

Evolving Predictive Neural Models for Complex Processes

Matteo DE FELICE

Dept. of Computer Science and Automation, University of Rome "Roma Tre"
Via della Vasca Navale 79, 00146 Rome, Italy

Mauro ANNUNZIATO

Energy, New technology and Environment Agency (ENEA)
Via Anguillarese 301, 00123 Rome, Italy

Ilaria BERTINI

Energy, New technology and Environment Agency (ENEA)
Via Anguillarese 301, 00123 Rome, Italy

Stefano PANZIERI

Dept. of Computer Science and Automation, University of Rome "Roma Tre"
Via della Vasca Navale 79, 00146 Rome, Italy

Stefano PIZZUTI

Energy, New technology and Environment Agency (ENEA)
Via Anguillarese 301, 00123 Rome, Italy

ABSTRACT

In this work we present a study on Artificial Neural Networks (ANN) performance with different topologies and training algorithms in order to develop a model of a dynamical system, focusing on the effect of unknown inputs (disturbance). Therefore, we study the ANN performance in model identification and prediction by training them using traditional gradient based and evolutionary methods.

Tests have been made with different prediction horizons on two experimentations : without disturbance and with a pulse train unknown disturbance. Results show that without disturbances the performance of gradient trained ANN is slightly better than that of evolutionary trained ANN. The situation is different when in the presence of disturbance: gradient trained ANN performance gets much worse compared to evolutionary ANN.

Keywords : Evolutionary Neural Networks, Complex Systems, Modelling, Dynamical Systems, Artificial Life

1. INTRODUCTION

In modern systems theory, a typical optimal control problem consists of finding the optimal regulation to apply to the modelled process that minimize/maximize some certain criteria of optimality (i.e. optimal tracking, min/max cost function, etc..). The case in which the optimisation routine needs a model of the process this dual necessity becomes clear: the model has to be as close as possible to the modelled process (especially for transient regimes). Therefore, minimizing the prediction error of the future samples of a signal generated by a dynamical non-linear process, like in most real systems, is a very critical task in optimisation and control problems. In order to cope with this situation, in the past years the classical approach of modelling a dynamical system was based on the study of the chemical-physical properties of the process itself with the aim to build an

appropriate mathematical representation, establishing a first principles model. This kind of methodology fails when the real system is characterized by a chaotic behaviour and is non stationary [17].

For this reasons, in the last decades there was a raised interest in black box methodologies, in which the mathematical approach is replaced by an empirical study based on input-output signals. In this framework the theory of Artificial Neural Networks (ANN), which are proved to be powerful instruments to solve complex modelling problems for non-linear systems [16], is inserted. A detailed comprehensive foundation on neural networks can be found in [13]. Most ANN applications in dynamic modelling have involved the use of feed forward networks to identify a model in autoregressive form, in which the inputs of the network are the past values of the system inputs-outputs, and the network output corresponds to the current system output; these values identify a time window of the system inputs and outputs to be chosen wide enough to capture the essentials of the plant dynamics [15]. Recurrent neural networks, due to their internal structure, are suited to provide a state-space description of the system, but their use is limited by the additional degree of difficulty involved in the training phase and by stability problems due to the presence of the feedback loops [12]. The main drawback of ANN is that different topologies and training algorithms may lead to remarkable differences in the error between the real and the predicted signal. Recent investigations in combining evolutionary algorithms (EA) and ANN have shown very promising results [6] giving rise to a new class of ANN called Evolutionary Neural Networks (ENN) [21]. ENN have been developed in these last years to improve the performances of the created model in presence of noises, disturbances and other modelling uncertainties [9]. Indeed, in such cases it seems to be necessary to update the model to the new variations and to make possible an on-line modelling task: one approach to reach the goal consist in combining the neural networks modelisation capabilities with the adaptation property of evolutionary algorithms. In an off-line contest EAs may be very useful in

tasks like training connection weights, network design topology, or both of them [6]. Their usefulness must be principally regarded in higher robustness, poor probability to get stuck in local minima and major simplicity of the resulting topology network with respect to those obtained with classical training algorithms [10] like Back-Propagation (BP). Recently, a study on how inertia affects the prediction of ANN models has been carried out[11], and in the following paragraphs it is shown how the presence of unknown disturbances affects the prediction of ANN modelling.

2. THE PROCESS

The process we carried out for the experimentation consists of two linearized Continuous flow Stirred Tank Reactor (CSTR) models in parallel (figure 1). The CSTR model describes an exothermic diabatic reaction of the first order and it is commonly studied for his characteristics [18][20]. The equations, written in dimensionless form [19], are the following:

$$\begin{aligned} \dot{x}_1 &= q(x_{1s} - x_1) - \Phi x_1 \kappa(x_2) \\ \dot{x}_2 &= q(x_{2s} - x_2) - \delta(x_2 - x_3) + \beta \Phi \kappa(x_2) \\ \dot{x}_3 &= \delta_1[q_c(x_{3s} - x_3) + \delta \delta_2(x_2 - x_3)] \end{aligned} \quad (1)$$

where x_1 , x_2 and x_3 are respectively dimensionless concentration, reaction temperature and cooling-jacket temperature. The first manipulated input is q_c which represents the cooling-jacket flow rate and the second input q is the reactor flow rate. The only output of the system is the dimensionless concentration x_1 . The other parameters are explained in table 1 with the values we set.

Name	Value	Explanation
x_{1s}	0.2028	Steady state of concentration
x_{2s}	5	Steady state of reaction temperature
x_{3s}	0.4079	Steady state of cooling-jacket temperature
x_{1f}	1	Dimensionless reactor feed concentration
x_{2f}	0	Dimensionless reactor feed temperature
x_{3f}	-1	Dimensionless cooling-jacket feed temperature
Φ	0.25-0.53	Damkholer number
q_{cs}	0.9785	Steady state of cooling-jacket flow rate
q_s	1	Steady state of reactor flow rate
β	8	Dimensionless heat of reaction
δ	0.3	Dimensionless heat transfer coefficient
δ_1	10	Reactor to cooling jacket volume ratio
δ_2	1	Reactor to cooling jacket density heat capacity ratio
γ	20	Dimensionless activation energy

Table 1 : process parameters

The term $\kappa(\cdot)$ represents the dimensionless Arrhenius reaction rate which corresponds to:

$$\kappa(x) = e^{\frac{x}{1+\frac{x}{\gamma}}} \quad (1)$$

The model we used is represented in the schema in figure 1

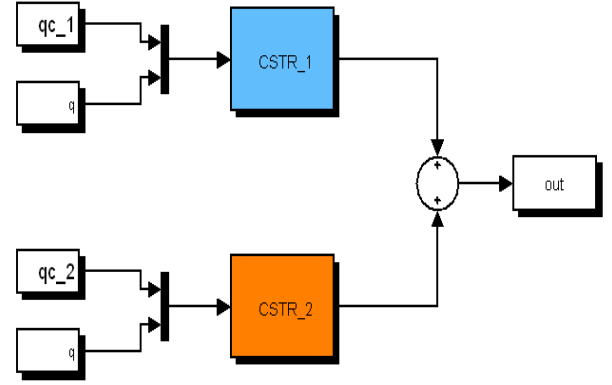


Figure 1 : process layout

As stated before, the process consists of two CSTR linearized models with different Damkholer numbers, therefore we have two different inertias (the length of the transient regime), one fast and one slow. In order to simulate a non-stationary environment (like most of real situations are), we consider the input q (reactor flow rate) of the CSTR model as “disturbance”, because it is not given to the neural models. The system matrices of the model with “disturbance” are the following:

$$A = \begin{bmatrix} -q_s - \Phi K & -\Phi x_{1s} K' & 0 \\ \beta \Phi K & -q_s - \delta - \beta \Phi x_{1s} K' & \delta \\ 0 & \delta \delta_1 \delta_2 & -\delta_1 q_{cs} - \delta \delta_1 \delta_2 \end{bmatrix} \quad (2)$$

$$B = \begin{bmatrix} 0 & (x_{1f} - x_{1s}) \\ 0 & (x_{2f} - x_{2s}) \\ \delta_1 (x_{3f} - x_{3s}) & 0 \end{bmatrix} \quad (3)$$

$$\text{With } K = e^{\frac{x_{2s}}{1+\frac{x_{2s}}{\gamma}}} \quad (4)$$

$$\text{and } K' = e^{\frac{x_{2s}}{1+\frac{x_{2s}}{\gamma}}} \frac{1}{\left(1+\frac{x_{2s}}{\gamma}\right)^2} \quad (5)$$

The reactor flow rate q (disturbance) is modelled as a train pulse of 0.1 Hz of frequency, 0.15 of amplitude and a pulse width of 40%.

Both CSTR have a random step ranging between 0 and 0.5 every 6.5 seconds. The step is filtered by a $\frac{1}{s+1}$ transfer function.

3. THE EVOLUTIONARY ENVIRONMENT

The implemented evolutionary environment is an *Artificial Life* (ALIFE) environment [14]. This approach has been tested on the optimisation of static well known benchmarks, as the Travelling Salesman Problem, the Chua's circuit and the Kuramoto dynamical system [2][5], as well as real cases [1][3][7][8]. The ALIFE context is a two-dimensional lattice (life space) representing a flat physical space where the artificial individuals (or autonomous agents) can move around.

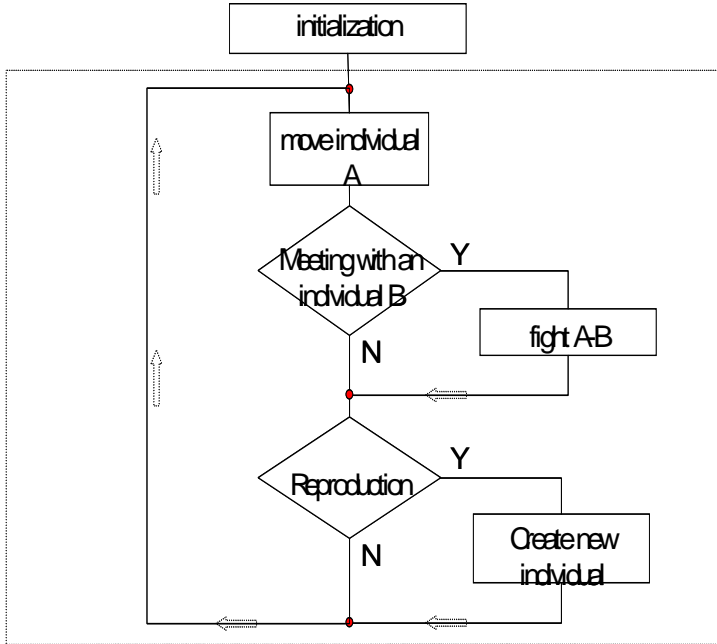


Figure 2 : main algorithm flow

At each iteration (or life cycle), individuals move in the life space and, in case of meeting with other individuals, interaction occurs. Each individual has a particular set of rules that determines its interactions with other agents basically based on a competition for energy in relation to the performance value. Individuals can self-reproduce via haploid mutation which occurs only if the individual has an energy greater than a specific birth energy. In fact, during reproduction, an amount of energy equal to the birth energy is transferred from the parent to the child. In the haploid reproduction a probabilistic test for self reproduction is performed at every life cycle and a probabilistic-random mutation occurs on the genes according to the mutation rate and the mutation amplitude, which are evolutionary themselves [4]. When two individuals meet, fight occurs. The winner is the individual characterized by a greater value of performance and the loser transfers an amount of energy (fighting energy) to the winner. At every life cycle each individual age is increased and when the age reaches a value close to the average lifetime, the probability of natural death increases. This ageing mechanism is very important to warrant the possibility to lose memory of very old solutions and follow the process evolution. Another mechanism of death occurs when the individual reaches the null energy due to reproduction or fighting with other individuals.

In this way, when there is a reproduction event the population size increases of one unit, while in a death event the population size decreases of one unit. Therefore, the population size is dynamic and it is limited by the dimension of the physical space.

Three blocks compose the data structure of each individual: the genotype, the information and the status. The first one includes a collection of behavioural parameters regarding dynamics, reproduction and interaction. The information block includes a series of parameters related to the process to control: the regulation and measurement values; both the information and the genotype don't change during the individual life. The status parameters include dynamics and structural parameters (position, direction, curvature, wire description), age, energy

and performance values. These parameters change during the individual life. An important feature that let us think of this strategy as the right one is the finiteness of the individual life. The optimisation, in fact, succeeds in keeping itself updated on the evolution of the process by continuously renewing the population on line. This is allowed by an ageing mechanism, according to which each individual dies after a fixed number of life cycles (average life). The performance is updated using an external problem-specific model. This is due to the possible changes in the unknown variables of the process not represented in the genotype. For this reason the performance variable is located in the status block.

For interested reader, a detailed description of the methodology is reported in [2][4].

4. EXPERIMENTATION

In our study we used two training algorithms: a classic BP implementation and the ALIFE environment.. The whole dataset (3000 points sampled every 0.3 seconds) has been partitioned in training and testing in two equal parts (50%-50%) and the same sets have been used for both experimentations. Results refer to the testing stage.

For both the ALIFE and BP experimentation we carried out the following tests and compared the results (figures 4-7) on different prediction horizons h ($h=1, \dots, h=40$).

- Only dynamics as input with no disturbance
- Dynamics + regulations as input with no disturbance
- Only dynamics as input with disturbance
- Dynamics + regulations as input with disturbance

The last two cases involve the presence of an unknown pulse train disturbance (parameter q in Eq. (1), figure 1) .

Therefore, the neural topology (figure 3) was set to :

- Input neurons : six (representing the last six past samples of the signal) + $2h$ (representing the regulations of the two CSTR from time t to $t+h$, where h is the prediction horizon) .
- Hidden neurons : six with sigmoidal transfer function.
- Output neurons : one (representing the prediction of the signal at time $t+h$) with linear transfer function.

The same topologies were used for both the ALIFE and BP tests.

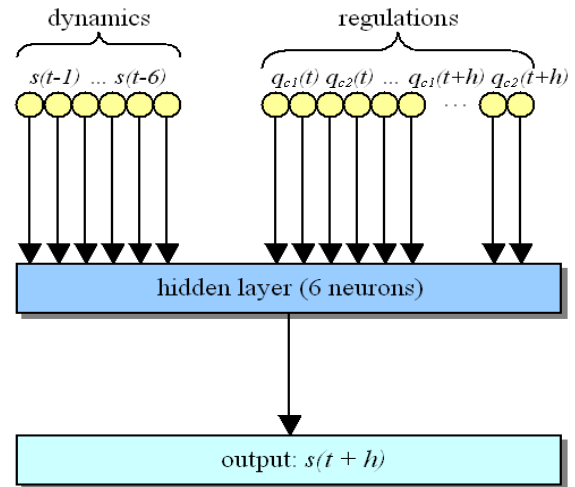


Figure 3 : neural topology

In the ALIFE approach each individual represents a FF-ANN, in competition with each other to optimise the fitness function (Eq. 6), which indicates how accurate the neural model is.

$$P_{fit} = 1 - E_{rmse} \quad (6)$$

where E_{rmse} is the normalized mean square error, described by the following:

$$E_{rmse} = \sqrt{\frac{\frac{1}{2} \sum_{i=1}^M (y(i) - \bar{y}(i))^2}{M}} \quad (7)$$

where M is the dataset size, $y(\cdot) \in [0,1]^M \subset \mathfrak{R}^M$ is the normalized modelled process output and $\bar{y}(\cdot) \in [0,1]^M \subset \mathfrak{R}^M$ is the neural model output.

The experimental set-up consisted of 30000 request of performances which are equivalent to approximately 800 generations. The physical space was set to 25 x 25 cells (corresponding to a maximum population size of 625 individuals) with an initial population of 210 individuals.

As BP implementation we used the MATLAB Neural Network Toolbox based on the gradient descent algorithm (traindx) with constant momentum set to 0.9, learning rate to 0.01 and 1600 training epochs.

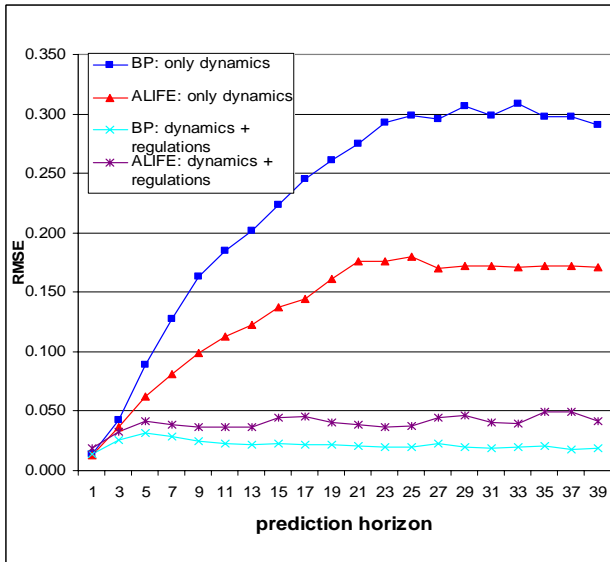


Figure 4 : RMSE comparison with no disturbance

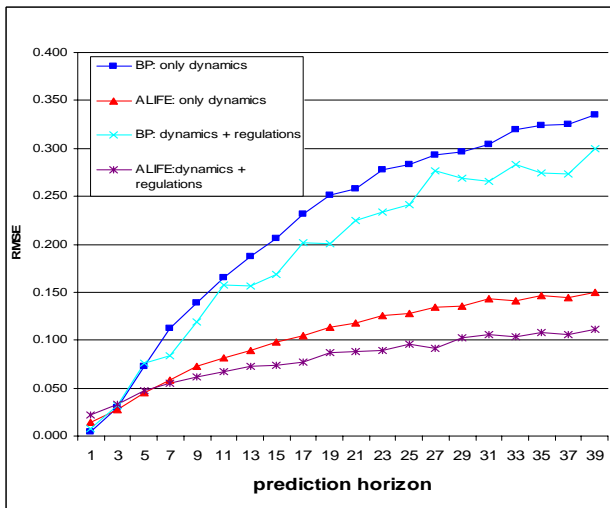


Figure 5 : RMSE comparison with disturbance

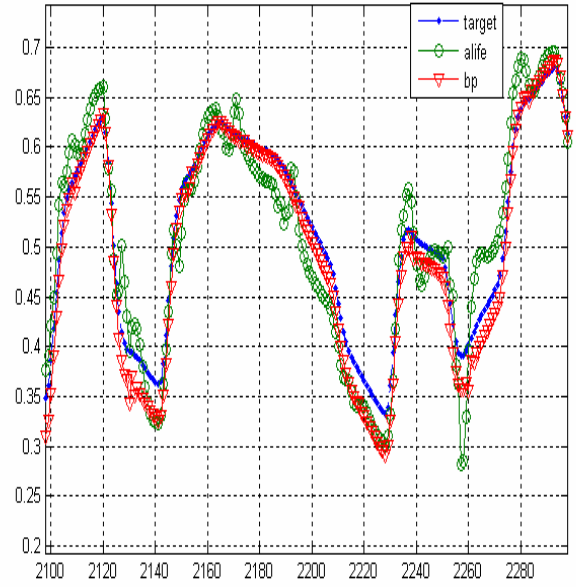


Figure 6 : signal comparison with no disturbance (h=10)

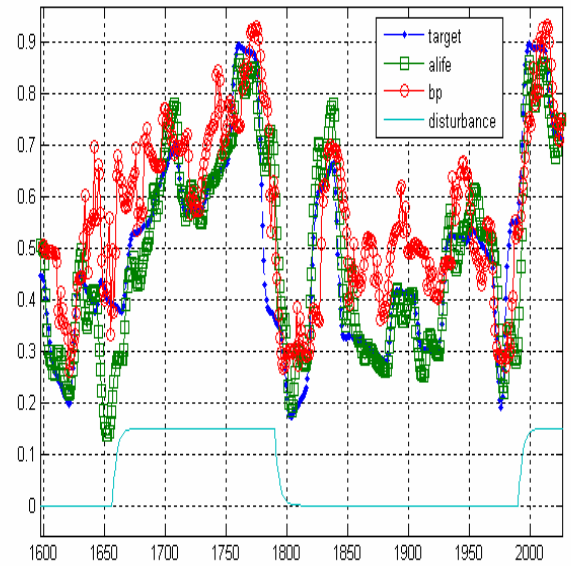


Figure 7 : signal comparison with disturbance (h=10)

In the following tables we report the RMSE averaged on all the different prediction horizons from h=1 to h=40.

	BP	ALIFE
Only dynamics	0.22	0.135
Dynamics+regulations	0.02	0.04

Table 2 : average RMSE without disturbance

	BP	ALIFE
Only dynamics	0.22	0.104
Dynamics+regulations	0.19	0.08

Table 3 : average RMSE with disturbance

Experimental results let us do two main considerations. The first one concerns the input of the model. It is clear that in

general regulations remarkably improve the modelling capabilities. The second one regards the BP vs. ALIFE comparison. In a stationary case, with no disturbance, the BP algorithm performs slightly better than ALIFE, but in a non-stationary situation, with a nasty disturbance, ALIFE clearly outperforms BP.

5. CONCLUSION

Minimizing the prediction error of the future samples of a signal generated by a dynamical non-linear process, like in most real systems, is a very critical task in optimisation and control problems. Artificial neural networks are surely important tools to carry out effective predictive models for control purposes, but different topologies and training algorithms lead to remarkable differences in the error between the real and the predicted signal. Therefore, in this work we presented a study on the ANN performance with different topologies and training algorithms in order to develop a model of a chemical process, focusing on the effect of unknown inputs (disturbance). The model we have used in our work consist of two in-parallel exothermic CSTRs (Continuous Stirred Tank Reactor). Each of these is a linearised dynamical model with two inputs describing an exothermic diabatic irreversible first-order reaction.

Therefore, we studied the performance of multilayer feed-forward neural networks in model identification and signal prediction.

We considered feed-forward ANN with two class of inputs: past samples of the signal (time history) with and without regulations.

We used two training algorithms: a classic implementation of gradient-descent back-propagation and an algorithm based on Artificial Life (ALIFE) environment. Neural networks are trained with a subset of the entire dataset (training set) and then tested on the testing set.

Tests have been made with different prediction horizons on two experimentations : without disturbance (stationary case) and with an unknown pulse train disturbance. Results showed that without disturbance the ANN with regulations as input have the best RMSE (very close to zero) even with high prediction horizons and the BP performance is slightly better than the ALIFE's one. The situation is different when in presence of disturbance (non stationary situation) : BP performance gets much worse, also with regulations as input, compared to the ANN evolved with ALIFE.

Future work will involve experimentation on real non stationary situations as well as further research on signal analysis and on different training algorithms and topologies.

6. REFERENCES

- [1] Annunziato M. , Bertini I. , Pannicelli A., Pizzuti S. , Tsimring L., "Complexity and Control of Combustion Processes in Industry", Proc. of CCSI 2000 Complexity and Complex System in Industry, Warwick, UK, 2000
- [2] Annunziato M. , Bruni C., Lucchetti M. , Pizzuti S. "Artificial life approach for continuous optimisation of non-stationary dynamical systems", Integrated Computer-Aided Engineering, vol. 10, n. 2, 2003, pp. 111-125
- [3] Annunziato M. , Lucchetti M., Orsini G., Pizzuti S. , "Artificial life and on-line flows optimisation in energy networks", IEEE Swarm Intelligence Symposium, Pasadena (CA), USA, 2005
- [4] Annunziato M., Bertini I., Iannone R., Pizzuti S. "Evolving feed-forward neural networks through evolutionary mutation parameters", 7th International Conference of Intelligent Data Engineering and Automated Learning (IDEAL 06), Burgos, Spain, 2006, 554-561
- [5] Annunziato M., Bertini I., Lucchetti M., Pannicelli A., Pizzuti S. "Adaptivity of Artificial Life Environment for On-Line Optimization of Evolving Dynamical Systems", in Proc. EUNITE01, Tenerife, Spain, 2001
- [6] Annunziato M., Bertini I., Lucchetti M., Pizzuti S., "Evolving Weights and Transfer Functions in Feed Forward Neural Networks", Proc. EUNITE2003, Oulu, Finland, 2003
- [7] Annunziato M., Bertini I., Pannicelli A. and Pizzuti S. "A Nature-inspired-Modeling-Optimization-Control system applied to a waste incinerator plant", 2nd European Symposium NiSIS'06, Puerto de la Cruz, Tenerife (Spain), 2006
- [8] Annunziato M., Bertini I., Pannicelli A., Pizzuti S. "Evolutionary Control and On-Line Optimization of an MSWC Energy Process", Journal of Systemics, Cybernetics and Informatics, Vol.4, Num. 4, 2006
- [9] Annunziato M., Lucchetti M., Pizzuti S., "Adaptive Systems and Evolutionary Neural Networks : a Survey", Proc. EUNITE2002, Albufeira, Portugal, 2002
- [10] Balakrishnan K., Honovar V. "Evolutionary design of neural architectures – a preliminary taxonomy and guide to literature". Technical report CS-TR 95-01, Iowa State University, Ames, IA, 1995.
- [11] Di Giamberardino M., Annunziato M., Pizzuti S. , "System modelling and analysis in optimal control using evolutionary artificial neural networks", EMCSR2006, Vienna, Austria, 2006, 495-500
- [12] Eaton J.W., Rawlings J.B., Ungar L.H. "Stability of neural net based model predictive control". Neurocomputing, Vol. 15, 1997, pp. 183-223.
- [13] Haykin S. "Neural Networks, a comprehensive foundation". Prentice Hall. New Jersey, 2nd edition. ISBN 0-13-273350-1, 1999.
- [14] Langton, C., "Artificial Life", Addison-Wesley, Redwood City/CA, USA, 1989
- [15] Narendra K.S., Parthasarathy K. "Identification and control of dynamical systems using neural networks". IEEE Trans. Neur. Netw., Vol. 1, No. 1, 1990, pp. 4-27.
- [16] Nikravesh M., Farell A.E., Stanford T.G. "Model identification of non linear time variant processes via artificial neural network". Computes and Chemical Engineering, Vol. 20, No. 11, 1996, pp. 1277-1290.
- [17] Prasad V., Bequette B.W. "Non linear system identification and model reduction using artificial neural networks". Computes and Chemical Engineering, Vol. 27, 2003, pp. 1741-1754.
- [18] Russo L.P., Bequette B.W. "Impact of process design on the multiplicity behaviour of a jacketed exothermic CSTR", AIChE Journal, Vol. 41, 1995, pp 135-147
- [19] Russo, L.P. and Bequette, B.W., "State-Space versus Input/Output Representations for Cascade Control of Unstable Systems", Ind. Eng. Chem. Res., 36, 6, 1997, 2271 - 2278
- [20] Saraf V.S., Bequette B.W., "Auto-tuning of cascade controlled open-loop unstable reactors", American Control Conference, 2001. Proceedings of the 2001, Vol.3, Iss., 2001, pp. 2021-2026
- [21] Yao X. "Evolving Artificial Neural Networks", in Proceedings of the IEEE, 87(9): 1423-1447, 1999