On the Recurrent Neural Network Induction By Evolutionary Computations With Applications in Forecasting

Ion Ciuca and Elena Jitaru

Research Institute for Informatics
8-10 Averescu Avenue
71316 Bucharest
ROMANIA

Abstract: This paper considers recurrent neural networks with one hidden layer. The recurrence is realized in the hidden layer by inner connections. These neural structures are very suitable for dynamic process neurocontrol. The paper presents an algorithm from evolutionary computation for evolving the parameters and structure of these neural networks. Also, by way of example, a simulation application in forecasting, based on time-series of a dynamic process is presented.

Keywords: evolving, induction, evolutionary programming, neural networks

1. Introduction

Neural networks with one hidden layer are good approximators for continuous functions even when these functions are of a classification type. In the following, focus will be on these neural networks, but will also include the recurrence on the hidden layer by inner connections, when these networks are also called dynamical recurrent neural networks [Sudharsanan and Sundareshan, 1990] and the recurrence is called radical recurrence [McDonnell and Waagen, 1994].

The aim is to evolve a recurrent neural network for a given application as regards both the parametric and structural learning (that is, the value of weights and appropriate topology of nodes and links).

The induction algorithm uses evolutionary programming techniques as a component of general evolutionary computation, which, to facilitate a good degree of generalisation, works in two phases. First, the weights and hidden structure are evolved - this continues until the output error falls below a given threshold. Second, the input mutations are evolved.

Recurrent neural networks are more powerful than feedforward nets and their use is very appropriate for simulation of dynamic systems. A dynamic system is one of the form $y_t=f(y_{t-1}, y_{t-2},\ldots,y_{t-k},x_t)$ where y_t and x_t are the output and the input at time t. Recurrent neural networks are also very appropriate when the input vector includes groups of components specified at several consecutive periods, t, t-1, t-2,..., t-k, as often occurs in time-series forecasting scenarios [McDonnell and Waagen, 1994].

2. Recurrent Neural Network Structures

The neural network structures considered in this paper are the so-called (1+1/2)-layer neural networks with one hidden layer. (The hidden layer makes use of a sigmoid activation function, while the activation function in the output layer is the linear identity function f(x)=x). These structures are good universal approximators for nonlinear continuous functions [Cybenko, 1989] [Funahashi , 1989].

To make them suitable for dynamic systems, the paper considers such structures enhanced with recurrence in the hidden layer, as shown in Figure 1.

The dynamics of this architecture is described by the following equations [Karakasoglu et al, 1993]:

$$\dot{z} = -z + r\sigma(z) + wx$$

$$y = W z \qquad (1)$$

On the attainment of the steady-state, when z becomes z^* these conditions are of the form:

$$z^* = r\sigma(z^*) + wx \quad \text{and}$$

$$y = Wz^* \qquad (2)$$

or equivalently

$$z_j^* = \sum_{l=1}^{h} r_{jl}\sigma_l(z_l^*) + \sum_{i=1}^{n} w_{ji}x_i$$

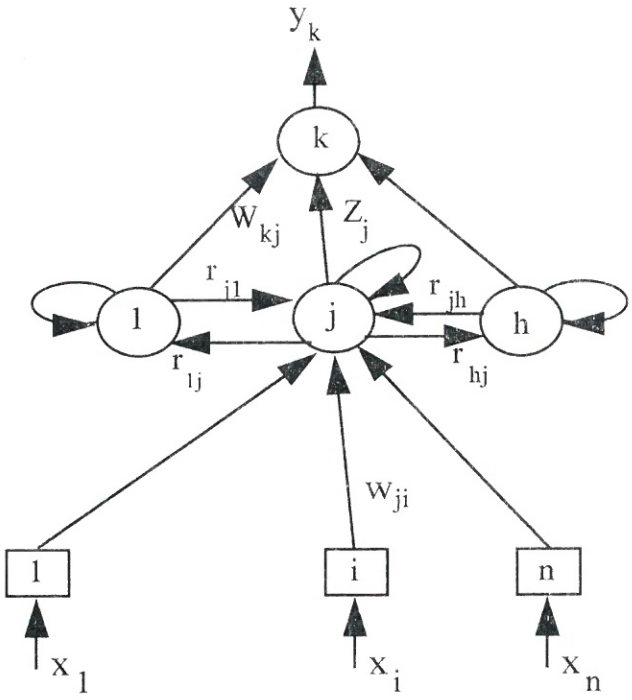

Figure 1. (1+1/2)-Layer Recurrent Neural Network Structure with Recurrence on Hidden Layer

$$y_k = \sum_{j=1}^{h} W_{kj} z_j^* \qquad (3)$$

3. Recurrent Neural Network Induction By Evolutionary Computations

An evolutionary programming algorithm - in essence a systematic multi-agent stochastic search - is used for training and evolving the recurrent neural networks [McDonnell and Waagen, 1994]. Using the argument that the mutation operator is the most suitable for a structure and parametric evolution [Angeline et al, 1994] - though at present there are also implemented genetic algorithms [Thierens, 1996] and combined techniques of genetic algorithms, evolution strategies and evolutionary programming [Yang and Kao, 1996] just emerged - an evolutionary programming (EP), that is a form of evolutionary computation to network induction, is considered. This algorithm simultaneously performs recurrent neural network learning by parameter mutations and, if desired, also a structure evolving by node and link mutations, finally geting an irrelevant feature discarded for classification applications.

The structure mutations alter the number of structure components in the parent networks. They consist in deletions and introductions of nodes and links in input and hidden layers and must proceed on in order - deletions, introductions - on a uniform distribution basis. This order is imposed by the necessity of having no network with zero components in a propagating cycle.

The number of deletions and introductions can vary uniformly over a shrinking interval based on the parent network's fitness. In the experiments below, the maximum number of nodes deleted and added was three and the minimum was one. Also α was considered as being equal to 1.

To maintain the generalisation capability, the input mutations were executed in the final part of learning, once the output error function reached a small value (in the quoted example, this being three times the final output error).

The order of parametric and structure (topology) mutations was the following: weight mutations; node deletion mutations; link deletion mutations; node introduction mutations; link introduction mutations. After all structure mutations, in every cycle a compatibility structure examination has been performed to avoid the situation when there is no path between input and output.

During execution the following variants can be selected via a menu: only parameter mutations; parameter and structure mutations with irrelevant feature discarding; parameter and structure mutations without irrelevant feature discarding. More than that, all the three variants can be executed with or without recurrence on the hidden layer.

The search population consisted of 10 parents, each generating a single offspring by mutations.

4. Simulation Results

Relevant areas where recurrent neural networks seem to be very promising for modelling and simulation will include: neuroidentification, neurocontrol, diagnosis and forecasting. Actually, all these are problems of pattern recognition. In neuroidentification, nonlinear dynamic systems could be identified by means of recurrent neural networks. In neurocontrol, it starts on with a model including a neural network which describes the system or plant to be controlled. The problem is to adapt a second network, the action network, which looks upon the output of the respective system as an input, and gives to the output the desired control [Werbos 1990]. As regards diagnosis, it is also a problem of pattern recognition (and the treatment is as with a problem of neurocontrol).

About forecasting, statisticians say that this is something equivalent to identifying dynamic systems, like the problem of estimating stochastic time-series models. Forecasting is the rational prediction of future events based on information about past and current events. The process is very similar to modelling, where the outcome of an unknown variable is predicted from the known or controllable variables.

Simulation data from a dynamical process seem to be very useful. In general, three types of forecasts are produced:

- short-term forecasts;
- medium-term forecasts;
- long-term forecasts

All the three types of forecasts have been successfully used in economy. In general, forecasting the behaviour of a given system follows two distinct approaches [Refenes et al, 1993]:

The first and most important approach depends on the exact knowledge of the laws underlying a given phenomenon. When this knowledge is expressed in terms of precise equations, it is possible to predict the behaviour of the system once the initial conditions are fully specified.

The second prediction method for relies on discovering strong empirical regularities in system's observations. With this approach a problem will be that regularities are not always obvious and are often masked by noise. A common method for identifying regularities within a by noise contaminated time-series is windowing, where the basic idea is to use two windows W^i and W^o of fixed sizes n and m, respectively to look into the dataset, as shown in Figure 2.

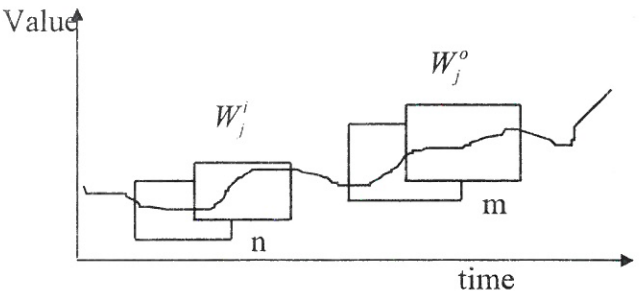

Figure 2. Windowing - Looking for Hidden Correlation and Regularities in Time-Series

In case of a neural network , W^i W^o can be used as a training vector. Both windows are shifted along the time- series using a fixed step size s.

There exist two types of forecasting [Refenes, 1993]:

- multi-step prediction
- single-step prediction

Multi-step prediction is used for long-term forecasting, aiming to identify general trends and major turning points. The prediction is fed back to the network in order to forecast the next period. Using a mapping neural network with a generalization ability, this helps us simulate variants in the future of the analysed system by altering some input parameters.

In fact, if referring diagnosis and treatment, diagnosis is also a problem of pattern recognition and treatment is a problem of neurocontrol.

As process evolution prediction is a mapping and not a classification, input node mutations have been avoided. Data used had the following characteristics: n = 5, m = 1, s = 1, T = size of the training set = 60, and represent the dynamic process of displacements from a hydraulic power barrage.

Table 1. Simulation Results

Execution Variant	Final hidden node number	Links removed %	Generalization degree	Execution time %
Evolution of Parameters only	5	0	80	100
Evolution of both Parameters and Structures	3	35	88	400

The results are presented in Table 1 where the maximum number of hidden nodes was 5, the network was considered with recurrence on the hidden layer and the number of cycles for every pattern on the hidden layer up to steady-state was considered as 3.

Table 1 reveals that the learning time was four times longer when evolving both parameter and structure if compared with when evolving only parameters, but the degree of generalization in predicting the next output was higher.

5. Conclusions

Evolutionary programming is a systematic multi-agent stochastic search. Simultaneous mutations of parameters and structure based on fitness allows that the algorithm discovers appropriate networks quickly. The fitness measure was a cost function given by the minimum square error function. Other criteria could also be introduced, including a minimum number of hidden nodes or links as well as constraints on generalization. Dynamical recurrent neural networks are very appropriate in time-series modelling for forecasting and prediction and, hence, for dynamical processes.

REFERENCES

CYBENKO, G., **Approximation By Superpositions of A Sigmoidal Function**, MATH. CONTROL SIGNAL SYSTEM, 2, 1989, pp. 303-314.

FUNAHASHI, K., **On the Approximate Realization of Continuous Mapping By Neural Networks**, NEURAL NETWORKS, 2, 1989, pp.183-192.

HORNIK, K., STINCHCOMBE, M. and WHITE, H., **Multilayer Feedforward Networks Are Universal Approximators**, NEURAL NETWORKS, 2, 1989, pp.359-366.

SUDHARSANAN, S. I. and SUNDARESHAN, M.K., **Training of A Three-layer Dynamical Recurrent Neural Network for Nonlinear Input-Output Mapping**, Proceedings of International Joint Conference on Neural Networks (IJCNN-91), Vol. II, Seattle, WA, 1990, pp. 111-116.

WERBOS, P., **Backpropagation Through Time: What it Does and How To Do It** , Proceedings of IEEE, Vol. 78, No. 10, October, 1990, pp.1550-1560.

HORNIK, K., **Approximation Capabilities of Multilayer Feedforward Networks**, NEURAL NETWORKS 4, 1991, pp. 251-257.

WERBOS, P.J., **Neurocontrol and Fuzzy Logic:Connections and Designs**, INTERNATIONAL JOURNAL OF APPROXIMATE REASONING, 6, 1992, pp. 185-219.

REFENES, A.N., AZEMA-BARAC, M., CHEN, L. and KAROUSSOS., S.A., **Currency Exchange Rate Prediction and Neural Network Design Strategies**, NEURAL COMPUTING & APPLICATIONS, Vol. 1, No. 1, 1993, pp. 47-58.

MCDONNELL, J.R. and WAAGEN, D., **Evolving Recurrent Perceptrons for Time-Series Modeling**, IEEE TRANS. ON NEURAL NETWORKS, Vol. 5, No. 1, January, 1994, pp. 24-38.

ANGELINE P.G., SAUNDERS, G. and POLLAK, J., **An Evolutionary Algorithm That Constructs Recurrent Neural Networks**, IEEE TRANS. ON NEURAL NETWORKS, Vol. 5, No. 1, 1994, pp. 54-65.

FOGEL D.B., **An Introduction To Simulated Evolutionary Optimization**, IEEE TRANS. ON NEURAL NETWORKS, Vol. 5, No. 1, 1994, pp. 3-14.

THIERENS, D., **Non-Redundant Genetic Coding of Neural Networks**, Proceedings of IEEE ICEC'96, International Conference on Evolutionary Computation, Osaka, 1996, pp. 571-575.

YANG J. M. and KAO, C.Y., **A Combined Evolutionary Algorithm for Real Parameters Optimization,** Proceedings of IEEE ICEC'96, International Conference on Evolutionary Computation, Osaka, 1996, pp. 732-737.