ESANN'1997 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 16-17-18 April 1997, D-Facto public., ISBN 2-9600049-7-3, pp. 327-332

Evidence of Efficiency of Recurrent Neural Networks with ARMA-like Units

Jean-Philippe DRAYE*,[1] , Davor PAVISIC*,
Guy CHERON†,[2], Gaëtan LIBERT*

* Faculté Polytechnique de Mons - *P.I.P.* Laboratory
Rue de Houdain, 9 - B7000 Mons (Belgium)

† University of Brussels - Laboratory of Biomechanics
Avenue Paul Héger, 28 - B1050 Brussels (Belgium)

Abstract. We study in this paper a recurrent neural model where we associate an ARMA process to each neuron-like unit. In order to quantify the effects of the new free parameters on the network, we will use a statistical tool, the analysis of variance (or ANOVA) to establish our results. Finally, practical results highlighting the improvements are presented through the prediction of the Mackey-Glass chaotic signal.

1. Introduction

It is widely known that recurrent neural models can exhibit much powerful capabilities than feedforward neural networks. Nevertheless, at the same time, they are often avoided because of a fear of inordinate learning time and incomprehensible algorithms or mathematics. Moreover, it is usually considered that their learning phase is subject to more instabilities and/or bifurcations.

We will show in this paper that such fears, especially concerning the learning phase, are unjustified. We will also introduce a new neural model by associating to each neuron-like unit an autoregressive-moving average process. We will prove that if this architecture introduces new free parameters, it will also drastically decrease the learning time, improve the training error and avoid bifurcations. To establish our results, we will introduce a new innovative method to analyze the training errors using a statistical tool, the analysis of variance (or ANOVA).

2. Recurrent neural model and learning algorithms

We consider in this work a neural model whose individual units are governed by the following continuous-time equations (see [6]) :

$$T_i \frac{dy_i}{dt} = -y_i + F(x_i) + I_i \qquad \text{with} \qquad x_i = \sum_j w_{ji} \;\; y_j \tag{1}$$

where y_i is the state or activation level of unit i, T_i the adaptive time constant associated to the unit, $F(\alpha)$ a squashing sigmoid-like, I_i an external input (or bias) and x_i is called the total or effective input of the neuron. We wish

[1]Senior Research Assistant of the Belgian National Fund for Scientific Research.
[2]G. Cheron is also with the Department of Neuroscience at the University of Mons-Hainaut.

to emphasize that the previous general model is governed by continuous-time equations. Moreover, we have associated to each neuron-like unit an adaptative time constant that takes part in the learning process. Equations (1) define the most general recurrent model (except the fact that we do not consider time-delay connections). More simple models can be derived by discretization. For example, if we use a time-step Δt equals to 1 with all the time constants T_i equals to 1, the discretization of equations (1) gives the simple recurrent model proposed by Williams and Zipser [8].

Let us note that the learning algorithm for the network governed can be derived either using different techniques : the calculus of variation, the Lagrange multiplier, or even from the theory of optimal control in dynamic programming using the Pontryaguin Maximum Principle [1]. We used this latter method, which is based on the introduction of a Hamiltonian function, to derive the learning equations (for more details, see [4]).

3. Extension of the neural model to ARMA-like units

The autoregressive moving average models (ARMA) are extensively used in the theory of prediction. The ARMA models constitues a general class of models that appear to offer effective predictions of a wide range of dynamic systems in the minimum mean square sense. The general model is based on two distinct elements : the autoregressive and the moving average processes. The ARMA processes have been introduced by G. Box and G. Jenkins for the forecasting and control of time series [2].

In this section, we will introduce some modifications to our network by modeling each link between two units by an autoregressive (AR) filter (also done in [7]) and by adding to each output neurons (for which a target signal is available) a moving average (MA) process. The original aspect of our approach is the fact that our model is governed by continuous equations and can handle dynamic tasks. Moreover, the approach based on the Pontryagin Maximum Principle offers a general framework for the derivation of the learning algorithms of very general continuous-time neural networks.

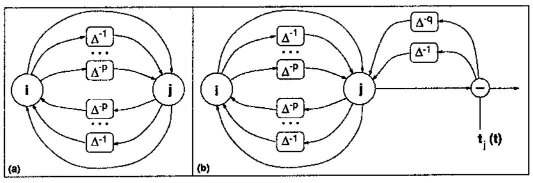

Figure 1: Illustration of the connections between two neurons for an AR process (a) and an ARMA model (b).

We introduce new weights denoted w_{ij}^k (i.e. the autoregressive weight between unit i and j from the k-th delay link). By coherency with the notation of Box and Jenkins [2], we will note p the maximum connection delay (which can be considered as the order of the autoregressive filter) (see Figure 1a). Accordingly, if a target signal is available for a particular unit (as it is the case for the output units in a supervised learning process, this signal is denoted $t_j(t)$ on Figure 1b), we add a moving-average process of order q (see Figure 1b). The new connection weights are denoted v_i^k (i.e. the moving-average weight of unit i from the k-th delay link).

ESANN'1997 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 16-17-18 April 1997, D-Facto public., ISBN 2-9600049-7-3, pp. 327-332

The governing equation of the model is thus the following :

$$T_i \frac{dy_i(t)}{dt} = -y_i(t) + F(x_i(t)) + I_i \tag{2}$$

$$x_i(t) = \begin{cases} \bullet \ \sum_{j=0}^{N} \sum_{k=0}^{p} w_{ji}^{k} \ y_j(t - k\Delta t) \quad \text{for the hidden units} \\ \bullet \ \sum_{j=0}^{N} \sum_{k=0}^{p} w_{ji}^{k} \ y_j(t - k\Delta t) + \\ \quad \sum_{k=1}^{q} v_i^k (y_i(t - k\Delta t) - t_i(t - k\Delta t)) \quad \text{for the output units} \end{cases} \tag{3}$$

Usually, the time-step Δt is equal to the discretization step used for the digital simulation of the network. The learning equations of this network can be derived with the Pontryagin Maximum Principle and a modified Hamiltonian function. These mathematical derivations are not presented here as they are out of the scope of this paper.

4. Influences of the ARMA parameters on the network

The important issue for the present paper is to examine the effect of the introduction of ARMA-like units in the network on the quality and the speed of the identification.

We trained a 20-neuron network to predict a chaotic series. Signal prediction is the classical task where the input to the network is the time-varying signal and the desired output is a prediction of the signal at a fixed time increment in the future. The test signal that we consider is the famous chaotic signal produced by integrating the Mackey-Glass delay-differential equation [5]. This signal provides a useful benchmark for testing predictive techniques. We chose to train the network to predict six time units into the future. We have already proved elsewhere that the adaptive time constants of the model improve the prediction capabilities of the recurrent neural network (see [3]).

4.1 Experimental results

Figure 2 presents the error for the training of three different architectures of the neural network : *I.* classical 20-neuron network, *II.* 20-neuron network with AR connections (order $p = 5$), *III.* 20-neuron network with ARMA processes (orders $p = 5$ and $q = 2$). Each error plot has been obtained by averaging the error curves of 50 different learning phases for each type of architecture. We see that the introduction of AR filters increase the speed and the quality of the learning but we also note that the bifurcations are still present. Accordingly, the ARMA processes also decrease the learning time and hugely improve the quality of the identification but they also suppress the bifurcations that are present in the two other mean learn-

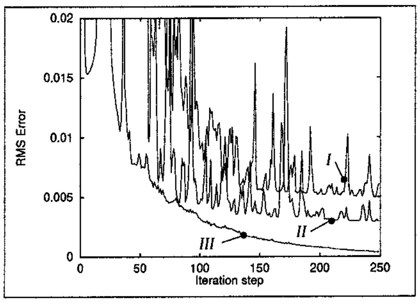

Figure 2: Error plots for the training of three different architectures of neural networks. Each curve is obtained by averaging 50 different learning curves.

that are present in the two other mean learning curves.

4.2 Methods

We will now quantify the influence of the new parameters (w_{ij}^k and v_i) on the network behavior using a new technique based on the statistical technique of the analysis of the variance (also known as ANOVA). The purpose of the analysis of variance (ANOVA) is to test for significant differences between means. ANOVA is a versatile statistical tool for studying the relation between a dependent variable and one or more independent variables. It does not require making assumptions about the nature of the statistical relation, nor does it require that the independent variables be quantitative. Even if the ANOVA procedure compares means, it is called analysis of variance because, in order to test for statistical significance between means, ANOVA actually compares (i.e., analyzes) variances. To validate the results given by the ANOVA technique, we have to compute a statistical degree of significance of these results. The statistical significance of a result is an estimated measure of the degree to which it is *true*. This degree is quantified using the value of the p-level which represents a decreasing index of the reliability of a result. The higher the p-level, the less we can believe that the observed relation between variables in the sample is a reliable indicator of the relation between the respective variables in the population. The ANOVA method is used to analyze the different error functions during the training of the network. The independent variable for the different tests presented below is the architecture of the network (i.e. the order p and q of the ARMA processes). It is interesting to note that these variables are qualitative. For each architecture of the network, we trained it for 50 times (using a different initial random weights distribution) during 1,000 epochs. The dependent variables are the values of the error signal every 20 iterations (the number of dependent variables was restricted to 50 to limit the computation time). The goal of the analysis of the variance of the different error functions is to determine if these ones are statistically different depending on the network architecture. Moreover, ANOVA provides a comparison where the parameter "iteration step" disappears i.e. the ANOVA technique gives the overall error mean value corresponding to a particular architecture and tells if this one is statistically different from the error mean value of another architecture.

There are mainly three interests for a comparison based on the computation of the error mean value over 1,000 epochs : *(i)* the bifurcations along the training process increase the value of the error mean value; *(ii)* if the training process learns fast, the error decreases also quickly and the mean value is smaller; *(iii)* the asymptotic value of the error (i.e. which is an image of the quality of the training) has a direct influence on the error mean value. To sum up, we can say that a small mean error value indicates simultaneously that the learning process converges quickly, that few bifurcations were present and that the asymptotic value of the error was small.

4.3 Influence of the autoregressive parameters

We will first study the influence of the autoregressive parameters w_{ij}^k (and their order p) on the learning process. For that, we consider a 20-neuron network with autoregressive neuron-like unit (no moving average process is present at this step). The order p of the autoregressive filters varies from 0 (no AR process, i.e. the neuron presented in Section 2) to 10.

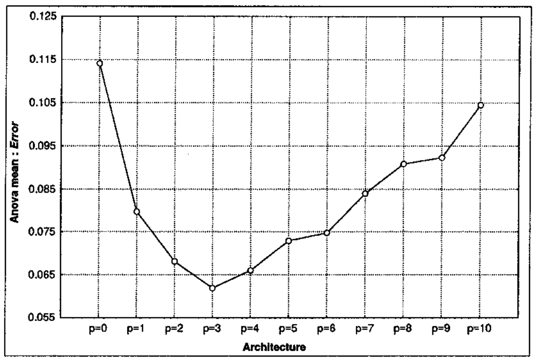

Figure 3: ANOVA results concerning the influence of the order p of the AR processes on the error function. The network consists in 20-neurons. The order p ranges from 0 to 10. The confidence p-level equals 0.052.

Figure 3 depicts the results of the ANOVA treatment on these architectures. The p-level of this analysis equals 0.052; that means that the absolute maximum error probability on the conclusions is less than 5.2%. We clearly see the positive influence of the autoregressive weights (w_{ij}^k for $k > 0$). The overall mean error for the classical network without ARMA process hits 0.114. The error decreases drastically with the order p of the AR processes. We note that only a first-order AR process leads to an ANOVA error parameter of 0.079 (a 30 % gain). The error parameter decreases to reach a minimum for p equals three to reach a value of 0.062.

The first conclusion is thus that the introduction of three new parameters for every neuron can divide the error mean value by a factor of two. We also see that for an order p that is greater than three, the ANOVA error parameter increases continuously. It is easy quite easy to understand this fact : the greater the order p is, the more free parameters w_{ij}^k the learning algorithm has to adapt. The training process is slower : the error mean value is higher according to the second point presented in Section 4.2.

4.4 Influence of the moving average parameters

We now consider the influence of the second parameters of the ARMA processes : the moving average parameters v_i^k. We wish to emphasize that these parameters are only associated to the output units for which a target signal is available during the training phase.

We consider a 20-neuron network with autoregressive process of 10^{th} order ($p = 10$). A moving average process is associated to the only output neuron of the network. The order q of this moving average filter varies from 0 (no MA process) to 9. Figure 4 depicts the results of the ANOVA treatment on these architectures. The p-level of this analysis equals 0.048; that means that the absolute maximum error probability on the conclusions is less than 4.8%. Figure 4 gives us a statistical proof of the observations of Section 4.1 : the small amount of free parameters added to the output unit impressively improve the quality of the learning phase. In the particular case of Figure 4, the best improvement is obtained for a first order MA process. The performances of the system remain quite constant for the other orders.

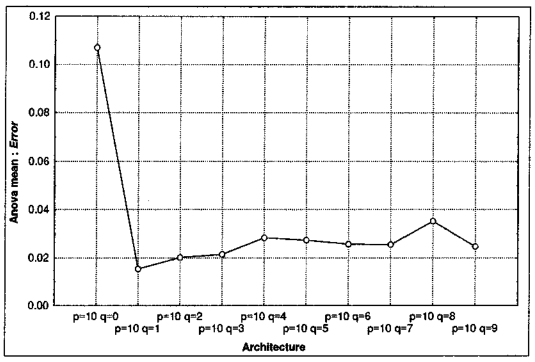

Figure 4: ANOVA results concerning the influence of the order q of the MA processes on the error function. The network consists in 20-neurons with AR filter (p=10). The order q ranges from 0 to 9. The confidence p-level equals 0.048.

5. Conclusion

In this paper, we have shown the interesting new capabilities of recurrent neural networks where an ARMA process is associated to each neuron. Using the ANOVA technique, we gave an original and statistical proof of these enhancements : the learning time has decreased, the training error is better and there are fewer bifurcations. Even if these architectures introduce some new free parameters to adapt, we believe that they can bring new features to recurrent neural models.

References

[1] V.G. Boltyanskii, R.V. Gamkrelidze, E.F. Mishchenko, and L.S. Pontryagin. The Maximum Principle in the theory of optimal processes of control. In *Optimal and Self-organizing Control*, pages 262–266. R. Oldenburger (Editor), 1966.

[2] G.E.P. Box and G.M. Jenkins. *Time series analysis : forecasting and control.* Holden-Day Inc., San Francisco, 1976.

[3] J.P. Draye, D. Pavisic, G. Cheron, and G. Libert. Adaptive time constants improve the prediction capability of recurrent neural networks. *Neural Processing Letters*, 2(3):12–16, 1995.

[4] J.P. Draye, D. Pavisic, G. Cheron, and G. Libert. Dynamic recurrent neural networks : a dynamical analysis. *IEEE Transactions on Systems, Man, and Cybernetics- Part B: Cybernetics*, 26(5):692–706, 1996.

[5] M.C. Mackey and L. Glass. Oscillations and chaos in physiological control systems. *Science*, 197:287–289, 1977.

[6] B.A. Pearlmutter. Learning state space trajectories in recurrent neural networks. *Neural Computation*, 1:263–269, 1989.

[7] E.A. Wan. Time series prediction using a connectionist network with internal delays. In A.S. Weigend and N.A. Gershenfeld, editors, *Time Series Prediction - Forecasting the Future and Understanding the Past*, pages 195–217. Addison-Wesley, 1993.

[8] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280, 1989.