ANN Rule Extraction using Evolutionary Programmed Fuzzy Membership Functions

Michael J. Watts

National Centre for Advanced Bio-Protection Technologies
PO Box 84
Lincoln University
Canterbury
New Zealand

wattsm2@lincoln.ac.nz

Abstract

An algorithm is presented that uses evolutionary programming to construct fuzzy membership functions that are used to extract Zadeh-Mamdani fuzzy rules from a constructive neural network. The algorithm has potential applications in fields such as data mining and knowledge-based decision support systems. Evaluation of the algorithm over two well known benchmark data sets shows that while the results are promising, some problems are apparent. These problems provide avenues for further research.

Keywords: Evolving Connectionist Systems, ECoS, Simple Evolving Connectionist System, SECoS, fuzzy rule extraction, evolutionary programming.

I. Introduction

Evolving Connectionist Systems (ECoS) [3] are a class of constructive neural network algorithms that are capable of fast one-pass learning and are resistant to catastrophic forgetting. They are thus well-suited to applications where new data is becoming continuously available, such as speech-recognition systems [2].

The original ECoS network was the Evolving Fuzzy Neural Network (EFuNN) [4] but other versions, including the minimalist Simple Evolving Connectionist System (SECoS) [7] have also been developed.

The motivation for SECoS was the desire for an ANN that would learn in the same way as EFuNN, but was without the limitations of the in-built fuzzy membership functions. Thus, SECoS is a constructive ANN that learns in a manner identical to EFuNN (see Subsection IIA for details of ECoS learning) but is not limited by the fixed MF that constrains EFuNN. SECoS networks also tend to be much smaller than EFuNN, in terms of the number of neurons that are added during training.

While the major advantage of EFuNN has been claimed to be the ability to extract fuzzy rules [5], an algorithm has also been developed to extract fuzzy rules from SECoS [6]. This algorithm uses membership functions (MF) that are external to the network, and are thus able to be arbitrarily modified. However, the design and optimisation of these MF can be a difficult task.
The work reported in this paper investigated the use of evolutionary programming (EP) [1] in designing these MF.

II. Simple Evolving Connectionist Systems

The SECoS ANN [7] is a minimalist implementation of the ECoS principles and consists of three layers of neurons. The first is the input layer. The second is the evolving or constructive layer: it is this layer to which neurons are added and in which learning takes place. The activation of the evolving layer neurons is based on the distance between the input to evolving layer weight vectors and the current input vector. The third layer is the output layer.

The activation *A* of an evolving layer neuron *n* is determined by Equation 1.

$$A_n = 1 - D_n \qquad (1)$$

where:
A_n is the activation of the neuron *n*, and
D_n is the distance between the input vector and the incoming weight vector for that neuron.

Since SECoS networks are fully connected, the number of connections coming into an evolving layer neuron from the input layer is the same as the number of input neurons. Thus, the incoming weight vector has the same dimensionality as the vector input to the evolving layer. It is therefore possible to directly measure the distance in Euclidean space between the two vectors. Although the distance can be measured in any way that is appropriate for the inputs, this distance function must return a value in the range of $[0,1]$. For this reason, the SECoS algorithm assumes that the input data will be normalised, as it is far easier to formulate a distance function that produces output in the desired range if it is normalised to the range $[0,1]$.

Thus, examples which exactly match the exemplar stored within the neurons incoming weights will result in an activation of 1, while examples that are entirely outside of the exemplars region of input space will result in an activation of near 0.

Whereas most ANN propagate the activation of each neuron from one layer to the next, in SECoS only the activation of the winning (most highly activated) evolving layer neuron is propagated to the following neuron layers.

A. SECoS Training

The SECoS Learning algorithm is based on accommodating within the evolving layer new training examples, by either modifying the weight values of the connections attached to the evolving layer neurons, or by adding a new neuron to that layer. The algorithm employed is described below:

- Propagate the input vector *I* through the network.
- Find the most highly activated (winning) neuron *j* and its activation A_J .
- IF A_J is less than the sensitivity threshold S_{thr} ..
    - Add a neuron.
- ELSE
    - Evaluate the errors between the calculated output vector O_c and the desired output vector O_d .
    - IF the absolute error over the desired output is greater than the error threshold E_{thr}
        - Add a neuron.
    - ELSE
        - Update the connections to the winning evolving layer neuron.
- Repeat for each training vector.

When a neuron is added, its incoming connection weight vector is set to the input vector *I*, and its outgoing weight vector is set to the desired output vector O_d .

The weights of the connections from each input *i* to the winning neuron *j* are modified according to Equation 2.

$$W_{i,j}(t+1) = W_{i,j}(t) + \eta_1 I_i - W_{i,j}(t) \quad \textbf{(2)}$$

where:
$W_{i,j}(t)$ is the connection weight from input *i* to *j* at time *t*
η_1 is the learning rate parameter for the input to evolving layer connections
I_i is the *i*th component of the input vector *I*

The weights from neuron *j* to output *o* are modified according to Equation 3.

$$W_{j,o}(t+1) = W_{j,o}(t) + \eta_2 A_j E_o \quad \textbf{(3)}$$

where:
$W_{j,o}(t)$ is the connection weight from *j* to output *o* at time *t*
η_2 is the learning rate parameter for the evolving to output layer connections
A_j is the activation of *j*
E_o is the signed error at *o*, as measured according to Equation 4.

$$E_o = O_d - A_o \quad \textbf{(4)}$$

where:
O_d is the desired activation value of *o* and
A_o is the actual activation of *o*.

II. Fuzzy Rule Extraction from SECoS

The algorithm for extracting Zadeh-Mamdani fuzzy rules from trained SECoS networks is as follows:

- for each evolving layer neuron *h*
    - create a new rule
    - for each input neuron *i*

- find the MF associated with *i* that activates the most strongly for the weight $W_{i,h}$
- add that MF to the antecedent of the rule for that input
  - this is the MF for that input for this rule
  - insert the membership degree of the weight in the winning MF as the degree of importance for this condition

o for each output neuron *o*

- find the MF associated with *o* that activates the most strongly for the weight $W_{h,o}$
- add that MF to the consequent of the rule for that output
  - this is the MF for that output for this rule
  - insert the membership degree of the weight in the winning MF as the degree of confidence for this consequent

The most significant problem with the extraction of fuzzy rules from SECoS can best be shown graphically. Figure 1 plots the positions in input space of the evolving layer neurons of a trained SECoS network as well as the Voronoi regions defined by each neuron. Overlaid on this plot is a grid that shows the partitioning of the input space by the five MF attached to each input variable. It can be seen that the partitions affected by the MF do not correspond well to either the positions of the neurons or the Voronoi regions. Yet, when fuzzy rules are extracted from this network using these MF, these are the regions that will be defined by these rules: any examples that fall within these regions will cause the rule to activate. This will cause a great deal of confusion within the rule set, and can potentially lead to contradictory or inconsistent rules. That is, the rules may not be able to generalise properly nor will the rules be able to accurately replicate the performance of the network. Thus, the naive application of unoptimised MF to the task of extracting rules from SECoS networks is not guaranteed to lead to accurate rules. The optimisation of these MF is the motivation for the work in this paper.

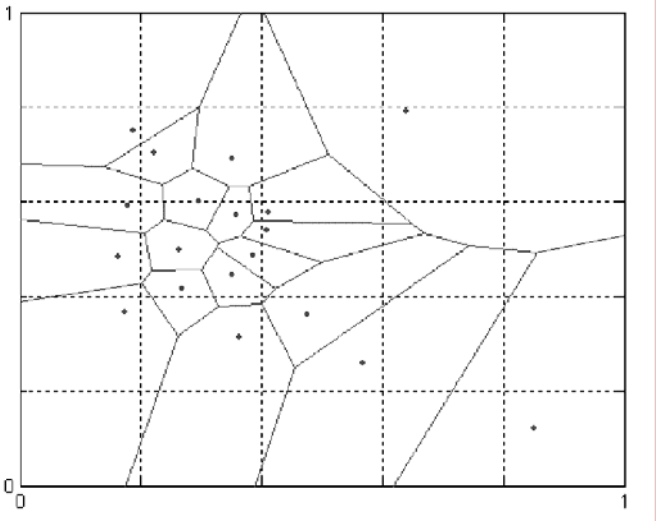

Figure 1. Regions defined by neurons compared to regions defined by fuzzy MF

IV. Optimizing Membership Functions with Evolutionary Programming

EP was selected for this work because it obviates the need to formulate a representation schema of the fuzzy MF. A genetic algorithm (GA) schema that allows for the addition, deletion and

modification of the sets of MF being evolved would have been difficult to implement. This is especially true if MF of different types were to be used, as different types of MF have different numbers of parameters. The fact that EP operates directly on the MF allowed the entire encoding issue to be ignored.

The algorithm to select MF for rule extraction is as follows:

1. Randomly generate *p* individuals as the initial population, where *p* is the size of the population and each individual is a set of MF.
2. Use the MF in each individual to extract fuzzy rules from the provided SECoS network.
3. Evaluate the accuracy of the rules over the provided evaluation data set.
4. Create the child generation *c* by mutating the best *n* performing parents.
5. Repeat steps 2 to 4 until the specified number of generations have passed.

The mutation was carried out by randomly selecting one of the following operations for each MF in each parent:

- No operation
- Delete MF
- Insert new MF
- Mutate MF

If "No operation" was selected, then the current MF was simply copied unchanged into the child. Deletion or insertion means that either the current MF was not included in the child or a new, randomly selected MF was inserted into the child at the current position. MF were mutated by having normally distributed changes added to each of their parameters. The probabilities of each of these operations occurring were encoded within the individual and were also mutated each time a child was created from that parent. At the end of the EP run, the best performing rule set was saved.

V. Experimental Method

The purpose of these experiments was to evaluate the performance of the evolved MF.
This was done via the following procedure. Firstly, SECoS networks were created and trained.
Secondly, fuzzy rules were extracted from the trained networks, using the rule extraction procedure described in Section 3 in combination with manually-created MF. Finally, rules were extracted from the networks using EP designed MF.

Ten-fold cross validation was used at each step of these experiments. The benchmark data sets chosen were randomly split into ten equally sized subsets. Eight of the subsets were combined to form the first of the training sets, or data set A. The remaining two subsets were held out as sets B and C. The subsets held out as sets B and C were rotated so that each of the ten subsets was used as set B and C at least once.

Two benchmark data sets were selected, Fisher's iris data and the wine classification data set.

For the first step of the experiments, a SECoS network was trained on set A and its performance evaluated over sets A, B and C. The network was then further trained over set B, and its performance again evaluated over all three sets. This was done to test the memorisation and adaptation ability of the SECoS.

The second step of the experiments was extraction of fuzzy rules using manually created MF. The number of MF used for each input variable was varied from two to ten. Fuzzy rules were

extracted from each trained SECoS, including those that resulted from the further training over set B. Triangular MF were used.

The final step was evaluation of the rules created from EP designed MF. Since EP is a stochastic method, as opposed to the deterministic rule extraction method used previously, one hundred trials of the EP was carried out over each fold of the data. That is, there were a total of one thousand EP trials for each of the benchmark data sets.

The MF designed by EP were constrained to triangular MF so that a fair comparison could be made to the rules extracted using manually created MF. Each EP was run for one hundred generations with a population size of one hundred individuals. At each generation, the top fifty individuals were used as parents for the next generation. The evaluations within the EP were performed using the data set that the SECoS had most recently been trained with. That is, after training the SECoS over data set A, set A was used as the evaluation data set and after training the SECoS over data set B, set B was used as the evaluation data set.

At the completion of the trials at each step, the statistical parameters of the performance metrics were calculated. The total percentage correctly classified was used as the performance metric for both data sets. The statistical parameters calculated were the mean and standard deviation for the performance of the original SECoS and the rules extracted using manually created MF, while the mean and approximate variance were used for the rules extracted using EP designed MF.

VI. Results

The results over the iris data set are presented in Table 1. The results for the SECoS networks are in the rows labelled SECoS. The results for the fuzzy rules extracted using manually created MF are in the rows labelled Rules: only the best results are presented here, in this case the results from eight MF per input variable. The accuracies of the rules extracted using EP designed MF are in the rows labelled EP-Rules.

After further training over set B, the performance of the SECoS improved significantly over set B without significant change over sets B or C. That is, the network was able to learn more data without forgetting the previous training data.

The rules using manually-designed MF were compared with the accuracies of the original networks using a two-tailed *t*-test. Testing for equality with p=0.01 showed that while the performance of the extracted rules was significantly less accurate over set A, there was no significant difference in performance over sets B and C either before or after further training of the SECoS.

The performance of the rules extracted using EP-designed MF was compared to both the original networks and the manually extracted rules via a two-tailed pooled-variance *t*-test. Testing for equality with p=0.01 showed that while the performance was significantly less than the original networks it was significantly better than the manually extracted rules over sets A and C after the initial training over set A. Also, the performance was significantly better over set B after further training over set B. This appears to be because the EP optimised the MF for performance over that particular data set, without regard for generalisation performance. This would certainly explain the significantly poorer results over sets A and C when extracting rules from the SECoS after additional training over set B.

Recall Set	A	B	C	Neurons / Rules
		Trained on Set A		
SECoS	97.8/1.6	94.7/6.9	93.3/7.0	24.8/2.1
Rules	95.0/1.8	95.3/4.5	92.7/8.6	24.8/2.1
EP-Rules	96.2/0.2	94.2/0.6	94.5/0.6	24.8/2.1
		Trained on Set B		
SECoS	97.5/1.3	100.0/0.0	92.0/7.6	26.1/2.4
Rules	92.0/2.9	98.0/3.2	93.3/6.3	26.1/2.4
EP-Rules	89.2/0.7	99.9/0.3	88.3/0.9	26.1/2.4

Table 1. Mean, standard deviation and approximate variance of accuracies over the iris classification data

The mean and approximate variance of the number of MF selected by the EP for the input and output variables is presented in Table 2. The number of MF per input is much smaller than the number used to manually extract rules from the SECoS. Thus, while the EP generated rules that were highly optimised to the current data set, they did so with a much smaller number of input MF than was otherwise used. However, the manually designed MF used only two MF per output, while the mean number of MF selected by the EP was much larger.

Train Set	Input	Output
A	3.0/0.4	3.8/0.4
B	2.9/0.4	3.9/0.4

Table 2. Mean and approximate variance of number of membership functions per input and output variable for the iris classification problem

The results of the wine classification dataset are presented in Table 3. The labels of the row are as for Table 1 above. The results for manually created rules are the results for seven MF per input variable. Statistical tests for equality were carried out as above.

As with the iris data above, the SECoS was able to learn set B without forgetting set A or significantly affecting the performance over set C.

There were no significant differences between the accuracies of the original networks and the accuracies of the rules extracted using the manually created MF. The rules extracted from the SECoS after training on set A and using evolved MF were significantly less accurate than the original networks, but were significantly more accurate over set C than the rules extracted using manually designed MF. After further training on set B, the rules extracted using evolved MF were significantly less accurate over sets A and C, but were significantly more accurate over set B.

Recall Set	A	B	C	Neurons / Rules
		Trained on Set A		
SECoS	99.7/0.5	89.4/13.5	89.9/7.3	74.4/9.5
Rules	99.7/0.5	87.2/8.6	81.4/12.4	74.4/9.5
EP-Rules	98.3/0.3	83.1/1.0	83.6/1.1	74.4/9.5
		Trained on Set B		
SECoS	99.9/0.4	100.0/0.0	92.7/10.2	82.5/5.0
Rules	99.9/0.4	98.9/3.5	86.5/10.2	82.5/5.0
EP-Rules	30.9/0.6	100.0/0.0	74.7/0.5	82.5/5.0

Table 3. Mean, standard deviation and approximate variance of accuracies over the wine classification data

The mean and approximate variance of the number of MF selected for each input variable by the EP are presented in Table 4. As before, the EP had selected fewer MF per input variable than those that were manually created. Also as with the iris results, the number of MF selected per output variable was larger than the two per output that were manually selected.

Train Set	Input	Output
A	3.0/0.4	4.1/0.4
B	1.2/0.4	1.4/0.4

Table 4. Mean and approximate variance of number of membership functions per input and output variable for the wine classification problem

VII. Discussion

There are two conclusions that can be drawn from the results over the iris and wine classification data sets.

Firstly, while the evolved MF produce rules that are superior over the data sets to which the EP has optimised the MF, they tend to be inferior over other data sets. That is, the EP over-fits the MF to the data that is used in the evaluation function, compared to rules created using manually selected MF.

Secondly, the EP consistently selects fewer MF for the input variables than were manually selected and more MF for the output variables. This demonstrates that the parameters of the MF are more important than the number of MF.

VIII. Future Work

One of the motivations for using EP to optimise the MF was that it makes it easier to mix MF of different types, that is, to use MF of heterogeneous type. The work reported in this paper, however, was constrained to triangular MF. Investigating the use of mixed MF is therefore a priority for future work.

Also of interest is seeding the initial population with existing sets of MF. That is, introducing into the initial EP population MF sets that are known to perform well, such as the manually created MF that were used for comparison in this work. The EP could therefore improve the existing MF as well as discover new, advantageous MF.

A major problem with the work reported in this paper is that the EP algorithm over-fits the MF to the current data set. This is a symptom of a deeper problem, whereby the fuzzy rules are intended to represent the network, but must be evaluated over a data set. While the evaluation could be done by comparing the behaviour of the rules over the test data set to the behaviour of the network over the test data set, this is constraining because it is desirable for the rules to exceed the performance of the SECoS where possible. One possible way around this problem is to extract the test data directly from the SECoS. This is possible because the neurons that are inserted into the SECoS during training are direct copies of the training examples, which are then modified by further

training. Thus, the incoming and outgoing connections weights of the evolving layer neurons can be extracted and used as data that represent more closely what the SECoS has learned.

IX. Conclusion

A method was presented for constructing fuzzy membership functions using evolutionary programming and for using these membership functions to extract fuzzy rules from a constructive neural network. Experiments over two benchmark data sets have shown that while the extracted rules are able to perform competitively with those that are extracted using manually created MF they tend to over-fit the data. However, the evolutionary discovered rules make use of a much smaller number of membership functions, which may enhance readability of the rules.Future work will focus on methods of improving the generalisation accuracy of the evolved rules.

References

[1] Fogel, L.J., Owens, A.J. and Walsh, M.J. Artificial Intelligence Through a Simulation of Evolution. In Maxfield, M, Callahan, A. and Fogel, L., editors, Biophysics and Cybernetic Systems: Proceedings of the 2nd Cybernetics Sciences Symposium (1965) 131-155.

[2] Ghobakhlou, A. and Watts, M. and Kasabov, N. Adaptive Speech Recognition with Evolving Connectionist Systems. Information Sciences, 156 (2003) 71-83.

[3] Kasabov, N. The ECOS Framework and the ECO Learning Method for Evolving Connectionist Systems. Journal of Advanced Computational Intelligence, 2(6) (1998) 195-202.

[4] Kasabov, N., Evolving Fuzzy Neural Networks - Algorithms, Applications and Biological Motivation, in Methodologies for the Conception, Design and Application of Soft Computing, Takeshi Yamakawa and Gen Matsumoto, editors, World Scientific (1998) 271-274.

[5] Kasabov, N. and Woodford, B. Rule Insertion and Rule Extraction from Evolving Fuzzy Neural Networks: Algorithms and Applications for Building Adaptive, Intelligent Expert Systems, in IEEE International Fuzzy Systems Conference (1999) 1406-1411.

[6] Watts, M.J. Fuzzy Rule Extraction from Simple Evolving Connectionist Systems. International Journal of Computational Intelligence and Applications. 4(3) (2004) 1-10.

[7] Watts, M.J. and Kasabov, N. Simple Evolving Connectionist Systems and Experiments on Isolated Phoneme Recognition, in Proceedings of the First IEEE Conference on Combinations of Evolutionary Computation and Neural Networks, San Antonio, May 2000 (IEEE Press, 2000) 232-239.

Dr Michael J. Watts is a postdoctoral fellow in the National Centre for Advanced Bio-Protection Technologies at Lincoln University, New Zealand. He earned the degree of Bachelor of Science with First Class Honours in Information Science from the University of Otago in 1996. Dr Watts started work as a teaching fellow in the Department of Information Science, University of Otago, in February 2000, and was promoted to the grade of senior teaching fellow in 2002. He gained his PhD degree in 2004, on the topic of the characterisation, simplification, formalisation, explanation and optimisation of evolving connectionist systems. He commenced his postdoctoral work in October 2004. His research interests include ecological modelling, bioinformatics, artificial neural networks, evolutionary computation and knowledge discovery.