TRAINING OF THE RECURRENT NEURAL NETWORKS FOR PREDICTION

Jury Savitsky and Vladimir Golovko

Computers Department, Brest Polytechnic Institute,
Moskovskaja st. 267, Brest, Republic of Belarus
E-Mail: cm@brpi.belpak.brest.by
Fax: (+375162)422127

Abstract. In this paper the technique of creation of effective methods of training recurrent neural network for prediction problems are discussed. The various functions of activation of neural units are considered. The adaptive algorithms of training of neural networks with varied functions of activation of neural elements are considered. The computing experiments on prediction of time series demonstrate possibilities of the developed methods.

Keywords: recurrent neutral network, prediction.

Introduction

One of fundamental properties of neural networks is their ability after training to integration and prolongation of results. It creates the objective preconditions for creation on the basis of their intelligent neural systems for prediction of a various sort of processes. An important problem neural technology of forecasting is choice of neural network architecture, enabling adequately to describe predict process and to execute the successful prediction. Thus basic is the question of choice of a type neural elements in architecture predicting neural system, from which depends the probability of successful formation of optimum predicting function during neural network training. Other problem, worth at system engineering of a similar sort, is presence of effective training algorithms, having global convergence, enabling to reduce time of training and to increase accuracy of neural networks training [1]. In majority of cases the parameters of training algorithms determine efficiency of application neural technologies in practice.

In this work the problems of choice of types of neural elements in the architecture of the predicting neural network are considered. The adaptive algorithm of training for neural network with varied functions of activation is developed. The efficiency of the used approaches in the real tasks is parsed.

Neural Network Architecture

As basic networks architecture was accepted fully connected third-layered recurrent neural network containing one hidden layer of nonlinear units and a single output linear unit, as shown in a figure 1. The output activity of the neural network is defined by expression:

$$Y^p(t) = \sum_{i=1}^{N_h} w_{i0} h_i^p(t) - s_0 \qquad (1)$$

where N_h - number of units of the hidden level, $h_i^p(t)$ - output activity of hidden units, s_0 - threshold for output unit, w_{i0} - weights from hidden input units i to the output unit.

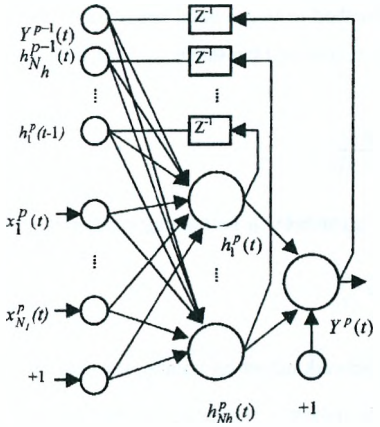

Fig.1. The recurrent neural network architecture

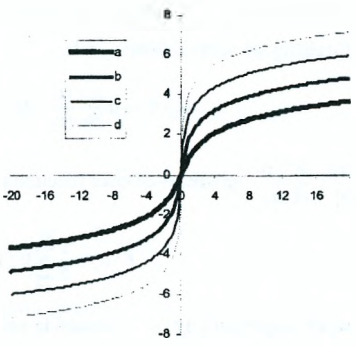

Fig.2. The logarithmic activation function for various parameters a: a) $a = 1.0$, b) $a = 0.1$, c) $a = 0.01$, d) $a = 0.001$

The output activity of the hidden units on the current training iteration t for training exemplar p is defined as:

$$h_j^p(t) = g\left(\sum_{i=1}^{N_I} w_{ij} x_i^p(t) + \sum_{k=1}^{N_h} w_{kj} h_k^{p-1}(t) + w_{0j} Y^{p-1}(t) + s_j\right) \qquad (2)$$

where $x_i^p(t)$ is a i'th element of the input vector $x^p(t)$, $p = 1 \div P$, P - size of the training set, N_I - size of an input vector, w_{ij} - weights from external units i to hidden units j, w_{kj} - weights from hidden units k to hidden units j, $h_k^{p-1}(t)$ - output activity of a hidden unit k for the previous training vector $p-1$, w_{0j} - weight to the hidden units from an output unit, $Y^{p-1}(t)$ - network output activity for the previous training vector $p-1$ and s_j - thresholds of the hidden units. In this work we use two

types for hidden units transfer function. At one case is used non-standard logarithmic function $g(x) = \ln\left((x + \sqrt{x^2 + a})/\sqrt{a}\right)$, $(a > 0)$. The choice by this transfer function is stipulated by that it is unlimited on all define area. It allows better to simulate and predict complex non-stationary processes. The parameter a defines declination of the activation function (see figure 2). At other case in hidden units is used sigmoid transfer function, defined as $g(x) = \left(1 + e^{-x}\right)^{-1}$.

Training

The most popular training algorithm for multilevel perceptrons and recurrent networks is back propagation. This algorithm is based on gradient descent method in neural units weights area and consists of fulfilment of an iterative procedure of updating weights and thresholds for each training exemplar of training set under following rule:

$$\Delta w_{ij}(t) = -\alpha \frac{\partial E^p(t)}{\partial w_{ij}(t)}, \ \Delta s_j(t) = -\alpha \frac{\partial E^p(t)}{\partial s_j(t)} \tag{3}$$

where $\frac{\partial E^p(t)}{\partial w_{ij}(t)}, \frac{\partial E^p(t)}{\partial s_j(t)}$ - gradients of error function for training exemplar p for training iteration t,

$$E^p(t) = \frac{1}{2}\sum_{j=1}^{N_o}\left(Y_j^p(t) - D_j^p\right)^2 \tag{4}$$

$Y_j^p(t)$ - network output activity, D_j^p - desirable value of a network output for training exemplar p, N_o - number of the output units. During training there is the reduction process of the total network error:

$$E(t) = \sum_{p=1}^{P} E^p(t) \tag{5}$$

For improvement of network training parameters and removal defects of classical back propagation algorithm, connected with empirical selection of a constant training step, use the steepest descent method for calculation of an adaptive training step, according to it:

$$\begin{cases} \Delta w_{ij}(t) = -\alpha^p(t)\dfrac{\partial E^p(t)}{\partial w_{ij}(t)}, \\ \Delta s_j(t) = -\alpha^p(t)\dfrac{\partial E^p(t)}{\partial s_j(t)}, \\ \alpha^p(t) = \min\left\{E^p\left(w_{ij}(t+1), s_j(t+1)\right)\right. \end{cases} \tag{6}$$

where $\alpha^p(t)$ - step value, adapted on each training iteration t for each external vector p.

According to expression (6) the formulas for calculation of adaptive step for sigmoid, logarithmic and linear functions of activation were obtained.

For linear transfer function the adaptive training step is defined by expression:

$$\alpha^p(t) = \frac{1}{\sum_{i=1}^{N_I} (h_i^p(t))^2 + 1} \tag{7}$$

where $h_i^p(t)$ - elements of input activity of the linear unit at the time t for the training vector p.

For logarithmic activation function the estimate of an adaptive training step can be received by the following expression:

$$\hat{\alpha}^p(t) = \frac{\sqrt{a}\sum_{j=1}^{N_h} (\gamma_j^p(t))^2 \left(\sqrt{(B_j^p(t))^2 + a}\right)^{-1}}{\left(1 + \sum_{i=1}^{N_I} (x_i^p(t))^2 + \sum_{k=1}^{N_h} (h_k^{p-1}(t))^2 + (Y^{p-1}(t))^2\right) \cdot \left(\sum_{j=1}^{N_h} (\gamma_j^p(t))^2 \left((x_j^p(t))^2 + a\right)^{-1}\right)} \tag{8}$$

where $B_j^{p(t)} = \sum_{i=1}^{N_I} w_{ij} x_i^p(t) + \sum_{k=1}^{N_h} w_{kj} h_k^{p-1}(t) + w_{0j} Y^{p-1}(t) + s_j$ is weighed sum of inputs of the hidden units j, γ_j^p - error of the unit j for training exemplar p:

$$\gamma_i^p(t) = \sum_{j=1}^{N_h} \gamma_0^p(t) w_{i0} \frac{1}{\sqrt{1 + (B_j^p(t))^2}}, \quad \gamma_0^p(t) = Y_0^p(t) - D_o^p \tag{9}$$

For standard sigmoid transfer function adaptive training step is computed as:

$$\hat{\alpha}^p(t) = \frac{4\sum_{j=1}^{N_h} (\gamma_j^p(t))^2 h_j^p(t)(1 - h_j^p(t))}{\left(1 + \sum_{i=1}^{N_I} (x_i^p(t))^2 + \sum_{k=1}^{N_h} (h_k^{p-1}(t))^2 + (Y^{p-1}(t))^2\right) \cdot \left(\sum_{j=1}^{N_h} (\gamma_j^p(t))^2 (h_j^p(t))^2 (1 - h_j^p(t))\right)^2} \tag{10}$$

Here γ_j^p - error of the unit j for sigmoid transfer function for training exemplar p:

$$\gamma_i^p(t) = \sum_{j=1}^{N_h} \gamma_0^p(t) w_{i0} h_j^p(t)(1 - h_j^p(t)), \quad \gamma_0^p(t) = Y_0^p(t) - D_o^p \tag{11}$$

Experiments

For simulation were used the time series of passenger airtransportations described in [2]. It size is 144 units. For training were used 110 units, taken with coefficient 1E-4. The training was carried out the method of the sliding window. For simulation two types of neural networks were used. One of them contained the sigmoid activation function of the hidden units. In other network the

logarithmic function of activation of the hidden units with the parameter $a = 0.01$ was used. Both networks consist 20 input units, 10 hidden units, of 1 output unit and were trained up to an equal error $E = 1.76E-5$. The prediction was carried out on 34 steps forwards. For an estimation of the prediction results is used the mean square predict error computed as:

$$E_{pr}(L) = \frac{1}{L}\sum_{l=1}^{L}\left(\hat{Y}(l) - x(l)\right)^2 \qquad (12)$$

where $\hat{Y}(l)$ - predict value for the step l, $x(l)$ - actual value of time series in the moment l, L - total of prediction steps. The training both neural networks was characterized by high accuracy, stability and speed. The outcomes of training and prediction are reduced in table 1.

	Sigmoid network architecture	Logarithmic network architecture
Inputs	20	20
Hidden units	10	10
Output	1	1
Training size	110	110
Training iterations	3500	2700
Total training error	1.76E-5	1.76E-5
Predicting steps	34	34
Mean square predict error	8.381E-6	2.769E-5

Table 1. Training and prediction experiments

Conclusion

Described in this paper of training methods have allowed considerably to improve parameters of training recurrence neural networks with various functions of activation of units. For calculation of adaptive step the methods of local optimizations permitting to minimize an error of training for each current measurement standard were used. In this work the advantage of the logarithmic function of activation of the hidden units to construction of architectures of predicting systems was shown.

This work executed with program INTAS 97-0606 "Development of an intelligent sensing instrumentation structure".

References

1. Vladimir A. Golovko, Jury V. Savitski, Vitaly B. Gladischuk. Predicting Neural Net //Proc. of the CMNDT-95, Minsk, Belarus, 1995, pp.348-353.

2. Box G.E.P., Genkins G. M. Time-Series Analysis, Forecasting and Control. New York, 1970.