HEURISTIC MODELING OF OBJECTS AND PROCESSES USING DYNAMIC NEURAL NETWORKS

Piotr PRZYSTALKA

Silesian University of Technology, Department of Fundamentals of Machinery Design
Konarskiego Street 18a, 44-100 Gliwice, Poland, e-mail: pp@polsl.pl

Summary

The methodology of heuristic modeling is one of the subjects included in the activities developed by the Department of Fundamentals of Machinery Design [4, 6]. Among all the approaches of heuristic modeling some of the most common are artificial neural networks. There are many papers and books devoted to applications of neural networks for modeling dynamic systems [1, 2, 4, 5, 6, 7]. In this paper, known approach basing on dynamic neuron model is presented (dynamic neuron with IIR filter in the activation block [2]) but some developments are introduced. Locally recurrent networks which are composed of dynamic neural units described in [2, 5, 7] are able to model behavior of complex dynamic systems. Nevertheless, they have one major disadvantage, that is, neural networks composed of these neurons are not able to represent stochastic behaviors of some objects [4, 6]. By introducing the ARMAX (or ARX) system into dynamic neuron model author has received dynamic neuron unit that never behaves in the same way (it brings an artificial neuron closer and closer to the biological model). In this paper the author presents formal description of dynamic neuron unit with ARMAX system in the feedback block. There are also described a general structure of dynamic neural network composed of these neurons, two known training methods and some commonly used quality measures. At the end of the paper three examples of applications are given.

Keywords: artificial neural networks, locally recurrent neural networks, linear, non-linear and chaotic dynamic systems, quasi-Newton methods, heuristic modeling.

INTRODUCTION

Nowadays there are many industrial plants that carry out complex processes. In general, a dynamic behaviour of them is difficult to be modelled with the use of classical analytical methods [1, 2, 3, 4, 6]. Sometimes it is easier and faster (in some cases it is only one way) to use selected heuristic methods (soft modeling methods) in order to solve various engineering problems in modeling tasks [4]. The
paper describes locally recurrent neural networks applied to model non-linear dynamic objects and processes. Presented networks consist of artificial neurons with linear dynamic system blocks. The general concept is not new [2, 5], but some further developments are introduced. Generally, neural networks which are composed of those neurons [2, 5] are able to represent dynamic behaviour of some systems but they are not able to model their stochastic behaviour themselves [4, 6]. Therefore, there is need to elaborate much more general neuron models which can be used for modeling both dynamic and stochastic systems simultaneously.

In this paper, the author develops dynamic neuron model with the IIR or FIR filter in the activation block by introducing the ARMAX (or ARX) system into its structure. The behaviour of that neuron is not always deterministic (it is described in Section 1).

The paper is composed as follows. In Section 1 only some equations for formal descriptions of dynamic neurons are shown. Next, in Section 2 a few different variants of a dynamic unit are presented. Section 3 describes applications of two known quasi-Newton methods (the BFGS method and the LM method) for training depicted neural networks. At the end of the paper, there are three examples presented. They show modeling of dynamic systems, fault-tolerant application and modeling of an electric furnace.

1. ARTIFICIAL NEURON WITH IIR/FIR AND ARX/ARMAX Dynamic Systems

Developing dynamic neural units is one of the most common ways to improve the ability of artificial neural networks to model linear, non-linear and chaotic dynamic systems.

Dynamic and stochastic behaviour is embedded in the neuron by introducing the IIR or FIR filter and the ARX or ARMAX system into its structure.

\[ \mathbf{\xi}(k) = \sum_{i=0}^{R} d_i \cdot \mathbf{z}_i(k) + \sum_{i=0}^{6} w_i \cdot x_i(k) + e \cdot \mathbf{\phi}(k) \]

where: \( \mathbf{\xi}(k) \) is its order, and \( A, B \) are the feedback and feed-forward filter parameters; \( C=c_0, d_1, \ldots, d_6, c_1, \ldots, c_7 \) are the mean value and variance of a white noise; \( c_1, \ldots, c_6, c_7 \) - system parameters; \( (A,B,C) \) is the structure representation of ARMAX system. The term \( e \cdot \mathbf{\phi}(k) \) is zero for deterministic dynamic systems and is a white-noise random process for stochastic systems.

There are three different ways of calculating its output:

- by using Gaussian activation function:
  \[ \mathbf{\eta}(k) = F(\mathbf{\xi}(k), \mathbf{\theta}) = e^{-\frac{1}{2} (\mathbf{\xi}(k) - \mu)^2} \]

- by using hyperbolic activation function:
  \[ \mathbf{\eta}(k) = F(\mathbf{\xi}(k), \mathbf{\theta}) = \frac{2}{1 + \exp(-\beta \cdot (\mathbf{\xi}(k) + \delta \mu))} + b - 1 \]

- by using linear activation function:
  \[ \mathbf{\eta}(k) = F(\mathbf{\xi}(k), \mathbf{\theta}) = a \cdot \mathbf{\xi}(k) \]

where: \( \mathbf{\theta} = [\sigma \ c \ \beta \ \delta \mu \ b \ a] \) - a vector depending on selected activation function.

2. Dynamic Neural Network

This class of artificial neural networks (with dynamic neuron models) is well-known as locally recurrent globally feed-forward networks [2,5]. As we can read in [2] that is somewhere in between a feed-forward and a globally recurrent architecture. The dynamic unit described in Sec. 1 may be used differently.

Figure 2 shows three units: the first one is a static neuron (for \( B=0, b_0=1, A=1, a_1=0, C=c_0=D=\infty, d_1=E=e_0=0 \)); the second unit is a completely dynamic neuron model with white-noise random process and hyperbolic (tan for short) activation function; the last one represents also
a dynamic neuron model with the finite impulse response filter and there is Gaussian (gauss for short) activation function. The topology of a dynamic neural network (described in Fig. 3) which has been used in this part of the research consists of three layers (a general structure was chosen based on information in the literature). The first layer has simple static neurons with a non-linear activation function. Hidden layer includes dynamic neurons with a non-linear activation function (dynamic neurons iir, armax, tan, fir, gauss, etc. were tested). The last layer consists of simple static units but the activation function is linear.

\[ f(k,\Theta) = \|y(k) - \hat{y}(k,\Theta)\| \]  

should be minimized basing on a given set of input-output patterns. All the line-search methods explore along the line containing current point \( \Theta(k) \), parallel to the search direction \( d(k) \) (Eq. 9).

\[ \Theta(k+1) = \Theta(k) + \alpha \cdot d(k), \]  

where: \( \alpha \) is a scalar step length parameter. In the next part of this Section the two ways of definition of a direction will be discussed.

For the BFGS method the direction is given as follows:

\[ d(k) = -H^{-1}(k;\Theta) \cdot \nabla f(k;\Theta), \]  

where: the gradient information \( \nabla f(k;\Theta) \) is derived by partial derivatives using the numerical differentiation method via finite differences (i.e. like in [3]). To avoid a large amount of computation during calculating \( H \) numerically the formula of BFGS for its approximation is used:

\[ H(k+1) = H(k) + \frac{q(k) \cdot q(k)}{q(k) \cdot s(k)} \cdot H(k) \cdot s(k) \cdot s(k)^T \cdot H(k) - \frac{H(k) \cdot s(k) \cdot s(k)^T \cdot H(k)}{s(k) \cdot H(k) \cdot s(k)}, \]  

where: \( H(k=0) \) can be a set as the identity matrix \( I \).

\[ s(k) = \Theta(k+1) - \Theta(k), \]  

\[ q(k) = \nabla f(k+1;\Theta) - \nabla f(k;\Theta), \]  

On the other hand, for the LM method the direction can be calculated as follows:

\[ d(k) = -[H(k;\Theta) \cdot J(k;\Theta) + \lambda \cdot I]^{-1} \cdot \nabla f(k;\Theta), \]  

where: the Jacobian information is derived using also a numerical differentiation method, and scalar \( \lambda \) controls both the magnitude and direction.

A very important thing is that the term \( e,c,\Phi(k-t) \) at each \( k\)-th algorithm step has to be fixed (for both algorithms).

3.2. Quality measures

The quality coefficients of neural network models under consideration are given below:

- Mean Absolute Percentage Error (MAPE for short):

\[ MAPE = \frac{100}{n} \sum_{n} \left| \frac{y(n) - \hat{y}(n)}{y_{\text{mean}}} \right|, \]

- Thiel’s statistic (U for short):
4. EXPERIMENTAL RESULTS

In the following parts of the paper some examples of applications will be presented. The first and second subsections show the results obtained using models which have been determined in the Matlab environment like Simulink and some other toolboxes. The last example describes the application of discussed neural structures to data gathered from the real process.

4.1. Modeling of dynamic systems

Problem 1

Given input-output pairs \( \{u(k), y(k)\} \) generated by the non-linear system defined by (Narendra K. Parthasarathy K.):

\[
\begin{align*}
\hat{y}(k+1) &= \hat{y}(k) + u(k) + u(k-2) + \hat{y}(k-1) + a(k-1) + a(k-2) \\
&\quad + y(k-2)^2 + a(k)
\end{align*}
\]

identify the topology of the network and its parameters.

For this task several different variants of dynamic and static neural structures were examined. Table 1 presents some results (one can compare different neural structures).

<table>
<thead>
<tr>
<th>Type</th>
<th>No. Iter.</th>
<th>J</th>
<th>MAPE</th>
<th>( U )</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ffbp</td>
<td>200</td>
<td>12e-05</td>
<td>2.76e-01</td>
<td>2.28e-01</td>
</tr>
<tr>
<td>ffbptd</td>
<td>200</td>
<td>8.35e-05</td>
<td>2.28e-01</td>
<td>1.19e-01</td>
</tr>
<tr>
<td>arx2(1,1)</td>
<td>9,12e-05</td>
<td>2.56e-01</td>
<td>2.11e-01</td>
<td></td>
</tr>
<tr>
<td>BFGS method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ffbp</td>
<td>200</td>
<td>32e-04</td>
<td>1.47</td>
<td>1.19</td>
</tr>
<tr>
<td>ffbptd</td>
<td>200</td>
<td>22e-04</td>
<td>9.47e-01</td>
<td>7.86e-01</td>
</tr>
<tr>
<td>arx2(1,1)</td>
<td>200</td>
<td>35e-04</td>
<td>7.21e-01</td>
<td>8.1e-01</td>
</tr>
</tbody>
</table>

There are results for three-layered perceptron ffbp (the input layer has 16 neurons with hyperbolic function; 1 neuron in the output layer), for time delay feedforward network ffbptd with the same architecture as previous one but also with a delay block \((0,1,2)\) in the input layer. The last one is dynamic neural structure \(iir2^{(2)})(arx(1,1))\) (discussed in Sec. 2, three-layered network 5-12-1 with dynamic neurons and hyperbolic activation function in the hidden layer). As we can see locally recurrent network usually requires much less neurons to obtain almost the same quality result as in the case of a simple feed-forward network with static neurons. However, we must remember that dynamic neuron model is described by much more parameters than straightforward neural unit.

In the training stage the input-output patterns were represented as follows:

\[
\{\{P(k), T(k)\}\}_{k=1,...,N}
\]

and \(u(k)\) was initialized with random values (1000 samples with a uniform distribution in the range \(<-2, 1,2>\)). In the testing stage three different input signals were used: polyharmonic signal, hard-limit signal and pseudo-random binary sequence (in Table 1 there are errors for the first signal only).

4.2. Fault-tolerant application

Problem 2

Design a virtual sensor \(\hat{L}_2\) that could serve as a plain fault detection and identification block (based on the non-linear analytical redundancy technique).

![Fig. 5. The three-tank system](image)

The three-tank system model as shown in figure 5 is written using well-known mass balance equations. The system can be represented like in [2] by:

\[
\begin{align*}
\dot{q}_i &= q_i(t) + q_i(t) - q_j(t) - q_k(t) \\
\dot{q}_j &= q_j(t) + q_i(t) - q_j(t) - q_k(t) \\
\dot{q}_k &= q_k(t) - q_j(t) - q_k(t)
\end{align*}
\]

where: \(q_i\) represents the water flow rate from tank \(i\) to \(j\) which is given by Torricelli’s rule:
\[
q_i(t) = \mu \cdot S_i \cdot \text{sign}[L_i(t) - L_i(t) + \Delta L_i(t)],
\]
(19)

where: \(\Delta L_i\) represents fault of the measuring channel \(i\), \(q_i\) is undesirable leakage from the tank \(i\).

The measured signals are: streams of the medium \(q_1, q_2\) that flow into the first and second tank; control signals \(U_1, U_2\); levels in tanks \(L_1, L_2, L_3\).

\[\hat{L}_2\]

Fig. 6. Virtual sensor for fault detection and identification block

In order to create a virtual sensor the three-layered dynamic neural network was used (7 neurons with hyperbolic transfer function in the input layer, 4 hidden neurons with the \(iir^{(1)}arx^{(1.0)}\) blocks and hyperbolic function, one output neuron with linear activation function, Fig. 6). The training and testing patterns were obtained by simulating model (Eq. 18) in the Matlab environment. The error (MAPE) was less than 5\%, therefore it was possible to use the model in the fault detection and control modules what is shown in Figures 8 and 9.

\[y_{\text{Cu}}(k) = f(\hat{L}(k), L(k), \hat{I}(k), \hat{I}(k), p(k), y_{\text{Cu}}(k)).\]
(20)

where: \(I(k)\) is a vector of three currents (for electrodes \(R, S, T\)), \(L(k)\) is a vector that determines that positions of three electrodes; \(p(k)\) is a position of a claw of the transformer; \(y_{\text{Cu}}(k)\) is a current state parameter describing the copper concentration in slag; \(\Delta k\) is the time horizon of prediction (for this example it equals 5, 10, 15 or 20 min.). \(f\) represents a non-linear input-output relation of the neural network; \(\hat{I}(k), \hat{L}(k)\) are suitably preprocessing currents and electrodes positions (detailed description was omitted in the paper).

4.3. Modeling of industrial furnace

Problem 3

The fault of the measuring channel no. 2 is detected by comparing value from the sensor \(L_2\) with estimated value \(\hat{L}_2\) obtains from the virtual sensor. When absolute difference between these values is greater than 10\% of scale range then control system change its rules (Fig. 9).

Fig. 7. Virtual and real sensor outputs (results for testing stage: MAPE<5\%)

Fig. 8. Fault in the measuring channel 2 (without analytical redundancy)

The database concerning process of copper reduction from slag has been collected by SCADA system. There are more than 180 attributes collected: parameters of power supply, temperatures in many points of the furnace, and state parameters such as mass of charge, and its chemical analysis determined three times a cycle. In order to create a submodel of the analyzed process a non-linear discrete difference equation is proposed (Eq. 20, in the first stage of research inputs and output were selected based on information from staff maintaining this object and technical documentation dealing with this process):
(6 neurons with hyperbolic activation function in the input layer; 5 hidden neurons with the \( f_t^{(2)} \) filter and the \( ax^{(1,0)} \) system and hyperbolic function; one output neuron with linear transfer function).

As we can observe, all the results (MAPE = 1%) for each example are quite similar to each other. However, in the case of the \( f_t^{(2)}ax^{(1,0)} \) network Thiel’s coefficients \((U)\) are lesser than in the \( ffbd \) network. Consequently, these models may be useful in fault detection systems or even in control systems.

5. SUMMARY

This work demonstrates theoretical and practical aspects of heuristic modeling based on locally recurrent neural networks. As one can see it is possible to introduce into artificial neuron structure some modules which represent its dynamic and stochastic behaviour (it brings an artificial neuron closer and closer to the biological model). Moreover, some known local optimization algorithms may be used in order to train dynamic network networks. Nevertheless, these algorithms are still too slow for the reason that the gradient information is computed using a numerical differentiation method.

6. TO DO

Many problems are still open to solution, such as the automatic identification of dynamic neural topology. However, the most important tasks are to find the gradient information using an analytical method and to apply some global optimization algorithms (e.g. genetic algorithms or recursive random search) for preliminary adjusting network parameters.

ACKNOWLEDGMENTS

Paper presents a selected part of the research supervised by Prof. W. Moczulski. This research has been partially supported by the Ministry of Education and Scientific Research under grant No. 4 T07B 018 27.

REFERENCES