GA-PARAMETER OPTMISATION OF EVOLVING CONNECTIONIST SYSTEMS FOR CLASSIFICATION AND A CASE STUDY FROM BIOINFORMATICS

Nikola Kasabov and Qun Song

Knowledge Engineering and Discovery Research Institute
Auckland University of Technology
Private Bag 92006, Auckland 1020, New Zealand
Emails: nik.kasabov@aut.ac.nz; qsong@aut.ac.nz

ABSTRACT

The paper describes an algorithm for parameter optimisation of evolving connectionist systems (ECOS) in an off-line processing mode. The algorithm is illustrated on a case study of a classification system that uses gene expression data to predict an outcome of a treatment of cancer disease.

1. INTRODUCTION – PARAMETER OPTMISATION IN EVOLVING PROCESSES

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.

For example, many functions associated with a living cell, are difficult to model in a model that has a fixed set of parameter values, e.g. a MLP network with a fixed learning rate or a fixed number of hidden neurons. This is especially true for evolving connectionist systems (ECOS) [1-4] that evolve their structure over time. ECOS need to have evolving, adapting parameter values.

This paper introduces a GA algorithm for optimisation of the parameters of an ECOS for classification and illustrates it on gene expression classification model.

2. PRINCIPLES OF EVOLVING CONNECTIONIST SYSTEMS

Evolving connectionist systems are multi-modular, connectionist architectures that facilitate modelling of evolving processes and knowledge discovery [1-4]. An evolving connectionist system may consist of many evolving connectionist modules.

An evolving connectionist system 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.

The evolving connectionist systems presented in [1-4] have the following specific characteristics:

1) They evolve in an open space, not necessarily of fixed dimensions.
2) They learn in on-line, pattern mode, incremental learning, fast learning - possibly by one pass of data propagation.
3) They learn in a life-long learning mode.
4) They learn as both individual systems, and evolutionary population 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 kinds of knowledge, mostly combined memory based, statistical and symbolic knowledge.

The evolving connectionist models presented in [1-4] are knowledge-based models, facilitating Zadeh-Mamdani fuzzy rules (EFuNN, HyFIS), Takagi-Sugeno fuzzy rules

(DENFIS), on-line fuzzy clustering (ECM). An example of Zadeh-Mamdani type of rule is given below:

IF x1 is High (0.7) and x2 is Low (0.8) THEN y is Medium (0.9), number of examples accommodated in the rule is 45; radius of the cluster covered by the rule is 0.5.

One version of the EfuNN model for classification classifies a data set into a number of classes and finds their class centres in the N-dimensional input space by "placing" a rule node. Each rule node is associated with class and with an influence (receptive) field representing a part of the N-dimensional space around the rule node. Usually, such an influence field in the N-dimensional space is a hyper-sphere.

There are two distinct phases of EfuNN operation. During the first, learning phase, data vectors are fed into the system one by one with their known classes. The learning sequence of each iteration is described as the following steps:

1) If all vectors have been inputted, finish the current iteration; otherwise, input a vector from the data set and calculate the distances between the vector and all rule nodes already created;

2) If all distances are greater than a max-radius parameter, a new rule node is create. The position of the new rule node is the same as the current vector in the input data space and its radius is set to the min-radius parameter, and then go to step 1; otherwise:

3) If there is a rule node with a distance to the current input vector less then or equal to its radius and its class is the same as the class of the new vector, nothing will be changed and go to step 1; otherwise:

4) If there is a rule node with a distance to the input vector less then or equal to its radius and its class is different from those of the input vector, its influence field should be reduced. The radius of the new field is set to the larger value from the distance minus the min-radius, and the min-radius.

5) If there is a rule node with a distance to the input vector less then or equal to the max-radius, and its class is the same to the vector's, enlarge the influence field by taking the distance as the new radius if only such enlarged field does not cover any other rule node which has the different class; otherwise, create a new rule node the same way as in step 2, and go to step 1.

The recall (classification phase of new input vectors) is performed in the following way:

1) If the new input vector lies within the field of one or more rule nodes associated with one class, the vector belongs to this class;

2) If the input vector lies within the fields of two or more rule nodes associated with different classes, the vector will belong to the class corresponding the closest rule node.

3) If the input vector does not lie within any field, then there are two cases: (1) one-of-n mode: the vector will belong to the class corresponding the closest rule node; (2) m-of-n mode: take m highest activated by the new vector rule nodes, and calculate the average distances from the vector to the nodes with the same class; the vector will belong to the class corresponding the smallest average distance.

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

1) Max-radius
2) Min-radius
3) Number of membership functions (mf)
4) m-of-n value

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

3. A METHOD FOR ECOS PARAMETER OPTMISATION BASED ON GENETIC ALGORITHM

A GA-based algorithm for on-line and off-line parameter optimization of ECOS is presented in [3]. Here the off-line algorithm is further elaborated and implemented for a particular class of EfuNNs for classification. The algorithm is presented in fig.1

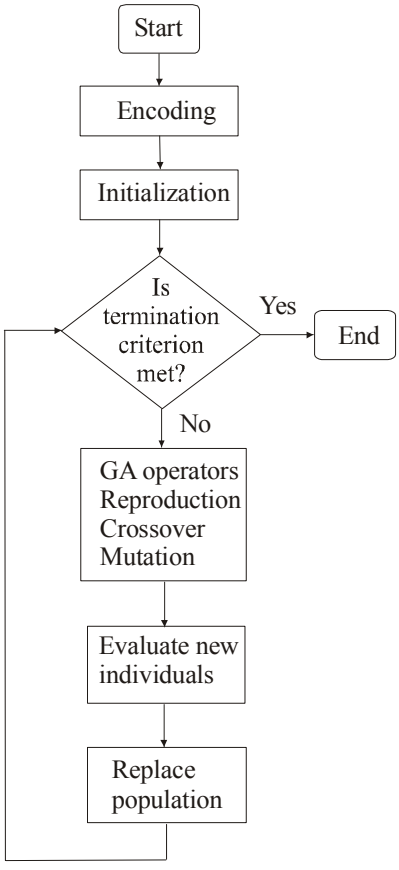

Fig. 1. A flow diagram of GA-optimisation algorithm for the parameter optimization of off-line EfuNN for classification

Many EfuNN modules (individuals) are evolved at the same time on randomly selected training data from the data set and tested on a selected test data. As a fitness criterion, the test RMSE error is used. The GA algorithm runs over generations of populations and standard operations are applied (see[3]) such as: binary encoding of the genes (parameters); roulette wheel selection criterion; multi-point crossover operation for crossover.

The algorithm is applied in the next section on a gene expression classification problem from bio-informatics.

4. PARAMETER OPTMISATION OF EVOLVING CONNECTIONIST SYSTEMS FOR BIOINFORMATICS

Genes are complex structures and they cause dynamic transformation of one substance into another during the whole life of an individual, as well as the life of the human population over many generations.

Micro-array gene expression data (see [3) can be used to evolve an EFuNN with inputs being the expression level of a certain number of selected genes (e.g. 11) and the outputs being the classes (e.g. cancer and normal; survive or die). After an EFuNN is trained on gene expression data examples, each taken from a tissue sample, rules can be extracted from the EFuNN that represent disease vs normal tissue profiles [3].

A case study task is taken here which is a prognostic (classification) system for classifying new gene expression data vectors taken from a sample of large B-cell lymphoma cancer tissue into two categories – category *survive* - people who will survive under chemotherapy treatment, and category *fatal* – people who will die despite of the treatment. The gene expression data is taken before treatment. The task and data are available from [5,6]. In [5], Shipp et al have demonstrated the potential of machine learning techniques for prognostic stratification of patients, however their approach misclassified 30% of the patients in terms of predicting the outcome of their treatment. They achieved 70% correct prognosis of cured cases of B-cell lymphoma cancer, and wrongly predicted 12% of the cases as cured in contrast to the actual fatal outcome. This accuracy is not appropriate for a clinical application of the model. The models on the same data presented in Alizadeh et al [6] are not clinically applicable either.

Here we apply a GA parameter optimisation procedure on the 58 available examples (32 cured and 26 fatal) represented as 11-gene expression values (the same genes as derived in [5] as significant when compared to the rest of the 6,000 genes).

For the task of EfuNN parameter optimisation during training, the following parameter value ranges are used:

1) Max-radius, from 0.16 to 0.8, mapped on a 8-bit string;
2) Min-radius, from 0.01 to 0.15, mapped on a 8-bit string;
3) Membership functions, mf, from 1 to 8, mapped on a 3-bit string;
4) Value for the m-of-n parameter, values from 1 to 8, mapped on a 3-bit string;

The following GA parameter values were used:
1) Number of individuals in a population: 16;
2) Mutation rate: 0.005;
3) Termination criterion (the maximum epochs of GA operation): 30;
4) Fitness function: RMS test error.
5) Selection of data from training and testing - 70 % of the data set is randomly selected for training, and the rest of 30 % is selected for testing (and fitness evaluation).

The resulted optimized values of the EFuNN parameters along with the error in 10 experiments (for each of them 70% of the data was selected for training and 30% for testing) are shown in tabl.1.

TABLE 1.

Number of the experiment	Error [in number of wrongly classified examples out of the 30% test examples]	Optimum value for the *Min-radius* found through the GA optimisation	Optimum value for the *Max-radius* found through the GA optimisation	Optimum value for the number of *mf* found through the GA optimisation	Optimum value for the m-of-n found through the GA optimisation
1	0	0.1	0.6	1	1
2	1	0.11	0.72	6	2
3	2	0.11	0.49	5	7
4	1	0.046	0.51	2	1
5	2	0.064	0.39	8	6
6	1	0.055	0.3	4	8
7	2	0.073	0.47	2	6
8	2	0.064	0.25	4	1
9	0	0.11	0.18	4	7
10	2	0.12	0.16	4	5

The max value of testing errors is 2, and the min value is 0, which is a much better result than the Support Vector Machine SVM method, or the k-NN method used in [5].

After the best parameter values for the four EfuNN parameters are found (see experiment 9 from tabl.1), an EfuNN is evolved on the whole data set with the use of these parameter values. Rules that link gene expression values of the 11 genes used with the outcome are shown in table 2. Fig.2 shows all rule nodes of the evolved EfuNN in the 11 dimensional space transformed through principal component analysis into a 2D space of the first two components.

Table 2

Rule 1:
if X2 is(2: 0.86 3: 0.14)
X3 is(1: 0.86 2: 0.14)
X4 is(3: 0.86 4: 0.14)
X6 is(1: 0.15 2: 0.85)
X8 is(1: 0.86 2: 0.14)
X9 is(2: 0.83 3: 0.17)
X11 is(1: 0.86 2: 0.14)
then Class is [1]

Rule 20:
if X1 is(1: 0.86 2: 0.14)
X3 is(1: 0.86 2: 0.14)
X4 is(1: 0.86 2: 0.14)
X5 is(1: 0.14 2: 0.86)
X6 is(1: 0.14 2: 0.86)
X7 is(2: 0.24 3: 0.76)
X8 is(1: 0.86 2: 0.14)
X9 is(1: 0.86 2: 0.14)
X10 is(1: 0.86 2: 0.14)
then Class is [2]

Notation: X1,…,X11 denote the 11 variables that represent the expression values of 11 genes; 1,2 and 3 represent membership functions Small, medium and High; the numbers represent membership degrees, e.g. X1 is(1: 0.86 2: 0.14) means that the expression value of gene 1 belongs to Small to a degree of 0.86 and to Medium to a degree of 0.14.

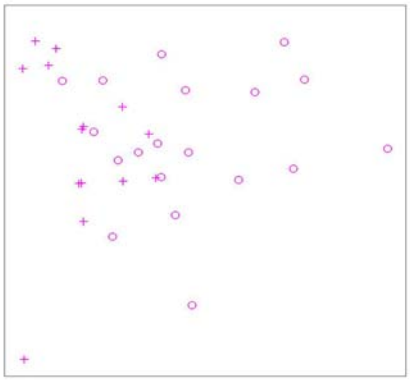

Fig.2 All rule nodes of the evolved EfuNN on the whole data set for the optimal parameter values. The rule nodes are shown in the 11 dimensional space transformed through principal component analysis into 2D space of the first two components. Rule nodes allocated for class "survive" are denoted as "+", while the rule nodes allocated for class "fatal" are denoted as "o".

5. CONCLUSIONS

The ECOS paradigm has a broad spectrum of applications in life sciences, especially in bio-informatics and brain study 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 applications.

The proposed algorithm can be applied to other evolving connectionist models based on clustering, such as RBF (see for example the ZISC system [7]). Major headings, for example, "1. Introduction", should appear in all capital letters, bold face if possible, centered in the column, with one blank line before, and one blank line after. Use a period (".") after the heading number, not a colon.

7. REFERENCES

[1] N. Kasabov, "ECOS: A framework for evolving connectionist systems and the ECO learning paradigm", Proc. of ICONIP'98, Kitakyushu, Japan, Oct. 1998, IOS Press, 1222-1235.

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

[4] N.Kasabov 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.

[5] M. Shipp, et al, Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning, Nature Medicine, vol.8, n.1, January 2002, 68-74

[6] Alizadeh, et al, Distinct types of diffuse large B-cell lymphoma identified by gene-expression profiling, Nature, vol.403, February 2000, 503-511.

[7] ZISC Manual, Silicon Recognition Ltd, http://www.silirec.com