Computer Assisted Mechanics and Engineering Sciences, 14: 715–727, 2007.

Heuristic modeling using recurrent neural networks: simulated and real-data experiments

Piotr Przystałka
Silesian University of Technology, Department of Fundamentals of Machinery Design
ul. Konarskiego 18A, 44-100 Gliwice, Poland

(Received in the final form September 18, 2007)

The focus of this paper is on the problems of system identification, process modeling and time series forecasting which can be met during the use of locally recurrent neural networks in heuristic modeling technique. However, the main interest of this paper is to survey the properties of the dynamic neural processor which is developed by the author. Moreover, a comparative study of selected recurrent neural architectures in modeling tasks is given. The results of experiments showed that some processes tend to be chaotic and in some cases it is reasonable to use soft computing models for fault diagnosis and control.

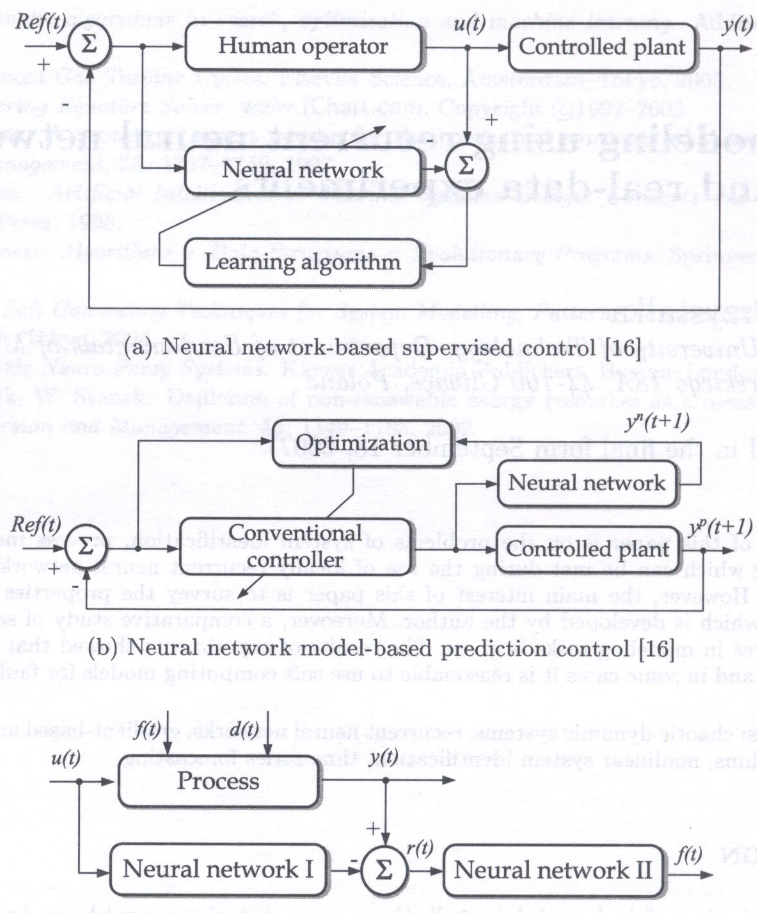
Keywords: chaotic dynamic systems, recurrent neural networks, gradient-based and soft computing learning algorithms, nonlinear system identification, time-series forecasting

1. Introduction

The growing complexity of industrial installations causes serious problems in the modern control system design and analysis. Chemical refineries, electrical furnaces, water treatments and other industrial plants are complex systems and in some cases can not be precisely described by classical mathematical models. Therefore, they are difficult to control using conventional model-based methods. Nevertheless, in practise, skilled operators are often able to achieve fairly acceptable control quality. On the other hand, modern industrial systems are subject to faults in their components. This is the reason for that fault tolerance of modern automatic control systems is gained more and more attention in recent years [2, 13, 14].

Heuristic modeling of objects and processes, as pointed out by W. Moczulski, is not a single methodology [18, 19]. This term is closely linked to *soft computing* philosophy of L. Zadeh, which constitutes the basis of a fusion of several methodologies such as fuzzy logic, neural networks and genetic algorithms [34]. Heuristic modeling takes advantage of the intuition in the same way as soft computing techniques and is similar to the knowledge discovery in databases (main source of knowledge are databases which collect data acquired by industrial systems). Due to the fact that there are a large number of real-world problems that can not be solved by conventional (hard) computing methods it is well-founded to use the human mind-based reasoning which is included in soft computing. It is very important to remember, that soft computing is not always an accurate solution for our task. In numerous industrial applications hard computing plays a major role. As we can observe, soft computing methods are usually combined with traditional hard computing approaches in industrial products instead of using them separately.

For instance, in the paper [16] the authors explore adaptive neural network controllers and notice that new and innovative neural network architectures are being introduced. They discuss different control schemes (e.g. Fig. 1a,b). In their opinion, much more complicated neural structures together with standard hard computing approaches may lead to enable more efficient implementation of these networks to real-world applications in control systems. There are also other industrial applications of soft computing techniques in fault-tolerant control systems [4, 5, 7, 8, 13, 14, 28, 30] confirm that



(a) Neural network-based supervised control [16]

(b) Neural network model-based prediction control [16]

(c) Neural network model-based fault diagnosis [14, 15]

Fig. 1. Neural network-based controllers (a), (b) and general scheme of a neural model-based fault diagnosis (c)

neural networks, genetic algorithms, fuzzy logic and others may aid conventional automatic control systems. Moreover, the idea and informal definition of the concept *"fusion of soft computing and hard computing"* may be found in the paper [22]. After analyzing a large number of application papers the authors introduce 12 categories and the qualitative measure for the fusion of soft computing and hard computing (e.g. the fusion grade of *"soft computing is cascaded with hard computing"* is low, the fusion grade of *"hard computing is designed by soft computing"* is high/very high, etc.).

Generally, recurrent neural networks are able to represent dynamic behavior of some systems but they are not able to model their stochastic behavior themselves. Therefore, there is the need to elaborate much more general neuron models which can be used for modeling both deterministic and stochastic systems simultaneously. In this paper the author discusses the general dynamic neural model with linear dynamic systems in the activation and feedback blocks. Dynamic features of the unit (deterministic and stochastic) are obtained by introducing linear dynamic systems into its structure. As it is presented in Fig. 2b, the largest Lyapunov exponent of the dynamic neural unit as a function of the input signal $u(k)$ and the refraction parameter a_1^A indicates a chaotic character of its behavior. This may make possible to identify much more precise models of objects and processes.

The paper is organized as follows. Section 2 describes mathematical preliminaries that are necessary for heuristic modeling of objects and processes using recurrent neural networks. In particular, there are some equations that can be used for description of the dynamic neural unit. There may be also found the general structure of a locally recurrent neural network and some learning rules. For comparison purposes, an identifier using a globally recurrent neural network has been designed.

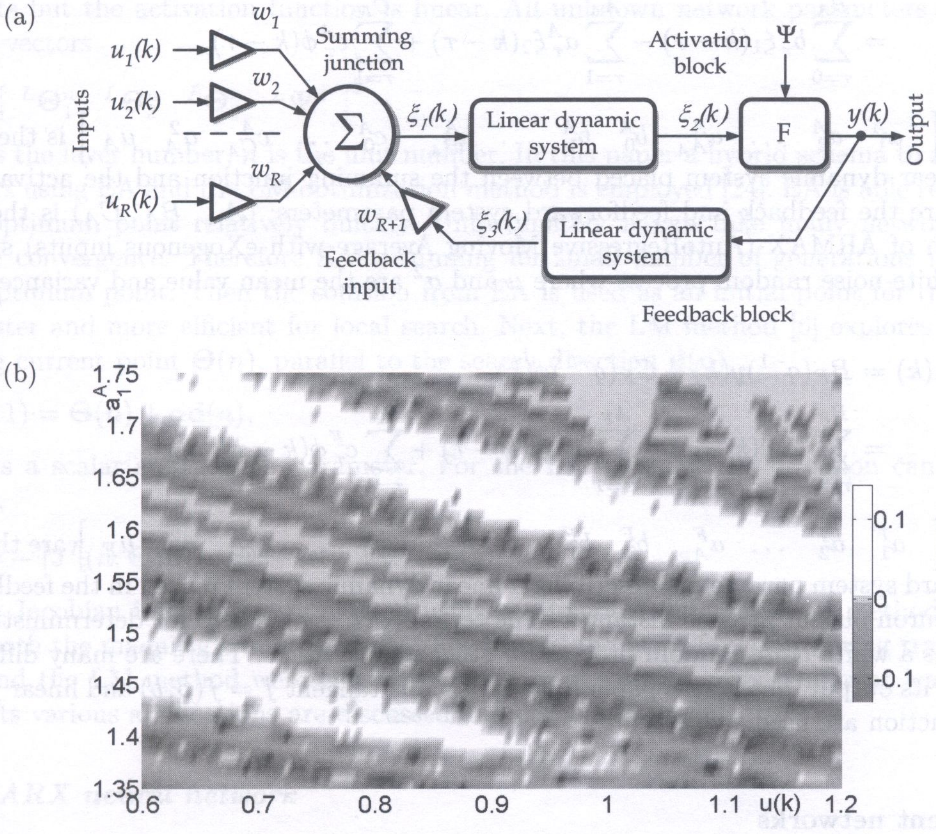


Fig. 2. Dynamic neural unit and its chaotic character behavior; (a) Structure of dynamic neural unit, (b) Lyapunov exponent for $R = 1$, $w_1 = -1$, $w_2 = 1$, $b_0^A = -1$, $b_0^F = -1$, $\beta = 5.85$, $b = 1$

There are also given two methods of input selection: the Gamma test statistic as well as attractor reconstruction using method of delays and mutual information. In Section 3 experimental results obtained for simulated and real-world data are presented. Finally, the paper is briefly concluded in Section 4.

2. Theoretical background

2.1. Dynamic neuron model

Developing dynamic neural units is one of the most common ways to improve the ability of artificial neural networks to model linear, nonlinear and chaotic dynamic systems. In this paper, a dynamic behavior (deterministic and stochastic) is embedded in the neuron by introducing linear dynamic systems into its structure. The general concept is not new [1, 3, 14, 23], but some further developments are introduced [25]. It brings an artificial neuron closer and closer to the biological model.

The behavior of the dynamic neural unit under consideration is described by the following equations,

$$\xi_1(k) = \sum_{i=1}^{R} u_i(k)\, w_i + \xi_3(k)\, w_{R+1}, \tag{1}$$

where: $\mathbf{\Theta}_1 = \begin{bmatrix} w_1 & w_2 & \dots & w_{R+1} \end{bmatrix}$ is the vector with external and feedback input weights w_i; $\mathbf{u}(k)$ is the neuron input vector of the dimension $[R \times 1]$. Using the time shining operator $q^{-1}f(k) = f(k-1)$ the neuron internal state can be formulated as follows,

$$A_A(q^{-1})\xi_2(k) = B_A(q^{-1})\xi_1(k) + C_A(q^{-1})\phi(k), \tag{2}$$

$$\xi_2(k) = \sum_{\tau=0}^{B_A} b_\tau^A \xi_1(k-\tau) - \sum_{\tau=1}^{A_A} a_\tau^A \xi_2(k-\tau) + \sum_{\tau=1}^{C_A} c_\tau^A \phi(k-\tau), \tag{3}$$

where $\Theta_2 = \begin{bmatrix} a_1^A & a_2^A & \dots & a_{A_A}^A & b_0^A & b_1^A & \dots & b_{B_A}^A & c_0^A & \dots & c_{C_A}^A & \sigma_A^2 & \mu_A \end{bmatrix}$ is the vector describing a linear dynamic system placed between the summing junction and the activation block; a_τ^A, b_τ^A, c_τ^A are the feedback and feedforward system parameters; (A_A, B_A, C_A) is the structure representation of ARMAX (AutoRegressive Moving Average with eXogenous inputs) system, ϕ – vector of a white-noise random process where μ and σ^2 are the mean value and variance of a white noise and

$$A_F(q^{-1})\xi_3(k) = B_F(q^{-1})y(k) + C_F(q^{-1})\phi(k), \tag{4}$$

$$\xi_3(k) = \sum_{\tau=0}^{B_F} b_\tau^F y(k-\tau) - \sum_{\tau=1}^{A_F} a_\tau^F \xi_3(k-\tau) + \sum_{\tau=1}^{C_F} c_\tau^F \phi(k-\tau), \tag{5}$$

where $\Theta_3 = \begin{bmatrix} a_1^F & a_2^F & \dots & a_{A_F}^F & b_0^F & b_1^F & \dots & b_{B_F}^F & c_0^F & \dots & c_{C_F}^F & \sigma_F^2 & \mu_F \end{bmatrix}$ are the feedback and feedforward system parameters describing a linear dynamic system placed in the feedback block; $y(k)$ is the neuron output at time instant k. The term $c_\tau \phi(k-\tau)$ is zero for deterministic dynamic systems and is a white-noise random process for stochastic systems. There are many different ways of calculating its output, but in this paper only hyperbolic tangent $f = f(\beta, b)$ and linear $f = f(a, b)$ activation function are used.

2.2. Recurrent networks

Recurrent neural networks, in general, can be classified into two categories: locally recurrent networks and globally recurrent ones [14, 32]. The first category encompasses structures similar to static feed-forward topologies but include dynamic neuron models. The second one encompasses neural structures with feedback connections between simple static neurons of different layers or these of the same layer.

2.2.1. Locally recurrent neural network

Locally recurrent neural networks are a class of parametric, nonlinear dynamic models that have found wide-spread use in fault-tolerant control systems, including identification of dynamic systems and modeling problems [14, 15]. The topology of a locally recurrent neural network (described in Fig. 3) which has been used in this part of the author's research consists of two or three layers (general structure was chosen based on information in the literature).

The first layer has simple static neurons with a nonlinear activation function (in the case when there are only two layers then the first one is called hidden and has dynamic units). Hidden layer includes dynamic neurons with a nonlinear activation function. The last layer consists of simple

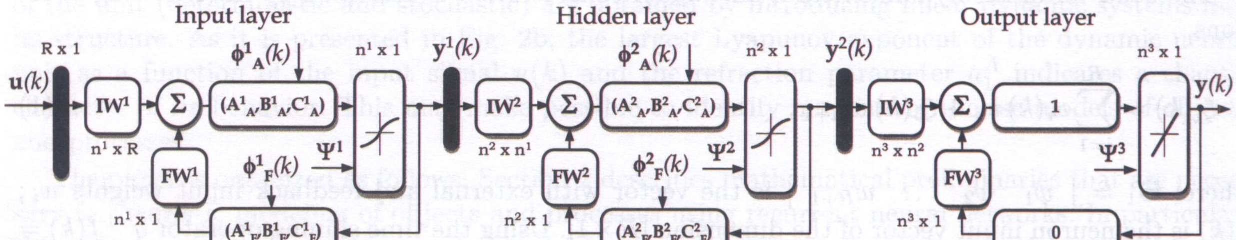


Fig. 3. Locally recurrent globally feed-forward neural network [25]

static units but the activation function is linear. All unknown network parameters may be represented by vectors

$$^{L}\mathbf{\Theta} = \left[\ ^{L}\mathbf{\Theta}_1^n \quad ^{L}\mathbf{\Theta}_2^n \quad ^{L}\mathbf{\Theta}_3^n \quad ^{L}\mathbf{\Psi}^n\ \right]^T \tag{6}$$

where L is the layer number, n is the unit number. In this paper a hybrid schema to adjust network parameters using EA and the LM optimization method is employed [24]. EA is able reach the region near an optimum point relatively quickly. Unfortunately it can take many network simulations to achieve convergence. Therefore EA is running for small number of generations in order to get near an optimum point. Then the solution from EA is used as an initial point for the LM method that is faster and more efficient for local search. Next, the LM method [6] explores along the line containing current point $\mathbf{\Theta}(n)$, parallel to the search direction $\mathbf{d}(n)$,

$$\mathbf{\Theta}(n+1) = \mathbf{\Theta}(n) + \alpha\mathbf{d}(n), \tag{7}$$

where α is a scalar step length parameter. For the LM method, the direction can be calculated as follows,

$$\mathbf{d}(n) = -\left[\mathbf{J}^T(n;\mathbf{\Theta})\,\mathbf{J}(n;\mathbf{\Theta}) + \lambda\mathbf{I}\right]^{-1} \nabla f(n;\mathbf{\Theta}), \tag{8}$$

where the Jacobian information is derived using a numerical differentiation method, and scalar λ controls both the magnitude and direction. The fitness and error function during training stage for the EA and the LM method were chosen as the sum of square errors. Different learning schemes and also its various applications are discussed and compared in [25].

2.2.2. NARX neural network

For comparison purposes, an identifier using a globally recurrent neural network has been designed. This method was chosen as a reference since it is one of the most efficient nonlinear identification technique [20, 33]. NARX (Nonlinear AutoRegressive models with eXogenous inputs) neural network models are a well–known subclass of recurrent networks and have been used in many applications.

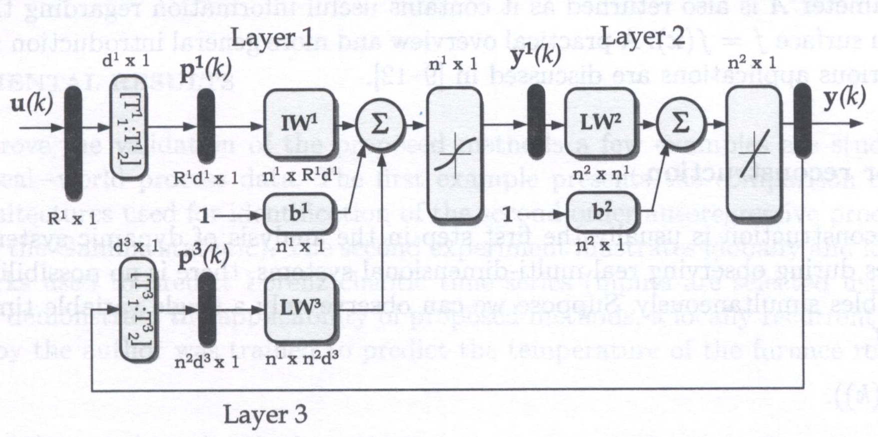


Fig. 4. General structure of NARX neural network [17]

In this paper, the author uses Bayesian regularization backpropagation method for training such networks. A much more detailed description is omitted here but can be found in [17, 20].

2.3. The Gamma test

The Gamma test [29] is a non-linear data-analysis technique that estimates the second moment of the noise distribution in an input/output dataset directly from the data itself. Generally speaking, the Gamma test acts on the hypothesis that if two points $\mathbf{x}$ and $\mathbf{x}'$ are close to each other in input space then their corresponding outputs y and y' should be also close in output space. If they are not close together then we consider that this difference is caused by noise.

In the standard Gamma test we consider a dataset of the MISO form

$$\{(\mathbf{x}_i\,,\, y_i)|1 \leq i \leq M\} \tag{9}$$

where the input vector $\mathbf{x}_i \in \mathbb{R}^{\mathrm{d}}$ is limited to some closed bounded set $C \subset \mathbb{R}^{\mathrm{d}}$. Under additive noise, the relationship between input and output is expressed as follows,

$$y_i = f(\mathbf{x}_i) + r_i\,, \tag{10}$$

where f is a smooth function which is unknown, and r is a random variable that represents noise that may be caused by: measurement errors, not all the relevant factors that influence on the output are included in the inputs, the underlying relationship between input and output is not smooth.

The Gamma test [11, 29] estimates $\mathrm{Var}(r)$ in $O(M \log M)$ time by first constructing a kd-tree using input vectors $\mathbf{x}_i|(1 \leq i \leq M)$ and then using the kd-tree to construct lists of the k-th $|(1 \leq k \leq p)$ nearest neighbors $\mathbf{x}'_{i,k}|(1 \leq i \leq M)$ of $\mathbf{x}_i$. Here p is small, fixed and bounded, typically $p \approx 10$. The algorithm next computes

$$\delta_M(k) = \frac{1}{M}\sum_{i=1}^{M}\|\mathbf{x}'_{i,k} - \mathbf{x}_i\|^2 \tag{11}$$

where $\|\cdot\|$ denotes Manhattan/Euclidean/Infinity norm, and

$$\gamma_M(k) = \frac{1}{2M}\sum_{i=1}^{M}\|y'_{i,k} - y_i\|^2 \tag{12}$$

where $y_{i,k}$ is the output value associated with $\mathbf{x}_{i,k}$. Finally the regression line $\gamma_M(k) = \Gamma + A\delta_M(k)$ is computed and the vertical intercept Γ (Gamma statistic) returned as the estimate for $\mathrm{Var}(r)$. The slope parameter A is also returned as it contains useful information regarding the complexity of the unknown surface $f = f(\mathbf{x})$. A practical overview and more general introduction to the method and also its various applications are discussed in [9–12].

2.4. Attractor reconstruction

Phase-space reconstruction is usually the first step in the analysis of dynamic systems. In a large number of cases during observing real multi-dimensional systems, there is no possibility to measure all of the variables simultaneously. Suppose we can observe only a single variable time series $g(k)$, $k = 1, 2, \ldots, M$,

$$g(k) = G(\mathbf{x}(k)). \tag{13}$$

At this time, if we want to analyze the original dynamic system described by state-space trajectory $\mathbf{x}(k)$, we should reconstruct the attractor [21] in the multivariate phase-space using the scalar time series $g(k)$. The most common approach to attractor reconstruction is the method of delays for the reason that is the most straightforward and the noise level is constant for each delay component (more details may be found in [26, 27, 31]). The method converts the time series $g(k)$ into vectors $\mathbf{y}(k)$ using time delay Δ,

$$\mathbf{y}(k) = \begin{bmatrix} g(k) & g(k+\Delta) & \cdots & g(k+(D{-}1)\Delta) \end{bmatrix}^T, \tag{14}$$

where D is the embedding dimension (according to Takens's theorem $D \geq 2n + 1$), Δ is the reconstruction delay or lag, n is the topological dimension.

The quality of the reconstruction strongly depends upon the delay parameter. A number of useful strategies that can be employed for selecting Δ are discussed in [27]. In this paper mutual information is applied. Mutual information is a measure of the nonlinear dependence between random variables (in contrast to the linear dependence measured by autocorrelation). Formally, the mutual information of two discrete random variables X and Y can be defined as

$$I(X;Y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}, \tag{15}$$

where $p(x,y)$ is the joint probability distribution function of X and Y, and $p(x)$ and $p(y)$ are the marginal probability distribution functions of X and Y, respectively. By making the assignment $[X;Y] = [g(k); g(k+\Delta)]$ we can measure how the values of $g(k+\Delta)$ are dependent on the values of $g(k)$,

$$I(\tau) = I(g(k), g(k+\Delta)). \tag{16}$$

Delay coordinates of $\mathbf{y}(k)$ are interpreted as relatively independent when the mutual information is small. Consequently, the optimal delay corresponds to the first minimum of the function $I(\Delta)$.

2.5. Quality measures

The quality coefficients of neural network models for testing signals under consideration are given below,

$$MAPE = \frac{100}{n} \cdot \sum_{n=1}^{N} \left| \frac{y(n) - \hat{y}(n)}{y_{\max} - y_{\min}} \right|, \qquad U = \sqrt{\frac{\sum_{n=1}^{N} [y(n) - \hat{y}(n)]^2}{\sum_{n=1}^{N} [y(n) - y(n - \Delta n)]^2}}, \tag{17}$$

where N is the number of patterns; $\hat{y}(k)$ is the network output. These measures may be applied for different tasks (e.g. Mean Absolute Percentage Error ($MAPE$) and Thiel's statistic (U) are meaningful for control systems [3, p. 556]).

3. EXPERIMENTAL RESULTS

In order to prove the validation of the proposed methods a few examples are studied using simulation and real–world process data. The first example presents the comparison of two different recurrent architectures used for identification of the second order autoregressive process (inputs are selected using the Gamma statistic). The second experiment illustrates globally and locally recurrent neural networks used to predict Lorenz chaotic time series (inputs are selected using the method of delays). To demonstrate the applicability of proposed methods, a locally recurrent network which is developed by the author was trained to predict the temperature of the furnace roof.

3.1. Investigations using simulation data

The first example

The second order autoregressive process with a linear correlated noise and the exogenous input signal in the form of x-component of the Lorenz system (the quadratic exponent in the coupling term) is described by the following difference equation,

$$\kappa : y_n - 1.095 y_{n-1} + 0.4 y_{n-2} = 0.6 x_n^2 + 0.1 \xi_n\,, \tag{18}$$

where ξ is white noise ($SNR = 20\,\mathrm{dB}$), x is normalized to standard deviation, $y = y_n$, $u_1 = x_n$. There are also inputs that do not have any influence or have but not physically powerful on

Table 1. Examplary results obtained for two recurrent neural network architectures

	Inputs									Gamma*	Neural models			
	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	$\Gamma(\mathbf{u}, y)$	$3^{\frac{(-,1,-)}{(-,1,-)}} - 1^{\frac{1}{0}}$		$3^{[0:1]}_{[1:2]} - 1$	
$\frac{1}{10}\sum_{i=1}^{10}\Gamma_i$	**0.6**	0.0	0.3	0.1	0.1	0.4	**0.7**	**1**	**1**	–	*MAPE*	*U*	*MAPE*	*U*
Γ_1	1	0	0	0	0	0	1	1	1	0.0036	0.1414	0.0267	0.4032	0.0442
Γ_2	0	0	0	0	0	1	1	1	1	0.0038	0.1552	0.0342	0.1069	0.0101
Γ_3	1	0	0	0	0	0	0	1	1	0.0040	0.0508	0.0238	0.3576	0.0426
Γ_4	1	0	0	0	0	1	1	1	1	0.0045	0.2285	0.0328	0.4283	0.0468
Γ_5	0	0	1	1	0	0	1	1	1	0.0045	0.1456	0.0381	0.4890	0.0500
Γ_6	1	0	0	0	0	1	0	1	1	0.0047	0.0503	0.0206	0.1876	0.0267
Γ_7	1	0	1	0	0	0	1	1	1	0.0052	0.1125	0.0321	0.2824	0.0295
Γ_8	1	0	1	0	1	0	1	1	1	0.0053	0.1179	0.0364	0.4124	0.0441
Γ_9	0	0	0	0	0	1	0	1	1	0.0053	0.0629	0.0227	0.3446	0.0400
Γ_{10}	0	0	0	0	0	0	1	1	1	0.0054	0.0777	0.0451	0.3329	0.0369
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Γ_{399}	0	0	1	1	1	1	0	0	0	0.2429	6.8530	0.5343	4.1918	0.4112
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

* Manhattan norm, $k = 15$

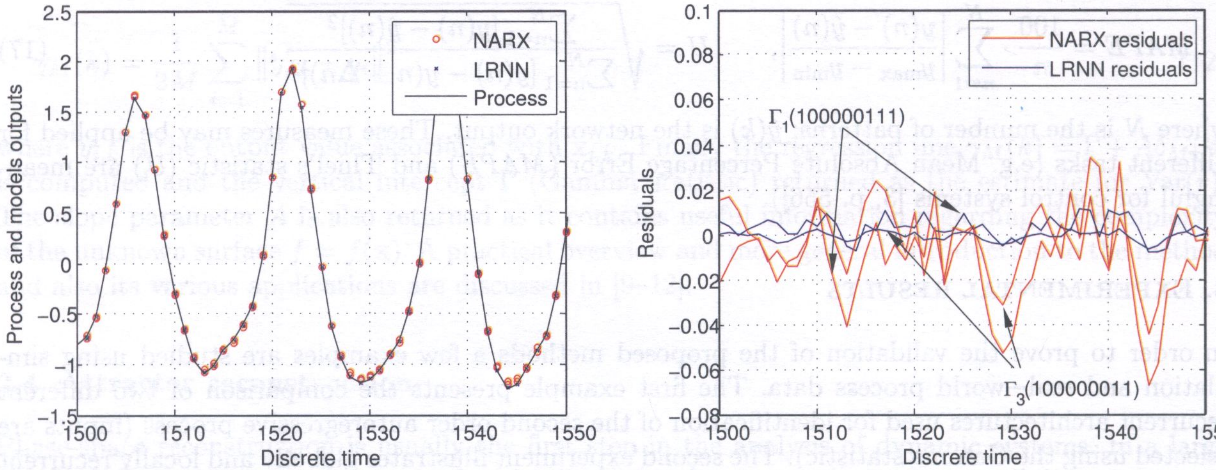


Fig. 5. Testing stage: Process and neural models outputs for inputs u_1, u_7, u_8, u_9 and u_1, u_8, u_9 (NARX — globally recurrent networks, LRNN — locally recurrent networks)

the process output: u_2 is the second orbit of the Lorenz system; u_3 is the third orbit of the Lorenz system; u_4 is a simple straightforward signal; u_5 is a poly-harmonic signal; u_6 is x-component of the Lorenz system to the power of three, u_7 is u_1 plus random process. In addition, u_8 equals y_{n-1} and u_9 equals y_{n-2}. Relevant inputs of neural models were then selected using the coefficient Γ only (Table 1). For this example the Manhattan norm and $k = 15$ were chosen since it gives improved inputs selectivity. It can be easily seen that for nine inputs it was needed to compute the Gamma test 2^9 times. The best subset of input variables is included in the first 10 subsets with smallest $|\Gamma|$.

In the batch mode, 1000 values from the generated data for $k = 1, \ldots, 1000$ were used as the training data set, and the succeeding 1000 values for $k = 1001, \ldots, 2000$ were then applied as the testing data set. A few globally and locally neural structures were examined in order to get sufficient quality of the results. For training of locally recurrent networks the EA-LM schema was used, whereas Bayesian regularization backpropagation method was used in order to train globally neural networks.

In Table 1 two different architectures are compared: $3^{(-,1,-)}_{(-,1,-)} - 1^{1}_{0}$ LRNN and $3^{[0:1]}_{[1:2]} - 1$ GRNN. The forecast errors (for NARX and LRNN) are shown in Fig. 5 as residual lines near the plots of the actual and predicted values. To estimate the quality of the obtained results a mean absolute percentage error and Theil's statistic were used.

The second example

In the following part of the paper a typical chaotic system (the Lorenz system) is chosen to demonstrate the abilities of the dynamic neuroidentifiers described in previous sections. The uncontrolled model is given by

$$\dot{x} = 10(x - y), \tag{19}$$

$$\dot{y} = -y + (28 - z)x, \tag{20}$$

$$\dot{z} = -\tfrac{8}{3}z + xy. \tag{21}$$

To obtain the time series value at integer points, the fourth-order Runge–Kutta method to find the numerical solution to the above Lorenz equations was applied (fixed-step size equals 0.01). The time-series prediction problem is formulated as follow: for known values of time series up to the point in discrete time, shall we say, k to predict the value at some point in the future, shall we say, $k + H$,

$$\langle x\,(k - (D - 1) \cdot \Delta), \ldots, x(k - \Delta), x(k)\rangle \to x(k + H), \tag{22}$$

where according to Takens's theorem $D = 2n + 1 = 7$ and the delay parameter Δ is selected using Mutual Information $H = \Delta = 5$ (the first minimum of the $I(\Delta)$, see Fig. 6a. The first 1000 data values were used for training dynamic neural models while other 800 data samples were applied in the testing stage (for Lorenz orbit x only). For this task several structures of globally and locally recurrent neural networks were examined in order to model chaotic behavior of the Lorenz system.

In Figure 6c there are presented results obtained for $3^{1}_{0} - 2^{(-,2,-)}_{(-,1,-)} - 1^{1}_{0}$ LRNN and $6^{[0:2]}_{[1:2]} - 1$ GRNN. For training of locally and globally neural networks the same algorithms as in the first example were used. It may be seen that MAPE coefficients are quite similar for both neural models. However, in terms of quality of prediction (Theil's statistic) the results for LRNN are better than for globally networks.

3.2. Investigations using technological process data

The database concerning process of copper reduction from a slag has been collected by some industrial system. There are more than 180 static and dynamic attributes collected: parameters of power supply, temperatures in many points of the electric furnace, state parameters such as mass of charge, its chemical analysis determined a few times a cycle. In order to model the temperature of the furnace roof (between the first and second electrode), the locally recurrent neuroidentifier is proposed (locally recurrent networks with dynamic neural processors described in previous sections).

First of all, input/output data subset was selected based on information from staff maintaining this plant and technical documentation dealing with this process. Moreover, relevant inputs are selected using the Gamma statistic. Let us consider that $y = y_n$ is the temperature of the furnace roof, $u_1, \ldots, u_4$ are currents that should not be negligible, $u_5, \ldots, u_7$ are positions of the electrodes, $u_8, \ldots, u_{11}$ are pressures in the selected points of the furnace, u_{12} and u_{13} are water flows of cooling system installed on the furnace, u_{14} equals y_{n-35}, u_{15} equals y_{n-10}, u_{16} equals y_{n-5}. The Gamma test was calculated using the infinity norm and $k = 15$. The first 10 best subsets of input variables in the ascending order of $|\Gamma|$ are presented in Table 2. It was found that the best subset of inputs for neural models would be $(u_1, u_2, u_3, u_4, u_6, u_8, u_{13}, u_{14}, u_{15}, u_{16})$

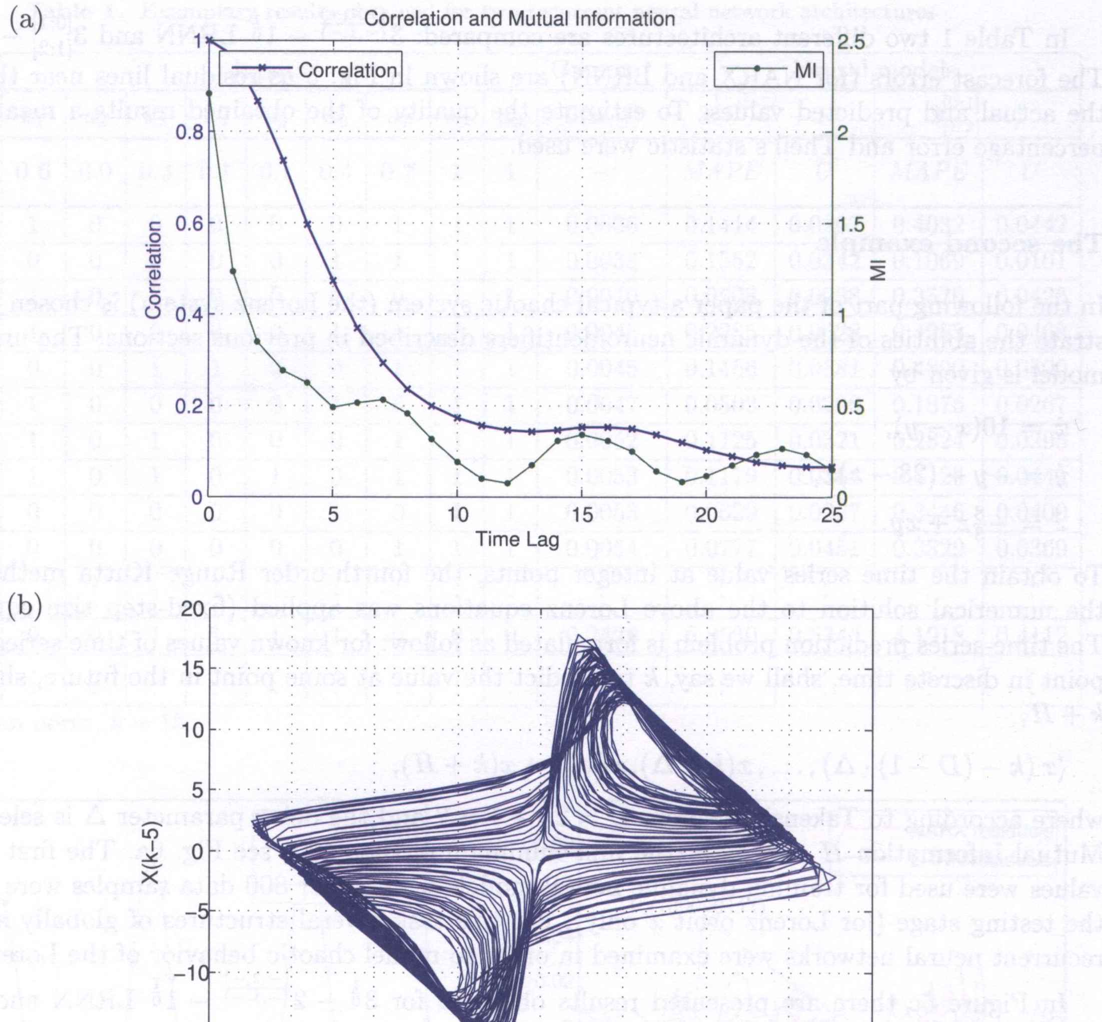


Fig. 6. Heuristic model of x-component of the Lorenz system; (a) Correlation and Mutual Information for x-component of the Lorenz system, (b) Reconstructed attractor in 2-dimension space, (c) The Lorenz time series and neural model outputs for $D = 7$ and $\Delta = 5$ (five-step ahead forecast)

Table 2. The Gamma statistic for input/output data subset

	Inputs												Gamma*
	u_5	u_6	u_7	u_8	u_9	u_{10}	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}	u_{16}	$\Gamma(\mathbf{u}, y)$
$\frac{1}{10}\sum_{i=1}^{10}\Gamma_i$	0.2	**0.7**	0.3	**0.7**	0.2	0	0.4	0.3	**0.9**	**0.6**	**0.6**	**0.9**	–
Γ_1	0	0	1	1	0	0	0	1	1	1	1	1	0.1224
Γ_2	0	1	0	1	0	0	1	0	1	0	1	1	0.3522
Γ_3	0	1	0	1	0	0	0	0	1	1	1	1	0.3809
Γ_4	0	1	0	0	1	0	0	0	0	1	0	1	0.4106
Γ_5	1	0	1	0	0	0	0	1	1	0	0	0	0.4184
Γ_6	1	0	0	1	0	0	0	0	1	1	1	1	0.7119
Γ_7	0	1	0	1	0	0	1	0	1	1	0	1	0.7578
Γ_8	0	1	0	1	1	0	1	0	1	0	1	1	0.8108
Γ_9	0	1	0	1	0	0	1	1	1	0	1	1	0.8311
Γ_{10}	0	1	1	0	0	0	0	0	1	1	0	1	0.8654

* Infinity norm, $k = 15$

(a)

(b)

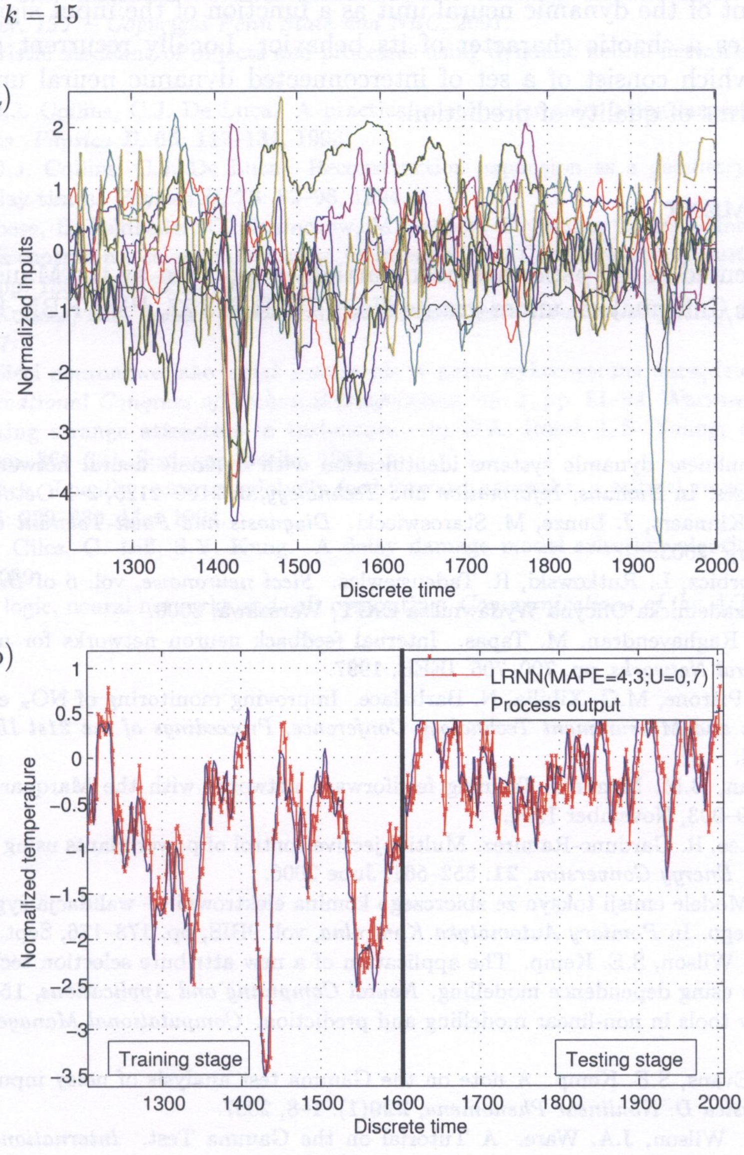


Fig. 7. Heuristic model of the temperature of the furnace roof; (a) Inputs of neural model selected using the Gamma test, (b) Process and neural model outputs

The second step was to build neural models of the temperature for selected inputs. For this experiment only the locally recurrent neural network which is developed by the author was considered. In order to model the temperature of the furnace roof several structures of such networks were examined. In the training stage 1600 samples were used, whereas the testing of the model was carried out using another set of 400 patterns. The EA-LM schema was used to train these models. Selected inputs, the prediction provided by the $3^{(-,2,-)}_{0} - 2^{(-,1,-)}_{(-,1,-)} - 1^{1}_{0}$ LRNN, and prediction errors are shown in Fig. 7. It can be readily seen that the results are very good in terms of quality of prediction. In this way such models may be useful in fault-tolerant control systems of complex plants.

4. Conclusions and future work

Some issues of identification, modeling and forecasting of the chaotic signals and industrial systems by means of locally and globally recurrent neural networks were considered in this paper. Examples of the processing of chaotic time series and the modeling of an industrial plant confirm the efficiency of the considered approach and might be used as a theoretical basis in the future work. The largest Lyapunov exponent of the dynamic neural unit as a function of the input signal and the refraction parameter indicates a chaotic character of its behavior. Locally recurrent globally feed forward neural networks which consist of a set of interconnected dynamic neural units allow to get very good results in terms of quality of prediction.

Acknowledgment

The research presented in the paper has been partially supported by the Ministry of Education and Scientific Research/Information under grants No. 4 T07B 018 27, PBZ-KBN-105/T10/2003.

References

[1] M. Ayoubi. Nonlinear dynamic systems identification with dynamic neural networks for fault diagnosis in technical processes. In *Humans, Information and Technology*, **3**: 2120–2125, 2–5 October 1994.

[2] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki. *Diagnosis and Fault-Tolerant Control.* Springer-Verlag, Berlin–Heidelberg, 2003.

[3] W. Duch, J. Korbicz, L. Rutkowski, R. Tadeusiewicz. *Sieci neuronowe*, vol. 6 of *Biocybernetyka i inżynieria biomedyczna.* Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2000.

[4] K. Gobbak, H. Raghavendran, M. Tapas. Internal feedback neuron networks for modeling of an industrial furnace. In *Neural Networks*, pp. 700–705. IEEE, 1997.

[5] S. Graziani, N. Pitrone, M.G. Xibilia, N. Barbalace. Improving monitoring of NO_x emissions in refineries. In *Instrumentation and Measurement Technology Conference, Proceedings of the 21st IEEE*, vol. 1, pp. 594–597, 18–20 May 2004.

[6] Martin T. Hagan, B.M. Menhaj. Training feedforward networks with the Marquardt algorithm. In *Neural Networks*, **5**: 989–993, November 1994.

[7] J.S. Heo, K.Y. Lee, R. Garduno-Ramirez. Multiobjective control of power plants using particle swarm optimization techniques. *Energy Conversion*, **21**: 552–561, June 2006.

[8] A. Jankowska. Modele emisji toksyn ze zbiorczego komina elektrowni — walidacja sygnałów wejściowych analizatora wirtualnego. In *Pomiary Automatyka Kontrolna*, vol. 9BIS, pp. 173–176, Sept. 2005.

[9] P.S. Jarvis, I.D. Wilson, S.E. Kemp. The application of a new attribute selection technique to the forecasting of housing value using dependence modelling. *Neural Computing and Applications*, **15**(2): 136–153, April 2006.

[10] A.J. Jones. New tools in non-linear modelling and prediction. *Computational Management Science*, **1**(2): 109–149, July 2004.

[11] A.J. Jones, D. Evans, S.E. Kemp. A note on the Gamma test analysis of noisy input/output data and noisy time series. *Physica D: Nonlinear Phenomena*, **229**(1): 1–8, 2007.

[12] S.E. Kemp, I.D. Wilson, J.A. Ware. A Tutorial on the Gamma Test. *International Journal of Simulation Systems, Science and Technology*, **6**(1-2), January 2005.

[13] J. Korbicz. Robust fault detection using analytical and soft computing methods. *Bulletin of the Polish Academy of Sciences: Technical Sciences*, **54**(1): 75–88, 2006.

[14] J. Korbicz, J.M. Kościelny, Z. Kowalczuk, W. Cholewa. *Fault Diagnosis. Models, Artificial Intelligence, Applications*. Springer-Verlag, Berlin–Heidelberg, 2004.

[15] J.M. Kościelny, M. Bartyś. Models in diagnostics of industrial processes. *Diagnostyka*, **30**: 291–296, 2004.

[16] P.G. Luebbers, A.S. Pandya. Neural networks for process control. In *Systems, Man and Cybernetics*, pp. 393–398, Chicago, IL, USA, October 1992.

[17] *MATLAB and Simulink for Technical Computing*. The MathWorks. http://www.mathworks.com/access/helpdesk/help/toolbox/nnet/, 2007.

[18] W. Moczulski. Methodology of heuristic modelling of dynamic objects and processes for diagnostics and control. In *Recent Developments in Artificial Intelligence Methods*, pages 123–126, November 2005.

[19] W. Moczulski. Metodyka heurystycznego modelowania obiektów i procesów dynamicznych w diagnostyce i sterowaniu. In *Pomiary Automatyka Kontrolna*, vol. 9BIS, pp. 39–42, Sept. 2005.

[20] K.S. Narendra, K. Parthasarathy. Identification and control of dynamic systems using neural networks. In *IEEE Transactions on Neural Networks*, pp. 4–27, 1990.

[21] E. Ott. *Chaos w Układach Dynamicznych*. Wydawnictwo Naukowo-Techniczne, Warszawa, 1997.

[22] S.J. Ovaska, A. Kamiya, Y.Q. Chen. Fusion of soft computing and hard computing: computational structures and characteristic features. *Systems, Man and Cybernetics, Part C: Applications and Reviews*, **36**: 439–448, May 2006.

[23] F. Pasemann. A simple chaotic neuron. *Physica D*, **104**: 205–211, May 1997.

[24] R.B.C. Prudêncio, T.B. Ludermir. Neural network hybrid learning: Genetic algorithms and Levenberg-Marquardt. *CiteSeer. IST - Copyright Penn State and NEC*, 2001.

[25] P. Przystałka. Heuristic modeling of objects and processes using dynamic neural networks. *Diagnostyka*, **2/38**: 15–18, 2006.

[26] M.T. Rosenstein, J.J. Collins, C.J. De Luca. A practical method for calculating largest Lyapunov exponents from small data sets. *Physica D*, **65**: 117–134, 1993.

[27] M.T. Rosenstein, J.J. Collins, C.J. De Luca. Reconstruction expansion as a geometry-based framework for choosing proper delay times. *Physica D*, **73**: 82–98, 1994.

[28] M. Schlang, E. Broese, B. Feldkeller, O. Gramckow, M. Jansen, T. Poppe, C. Schaeffner, G. Soergel. Neural networks for process control in steel manufacturing. In *International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 155–158. IEEE, 1997.

[29] A. Stefánsson, N. Končar, A.J. Jones. A note on the Gamma test. *Neural Computing and Applications*, **5**(3): 131–133, Sept. 1997.

[30] R. Tadeusiewicz. Sieci neuronowe jako wciąż jeszcze nie w pełni wykorzystane narzędzie diagnostyki. In *Proceedings of II International Congress of Technical Diagnostics*, vol. 1, pp. 81–94, Warsaw, 2000.

[31] F. Takens. Detecting strange attractors in turbulence. In D.A. Rand, L.S. Young, eds., *Lecture Notes in Mathematics* 898, pp. 366–381. Springer, Berlin, 1981.

[32] A.C. Tsoi, A.D. Back. Locally recurrent globally feed-forward networks: a critical review of architectures. In *Neural Networks*, **5**: 229–239, Mar 1994.

[33] L. Tsungnan, C.L. Giles, G. Bill, S.Y. Kung. A delay damage model selection algorithm for NARX neural networks. IEEE, 1999.

[34] L.A. Zadeh. Fuzzy logic, neural networks and soft computing. *Communications of the ACM*, **37**(3), March 1994.