m} w_i}$$

$$\text{where, } w_i = \prod_{j=1}^{q} R_{ij}(x_j^0);\ i = 1, 2, \ldots m;\ \ j = 1, 2, \ldots, q.$$

3.2. Learning in DENFIS

In DENFIS, the first-order Takagi-Sugeno fuzzy rules are employed. The linear functions in the consequence parts are created and updated by linear least-square estimator (LSE) [10] on the learning data. We can express the function y as: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_1 + \ldots + \beta_q x_q$.
The following weighted least-square estimator formula

$$\boldsymbol{b} = (\mathbf{A}^T \mathbf{W}\, \mathbf{A})^{-1}\, \mathbf{A}^T \mathbf{W}\, \boldsymbol{y} \quad (4)$$

is used to obtain $\boldsymbol{b} = [b_0\ \ b_1\ \ b_2 \ldots b_q]^T$. The least-square estimator (LSE) of $\boldsymbol{\beta} = [\beta_0\ \ \beta_1\ \ \beta_2 \ldots \beta_q]^T$ is calculated with on a learning data set that is composed of p data pairs $\{([x_{i1}, x_{i2}, \ldots, x_{iq}], y_i), i = 1, 2, \ldots, p\}$. Here $\mathbf{W}$ is the weight matrix and its element, w_{jj}, are defined by $1–d_j$ (d_j is the distance between the j-th example and the corresponding cluster centre), $j = 1, 2, \ldots p$.

We can rewrite equations (4) with the use of a recursive LSE formula [8] as follows:

$$\begin{cases} \mathbf{P} = (\mathbf{A}^T\ \mathbf{A})^{-1}, \\ \boldsymbol{b} = \mathbf{P}\, \mathbf{A}^T \mathbf{W}\, \boldsymbol{y}, \end{cases} \quad (5)$$

In DENFIS, we use a weighted recursive LSE with a forgetting factor defined as follows. Let the k-th row vector of a matrix A is denoted as a_k^T and the k-th element of $\boldsymbol{y}$ is denoted as y_k. Then $\boldsymbol{b}$ can be calculated iteratively as follows:

$$\begin{cases} \boldsymbol{b}_{k+1} = \boldsymbol{b}_k + w_{k+1}\, \mathbf{P}_{k+1}\, \mathbf{a}_{k+1}\, (y_{k+1} - \mathbf{a}_{k+1}^T\, \boldsymbol{b}_k), \\ \mathbf{P}_{k+1} = \frac{1}{\lambda}\left[\mathbf{P}_k - \frac{w_{k+1}\, \mathbf{P}_k\, \mathbf{a}_{k+1}\, \mathbf{a}_{k+1}^T\, \mathbf{P}_k}{\lambda + \mathbf{a}_{k+1}^T\, \mathbf{P}_k\, \mathbf{a}_{k+1}}\right]; \end{cases} \quad (6)$$

where $k = n, n+1, \ldots p - 1$; w_{k+1} is the weight of $k+1$-th example defined by $1 - d_{k+1}$ (d_{k+1} is the distance between the $k+1$-th example and the corresponding cluster centre); and λ is a forgetting factor which typical value is between 0.8 and 1. The initial values of $\mathbf{P}_n$ and $\boldsymbol{b}_n$ are calculated using equation (5).

In DENFIS, the rules are created and updated within the input space partitioning obtained with the use of the evolving clustering method (ECM) and equations (3) and (6). If no rule insertion is applied as an initialisation procedure, the following steps are used for creating the first m fuzzy rules and calculating their function initial values of $\mathbf{P}$ and $\boldsymbol{b}$:

(1) Take the first n learning data pairs from the learning data set.
(2) Apply on-line clustering using ECM to obtain m cluster centres.
(3) For every cluster centre $\boldsymbol{C}_i$, find m data points p_i, $i = 1, 2, \ldots, m$, which positions in the input space are the closest to the centre
(4) To obtain a fuzzy rule that corresponds to a cluster, create the antecedent of the fuzzy rule with the position of the cluster centre and equation (3). Use equation (5) to obtain the values of $\mathbf{P}$ and $\boldsymbol{b}$ from the p_i data pairs and create the function y as the consequence of the fuzzy rule. The distances between

p_i data points and the cluster centre are taken as the weights in equation (5).

In the above steps, m, n and p are parameters of DENFIS on-line learning process, and the value of p should be greater than the number of input elements q, i.e. $p > q$.

With new data pairs entered into the system, new fuzzy rules may be created and some existing rules may be updated. A new fuzzy rule is created when a new cluster centre is created by the ECM algorithm. The antecedent of the new fuzzy rule is formed with the use of equation (3). Alternatively, for a new data pair, existing fuzzy rules are updated by using equation (6) if the distances from the rule nodes that represent the rule antecedents and the input vector in the input space are not greater than $2 \times Dthr$ (the threshold value, which is a clustering parameter). The distances between these rule nodes and the data point in the input space are taken as the weights in equation (6). Antecedents of fuzzy rules may be changed by the ECM algorithm. A fuzzy rule then will have a new antecedent calculated with the use of equation (3).

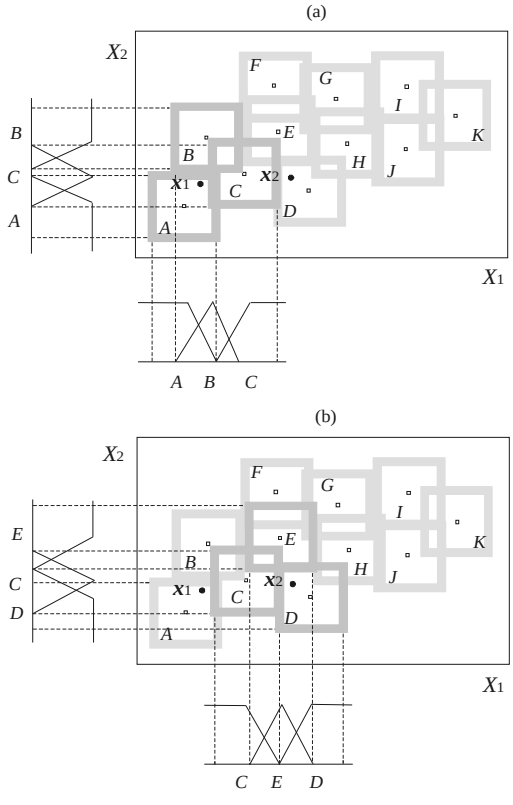


Figure 2. Two fuzzy rule groups depending on two input vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in the 2D space

3.3 Dynamic Takagi-Sugeno Fuzzy Inference in DENFIS

The DENFIS model uses a dynamic Takagi-Sugeno fuzzy inference system. In addition to dynamically creating and updating fuzzy rules during the learning process, the fuzzy rules that participate in the inference for each new input vector are dynamically chosen from the existing fuzzy rule set depending on the position of the current input vector in the input space. Figure 2 illustrates the cases of input vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in a 2-D space. For $\boldsymbol{x}_1$, fuzzy rules A, B and C are chosen to form an inference system, while for input vector $\boldsymbol{x}_2$, fuzzy rules C, D and E are chosen.

4 Time Series Modelling and Prediction with the DENFIS Model

In this section the DENFIS model is applied to model and predict future values of a chaotic time series - the Mackey-Glass (MG) data set [3, 5]. This set has been used as a bench-mark in the areas of neural networks, fuzzy systems and hybrid systems. The time series is created with the use of the MG time-delay differential equation defined below:

$$\frac{d\,x(t)}{d\,t} = \frac{0.2\,x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \qquad (7)$$

To obtain this time series values at integer points, the fourth-order Runge-Kutta method was used to find the numerical solution to the above MG equation. Here we assume that the time step is 0.1, $x(0) = 1.2$, $\tau = 17$ and $x(t) = 0$ for $t < 0$. The task is to predict the value $x(t + 85)$ from the input vector $[x(t - 18)\ \ x(t - 12)\ \ x(t - 6)\ \ x(t)]$. For the purpose of comparative analysis, we also applied some existing on-line learning models on the same task. These models are Neural gas [7], resource-allocating network (RAN) [14], evolving self-organising maps (ESOM) [6] and evolving fuzzy-neural network (EFuNN) [12]. Here, we take the non-dimensional error index (NDEI) [5], which is defined as the root mean square error (RMSE) divided by the standard deviation of the target series. The following experiments were conducted: 3000 data points, from t = 201 to 3200, are used as learning data, and 500 data points, from t = 5001 to 5500, are used as testing data. For each of the mentioned above on-line models, the learning data is used in the learning phase, and then testing data is used in the recall processes. Table 2 lists the prediction results (NDEI for the recall processes on testing data after on-line learning) and the number of rules or units created (evolved) in each model.

Table 2. Prediction results for several on-line learning models on Mackey Glass test data after on-line learning

Methods	Fuzzy rules (DENFIS) Rule nodes (EFuNN) Units (others)	NDEI for testing data
Neural gas [7]	1000	0.062
RAN [14]	113	0.373
RAN [14]	24	0.17
ESOM [6]	114	0.32
ESOM [6]	1000	0.044
EfuNN[12]	193	0.401
EfuNN[12]	1125	0.094
DENFIS	58	0.276
DENFIS	883	0.042
DENFIS with rule insertion	883	0.033

Depending on the parameter values different DENFIS models can be obtained. A trade off between the number of rules (rule nodes, data clusters) needs to be found for any particular application. The same is valid for the EFuNN and for the other models.

The property of DENFIS for rule insertion and rule extraction is used in the last experiment as shown in table 2. We first obtained a group of fuzzy rules using an off-line method, which is similar to the way initial fuzzy rules are obtained in section 3.1, with the use of the first half of learning data (1500 examples). Then we inserted these rules into a DENFIS model and trained it continuously in an on-line mode on next half of the learning data (1500 examples). Then, we used the test data set in a recall mode. The rule insertion procedure when used as an initialisation improves the generalisation (e.g. reduces the test error).

5. Conclusions and directions for further research

This paper presents the principles of a new fuzzy inference system, DENFIS for on-line knowledge-based, adaptive learning systems. The DENFIS model is based on Takagi-Sugeno fuzzy rules and fuzzy inference. It uses *m* highly activated fuzzy rules to dynamically compose an inference system for calculating the output values. The proposed method and system demonstrate their superiority when compared with other on-line learning models, such as Neural gas, RAN, EFuNN and ESOM. DENFIS utilises a local generalisation method, similar to EFuNN and CMAC neural network [1]. In order to obtain good generalisation results it requires more training data than the models which are based on global generalisation. During its learning, the DENFIS model forms areas in the input space with partitioned regions. If there are not sufficient data examples received, these areas may not cover the whole input space. In the recall process DENFIS would produce a satisfactory result if the new examples (test data) fall inside these areas.

Further directions for research include: improvement of the DENFIS on-line learning method; applying DENFIS to real problems of adaptive process control, mobile robot navigation, and adaptive classification

Acknowledgements

This research is part of a research programme funded by the New Zealand Foundation for Research Science and Technology, contract UOOX0016.

Reference

1. Albus, J.S., A new approach to manipulator control: The cerebellar model articulation controller (CMAC), Tarns. of the ASME: Journal of Dynamic Systems, Measurement, and Control, pp.220:227, Sept. (1975)
2. Bezdek, J.C., “Pattern Recognition with Fuzzy Objective Function Algorithms”, Plenum Press, New York, 1981.
3. Mackey, M. C., Glass, L., “Oscillation and Chaos in Physiological Control systems”, Science, 197:287-289, 1977.
4. Chiu, S., “Fuzzy Model Identification Based on Cluster Estimation”, Journal of Intelligent & Fuzzy System, Vol. 2, No. 3, Step. 1994.
5. Croder, III R. S., “Predicting the Mackey-Glass Timeseries with Cascade- correlation Learning”, in: D. Touretzky, G. Hinton and T. Sejnowski eds., Proc. of the 1990 Connectionist Models Summer School, 117-123, Carnegic Mellon University, 1990.
6. Deng, D., Kasabov, N., “ESOM: An Algorithm to Evolve Self-Organising Maps from On-line Data Streams, Proc. IJCNN’2000, Como, Italy, July 2000, vol. VI, 3-8, IEEE Computer Society Press , Los Alamitos
7. Fritzke, B. “A growing neural gas network learns topologies”, Advances in Neural Information Processing Systems, vol.7 (1995).

8. Goodwin, G. C., Sin, K. S., “Adaptive Filtering Prediction and Control”, Prentice-Hall, Englewood Cliffs, N. J., 1984.
9. G. E. P, Box., and G. M, Jenkins., “Time series analysis, forecasting and control”, San Francisco, CA: Holden Day, 1970.
10. Hsia, T. C., “System Identification: Least-Squares Methods”, D.C. Heath and Company, 1977.
11. Kasabov, N., “ECOS: A framework for evolving connectionist systems and the eco learning paradigm”, Proc. of ICONIP'98, Kitakyushu, Oct. 1998, IOS Press, 1222-1235.
12. Kasabov, N., “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, 1998, 271-274
13. MacQueen, J., “Some Methods for Classification and Analysis of Multi-variate Observations”, in: Proceedings of the Fifth Berkeley Symposium on Mathematics, Statistics, and Probability, eds. L.M. LeCam and J. Neyman Berkeley: U. California Press, 281, 1967.
14. Platt, J., “A Resource Allocating Network for Function Interpolation”, Neural Comp., 3, 213-225, 1991.
15. Takagi, T. and Sugeno, M., Fuzzy identification of systems and its applications to modeling and control. IEEE Trans. on Systems, Man, and Cybernetics, 116-132, 1985.