Interval type-2 fuzzy weight adjustment for backpropagation neural networks with application in time series prediction

Fernando Gaxiola, Patricia Melin, Fevrier Valdez, Oscar Castillo

Tijuana Institute of Technology, Tijuana, Mexico

Abstract

In this paper a new backpropagation learning method enhanced with type-2 fuzzy logic is presented. Simulation results and a comparative study among monolithic neural networks, neural network with type-1 fuzzy weights and neural network with type-2 fuzzy weights are presented to illustrate the advantages of the proposed method. In this work, type-2 fuzzy inference systems are used to obtain the type-2 fuzzy weights by applying a different size of the footprint of uncertainty (FOU). The proposed approach is based on recent methods that handle adaptation of weights using fuzzy logic of type-1 and type-2. The proposed approach is applied to a case of prediction for the Mackey-Glass time series (for \( \tau = 17 \)). Noise was applied in different levels to the test data of the Mackey-Glass time series for showing that the type-2 fuzzy backpropagation approach obtains better behavior and tolerance to noise than the other methods.

© 2013 Elsevier Inc. All rights reserved.

1. Introduction

In this paper we propose an adaptation of weights in the backpropagation algorithm for the neural network using type-2 and type-1 fuzzy inference systems. This proposed approach is different than other ones in the literature, where the adaptation is in the momentum and adaptive rate [3,14,27,33], or with triangular or trapezoidal fuzzy numbers for the weights [19,20], and also because the proposed method works with type-2 fuzzy weights, which is the main difference with respect to the others methods.

The proposed approach is applied to time series prediction for the Mackey-Glass series. In this case, the objective is obtaining the minimum prediction error for the time series data.

This paper is focused on the comparison of the traditional monolithic neural network with respect to the neural networks with type-1 and type-2 fuzzy weights. The same architecture and learning algorithm for the three neural models are used. Noise in the real test data to analyze the performance of the models is also applied.

This work is based on comparing the performance for the neural network with type-1 fuzzy weights and type-2 fuzzy weights, with the traditional approach of using real numbers for the weights, which is important because the weights affect the performance of the learning process of the neural network and therefore obtaining better results.

This conclusion is based on the application of neural networks of this type, where previous works have shown that the training of neural networks for the same problem initialized with different weights or its adjustment in a different way, but at the end is possible to reach a similar result.

* Corresponding author. Tel.: +52 6646236318.

E-mail addresses: forgaor_29@hotmail.com (F. Gaxiola), pmelin@tectijuana.mx (P. Melin), fevrier@tectijuana.edu.mx (F. Valdez), ocastillo@tectijuana.mx (O. Castillo).

0020-0255/$ – see front matter © 2013 Elsevier Inc. All rights reserved.

http://dx.doi.org/10.1016/j.ins.2013.11.006
The contribution of the paper is the proposed method for type-2 fuzzy weight adjustment in backpropagation learning of neural networks for providing them the ability to manage uncertainty in real data. The main idea of the proposed method is that enhancing the backpropagation method with type-2 fuzzy logic enables better management of uncertainty in the learning process and with this improved results can be achieved.

Simulation results show that neural networks with the proposed type-2 fuzzy approach have the ability of outperforming their type-1 and type-0 counterparts.

The proposed method is different from the adaptive neuro-fuzzy inference system (ANFIS) method, because ANFIS uses the neural network architecture for obtaining the characteristics of the fuzzy systems, and performs the operations based on the calculations of the fuzzy systems; this is different to the proposed method that uses the type-2 fuzzy systems to update the weights used in the neural network for the training (learning) process.

The next section presents a background of different weight management strategies and modifications of the backpropagation algorithm in neural networks. Section 3 explains the proposed method and the problem description. Section 4 describes the monolithic neural network architecture and neural network architecture with type-1 fuzzy weights. Section 5 describes the neural network with type-2 fuzzy weights proposed in this paper. Section 6 presents the simulation results for the proposed methods. Finally, in Section 7, conclusions are presented.

2. Background and basic concepts

2.1. Overview of related works

The most useful basic training method in the area of neural networks is the backpropagation algorithm and its variations [9,16,24,32]. When these methods are applied in practical problems, the training time of the basic backpropagation algorithm can be very high [11,35]. In the literature several methods have been proposed to accelerate the convergence of the backpropagation algorithm [3,14,27,33]. There exists many works about adjustment or managing of weights, but only the most important and relevant for this paper will be considered here [5,11,12,38].

Gedeon [15], using a discrete selection (following the Hebbian paradigm) performs the weight adjustment. Monirul and Murase [28], as a strategy used the same weights in epochs where the output does not change. Meltser et al. [26], performed a weight adjustment of the network through the BFGS Quasi-Newton method (Broyden–Fletcher–Goldfarn–Shanno). Barbounis and Theocharis [2], performed the weights updating using the identification of recursive error prediction (RPE). Yeung et al. [39], used a new training objective function to adjust the weights for a network with radial basis functions.

Kamarthi and Pittner [22] used extrapolation for obtaining a weight prediction of the network at a future epoch. Neville and Eldridge [30], worked with sigma-pi networks, which are transformed for performing a second task of assignation for which they were initially trained. Casasent and Natarajan [5] used weights with complex values and the nonlinear square law. Yam and Chow [38] developed an algorithm to find the initial optimal weights of feedforward neural networks based on the Cauchy inequality and a linear algebraic method. Draghici [12], calculates a range of weights for a category of given problems and ensures that the network has the capacity to solve the given problems with integer weights in that range.

Ishibuchi et al. [19,20], proposed a fuzzy network where the weights are given as trapezoidal fuzzy numbers or triangular fuzzy numbers. Feuring [13] developed a learning algorithm in which the backpropagation algorithm is used to compute the new lower and upper limits of weights. Castro et al. [7], use interval type-2 fuzzy neurons for the antecedents and interval of type-1 fuzzy neurons for the consequents of the rules to propose type-2 neuro-fuzzy models.

There are also recent works on type-2 fuzzy logic that have been developed in time series prediction, like that of Castro et al. [8], and other researchers [1,23,34].

3. Proposed method and problem description

The proposed approach in this paper has the goal of generalizing the backpropagation algorithm using type-1 fuzzy sets and type-2 fuzzy sets to allow the neural network to handle data with uncertainty. In the type-2 fuzzy sets, it will be necessary vary the footprint of uncertainty (FOU) of the membership functions using an optimization method to make it automatically or vary it manually for the corresponding applications [31,37,40,41].

The process of obtaining the weights in the connections for each neuron is performed differently to the traditional adjustment of weights with the backpropagation algorithm (Fig. 1).

The proposed method works with type-1 and type-2 fuzzy weights, considering the possible modification in the way we work internally in the neuron and the adjustment of the weights given in this way (Figs. 2 and 3) [4,28].

We considered the modification of the current methods for adjusting weights that allow convergence to the correct weights for the problem. We developed a method for adjusting weights to achieve the desired result, searching for the optimal way to work with type-2 fuzzy weights [10,21].

To define the activation function \( f(\cdot) \) to use, the linear and sigmoidal functions were tested, because these functions have been used in similar approaches.
4. The monolithic neural network architecture and neural network architecture with type-1 fuzzy weights

The proposed monolithic neural network architecture (see Fig. 4) is described as follows: we used a monolithic neural network with the test data of the Mackey-Glass time series for the input layer, 30 neurons in the hidden layer and 1 neuron in the output layer.

The proposed neural network architecture with type-1 fuzzy weights (see Fig. 5) is described as follows:

Layer 0: inputs.

\[ x = [x_1, x_2, \ldots, x_n]. \]

Layer 1: type-1 fuzzy weights for the hidden layer.

\[ w = \frac{\sum_{i=1}^{n} (f_i w_i)}{\sum_{i=1}^{n} (f_i)}. \]

Layer 2: hidden neuron with type-1 fuzzy weights.

Fig. 1. Scheme of current management of numerical weights (type-0) for the inputs of each neuron.

Fig. 2. Scheme of the proposed management of type-1 fuzzy weights for the inputs of each neuron.

Fig. 3. Scheme of the proposed management of type-2 fuzzy weights for the inputs of each neuron.
Net = \sum_{i=1}^{n} x_i w_i, \quad (3)

Layer 3: output neuron with type-1 fuzzy weights.

Out = \sum_{i=1}^{n} y_i w_i, \quad (4)

Layer 4: obtain a single output of the neural network.

The adaptation of type-1 fuzzy weights will be based on the backpropagation algorithm, as follows:

Step 1: initialize the type-1 fuzzy weights of the network with small random values.

Step 2: present an input pattern and specify the desired output that the network should generate.

Step 3: calculate the current output of the network. First, introduce the network inputs and calculate the outputs corresponding to each layer until the output layer, and this is the network output.

Step 4: calculate the error terms for all neurons. For a neuron “k” of the output layer, calculate delta (\delta_{pk}^O), as follows:

\[ \delta_{pk}^O = (d_{pk} - y_{pk}) f_O^O(\text{Out}). \quad (5) \]

For a neuron “j” of the hidden layer, calculate the delta (\delta_{pj}^h), as follows:

\[ \delta_{pj}^h = f_h^i(\text{Net}) \sum_k \delta_{pk}^O w_{kj}. \quad (6) \]

Step 5: for updating the type-1 fuzzy weights using a recursive algorithm, starting from the output neurons and working back up until the input layer, adjusting the type-1 fuzzy weights as follows:

The update uses two type-1 fuzzy inference systems with 2 inputs (Current type-1 fuzzy weight (w_{kj}) and Change of type-1 fuzzy weight (\Delta w_{kj}(t+1))) and 1 output (Resulting type-1 fuzzy weights (w_{kj}(t+1))):

The change of type-1 fuzzy weights is performed with the following equations: for the neurons of the output layers:
\[ \Delta w_{ij}(t + 1) = \delta_{pi}^O y_{pi}. \]  
(7)

For the neurons of the hidden layers:
\[ \Delta w_{ji}(t + 1) = \delta_{pj}^H x_{pi}. \]  
(8)

**Step 6:** the process is repeated until the error terms are sufficiently small for each of the learned patterns.
\[ E_p = \frac{1}{2} \sum_{k=1}^{M} \delta_{pk}^2. \]  
(9)

We used 2 similar type-1 fuzzy systems to obtain the type-1 fuzzy weights in the hidden and output layer for the neural network.

The first type-1 fuzzy system consists of two inputs: the current weight in the actual epoch and the change of the weight for the next epoch, and one output: the new weight for the next epoch (see Fig. 6).

The input of the current weight has two triangular membership functions with range of \(-1\) to \(1\). The input of change of the weight has two triangular membership functions with range of \(-0.003\) to \(0.003\). The output of the new weight has two triangular membership functions with range of \(-1\) to \(1\) (see Fig. 7).

We used six rules for the type-1 fuzzy inference system of the hidden layer, corresponding to the four combinations of the two triangular membership functions and we added two rules for when the change of weight is null (see Fig. 8).

The second type-1 fuzzy system consists of two inputs: the current weight in the actual epoch and the change of the weight for the next epoch (see Fig. 9).

The input of the current weight has two triangular membership functions with range of \(-1\) to \(1\). The input of change of the weight has two triangular membership functions with range of \(-0.01\) to \(0.01\). The output of the new weight has two triangular membership functions with range of \(-1\) to \(1\) (see Fig. 10).

We used six rules for the type-1 fuzzy inference system for the output layer, corresponding to the four combinations of two membership functions and we added two rules for the case when the change of weight is null (see Fig. 11).

**5. The neural network architecture with type-2 fuzzy weights**

The proposed neural network architecture with type-2 fuzzy weights (see Fig. 12) is described as follows:

**Layer 0:** inputs.
\[ x = [x_1, x_2, \ldots, x_n]. \]  
(10)

**Layer 1:** interval type-2 fuzzy weights for the hidden layer [7].
\[ \tilde{w} = [\bar{w}, \underline{w}]. \]  
(11)

where:
\[ \bar{w} = \frac{\sum_{k=L}^{L} f^k \cdot \bar{w}^k + \sum_{k=L}^{M} f^k \cdot \bar{w}^k}{\sum_{k=L}^{L} f^k + \sum_{k=L}^{M} f^k}. \]  
(12)

\[ \underline{w} = \frac{\sum_{k=R}^{R} f^k \cdot \bar{w}^k + \sum_{k=R}^{M} f^k \cdot \bar{w}^k}{\sum_{k=R}^{R} f^k + \sum_{k=R}^{M} f^k}. \]  
(13)

where \( L \) and \( R \) are the switch points [7].

---

**Fig. 6.** Type-1 fuzzy inference system used in the hidden layer for the neural network with type-1 fuzzy weights.
**Fig. 7.** Inputs (A and B) and output (C) of the type-1 fuzzy inference system used in the hidden layer for the neural network with type-1 fuzzy weights.

**Fig. 8.** Rules of the type-1 fuzzy inference system used in the hidden layer for the neural network with type-1 fuzzy weights.

1. (Current_Weight is lower) and (Change_Weight is lower) then (New_Weight is lower)
2. (Current_Weight is lower) and (Change_Weight is upper) then (New_Weight is lower)
3. (Current_Weight is upper) and (Change_Weight is lower) then (New_Weight is upper)
4. (Current_Weight is upper) and (Change_Weight is upper) then (New_Weight is upper)
5. (Current_Weight is lower) then (New_Weight is lower)
6. (Current_Weight is upper) then (New_Weight is upper)

**Fig. 9.** Type-1 fuzzy inference system used in the output layer for the neural network with type-1 fuzzy weights.
**Fig. 10.** Inputs (A and B) and output (C) of the type-1 fuzzy inference system used in the output layer for the neural network with type-1 fuzzy weights.

**Fig. 11.** Rules of the type-1 fuzzy inference system used in the output layer for the neural network with type-1 fuzzy weights.

**Fig. 12.** Neural network architecture with type-2 fuzzy weights.
Layer 2: hidden neurons with interval type-2 fuzzy weights.

\[ Net = \sum_{i=1}^{n} x_i \tilde{w}_i. \]  

(14)

Layer 3: output neurons with interval type-2 fuzzy weights.

\[ Out = \sum_{i=1}^{n} y_i \tilde{w}_i. \]  

(15)

Layer 4: obtain a single output of the neural network.
The adaptation of type-2 fuzzy weights is based on the backpropagation algorithm in the same way for the type-1 fuzzy weights.

Fig. 13. Type-2 fuzzy inference system used in the hidden layer for the neural network with type-2 fuzzy weights.

Fig. 14. Inputs (A and B) and output (C) of the type-2 fuzzy inference system used in the hidden layer for the neural network with type-2 fuzzy weights.
We used 2 similar type-2 fuzzy inference systems to obtain the type-2 fuzzy weights in the hidden and output layer for the neural network.

For obtaining the type-2 fuzzy inference systems an extension of the membership functions of the type-1 fuzzy inference systems was applied.

We increase or decrease the values for the triangular membership functions with a variable epsilon in terms of percentage to obtain the footprint of uncertainty (FOU). These are applied in the type-2 fuzzy inference systems used in the neural network with type-2 fuzzy weights [18].

We present, for example, the membership functions obtained with an epsilon of ±1% for the inputs and output the type-2 fuzzy inference systems. The first type-2 fuzzy system consists of two inputs: the current weight in the actual epoch and the change of the weight for the next epoch, and one output: the new weight for the next epoch (see Fig. 13).

The input of the current weight has two triangular membership functions with range of $-1$ to $1$. The input of change of the weight has two triangular membership functions with range of $-0.003$ to $0.003$. The output of the new weight has two triangular membership functions with range of $-1$ to $1$ (see Fig. 14).

We used six rules for the type-2 fuzzy inference system of the hidden layer, the four combinations of the two triangular membership functions and we added two rules for when the change of weight is null (see Fig. 15).

The second type-2 fuzzy system consists of two inputs: the current weight in the actual epoch and the change of the weight for the next epoch, and one output: the new weight for the next epoch (see Fig. 16).

The input of the current weight has two triangular membership functions with range of $-1$ to $1$. The input of change of the weight has two triangular membership functions with range of $-0.01$ to $0.01$. The output of the new weight has two triangular membership functions with range of $-1$ to $1$ (see Fig. 17).

We used six rules for the type-2 fuzzy inference system for the output layer, corresponding to the four combinations of two membership functions and we added two rules for the case when the change of weight is null (see Fig. 18).

The experiments were performed in time-series prediction, specifically for the Mackey-Glass time series (for $\tau = 17$) that behaves chaotically and it is a benchmark used in many studies.

We used the gradient descent backpropagation algorithm with adaptive learning rate for the experiments.

The neural networks handle type-1 and type-2 fuzzy weights in each one of their hidden layers and output layer [29,36]. In each hidden layer and output of each network we are using a type-1 fuzzy inference system or type-2 fuzzy inference system to obtain new weights in each epoch of the network [6,17,25].

6. Simulation results

The obtained results for the experiments with the monolithic neural network are shown in Table 1 and Fig. 19, and all parameters of the neural network are established empirically. The best prediction error is of 0.055, and the average error is of 0.077.

We are presenting 10 experiments in Table 1, but the average error was calculated considering 40 experiments with the same parameters and conditions.

<table>
<thead>
<tr>
<th>Rules for Type-2 Fuzzy Weights Hidden Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. (Current_WEIGHT is lower) and (Change_WEIGHT is lower) then (New_WEIGHT is lower)</td>
</tr>
<tr>
<td>2. (Current_WEIGHT is lower) and (Change_WEIGHT is upper) then (New_WEIGHT is upper)</td>
</tr>
<tr>
<td>3. (Current_WEIGHT is upper) and (Change_WEIGHT is lower) then (New_WEIGHT is lower)</td>
</tr>
<tr>
<td>4. (Current_WEIGHT is upper) and (Change_WEIGHT is upper) then (New_WEIGHT is upper)</td>
</tr>
<tr>
<td>5. (Current_WEIGHT is lower) then (New_WEIGHT is lower)</td>
</tr>
<tr>
<td>6. (Current_WEIGHT is upper) then (New_WEIGHT is upper)</td>
</tr>
</tbody>
</table>

Fig. 15. Rules of the type-2 fuzzy inference systems used in the hidden layer for the neural network with type-2 fuzzy weights.

Fig. 16. Type-2 fuzzy inference system used in the output layer for the neural network with type-2 fuzzy weights.
Fig. 17. Inputs (A and B) and output (C) of the type-1 fuzzy inference system used in the output layer for the neural network with type-1 fuzzy weights.

Fig. 18. Rules of the type-2 fuzzy inference systems used in the output layer for the neural network with type-2 fuzzy weights.

Table 1
Results for the monolithic neural network in time series Mackey-Glass.

<table>
<thead>
<tr>
<th>No.</th>
<th>Epoch</th>
<th>Network error</th>
<th>Time (s)</th>
<th>Prediction error</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.091</td>
</tr>
<tr>
<td>E2</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.088</td>
</tr>
<tr>
<td>E3</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.055</td>
</tr>
<tr>
<td>E4</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.083</td>
</tr>
<tr>
<td>E5</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>05</td>
<td>0.080</td>
</tr>
<tr>
<td>E6</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>05</td>
<td>0.077</td>
</tr>
<tr>
<td>E7</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.094</td>
</tr>
<tr>
<td>E8</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.071</td>
</tr>
<tr>
<td>E9</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.055</td>
</tr>
<tr>
<td>E10</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.066</td>
</tr>
</tbody>
</table>
The obtained results for the experiments with the neural network with type-1 fuzzy weights are shown on Table 2 and Fig. 20, in this case all parameters of the neural network and type-1 fuzzy inference systems are established empirically. The best prediction error is of 0.055, and the average error is of 0.094.

![Mackey-Glass Chaotic Time Series](image)

**Table 2**

Results for the neural network with type-1 fuzzy weights in Mackey-Glass time series.

<table>
<thead>
<tr>
<th>No.</th>
<th>Epoch</th>
<th>Network error</th>
<th>Time (s)</th>
<th>Prediction error</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>11</td>
<td>0.068</td>
</tr>
<tr>
<td>E2</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>11</td>
<td>0.097</td>
</tr>
<tr>
<td>E3</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>12</td>
<td>0.071</td>
</tr>
<tr>
<td>E4</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>12</td>
<td>0.100</td>
</tr>
<tr>
<td>E5</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>12</td>
<td>0.055</td>
</tr>
<tr>
<td>E6</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>12</td>
<td>0.102</td>
</tr>
<tr>
<td>E7</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>11</td>
<td>0.093</td>
</tr>
<tr>
<td>E8</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>11</td>
<td>0.108</td>
</tr>
<tr>
<td>E9</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>12</td>
<td>0.086</td>
</tr>
<tr>
<td>E10</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>11</td>
<td>0.106</td>
</tr>
</tbody>
</table>

![Mackey-Glass Chaotic Time Series](image)

**Fig. 19.** Plot of the real data against the prediction data of the Mackey-Glass time series for the monolithic neural network.

**Fig. 20.** Plot of real data against prediction data of the Mackey-Glass time series for the neural network with type-1 fuzzy weights.
The obtained results for the experiments with the neural network with type-2 fuzzy weights are shown on Table 3 and Fig. 21, in that all parameters of the neural network and type-2 fuzzy inference systems are established empirically.

We are also presenting results with different type-2 fuzzy inference systems; for this we vary the values of the triangular membership functions using a variable epsilon in different percentages. The best prediction error is of 0.039 (epsilon = ±80%), and the best average error is of 0.061 (epsilon = ±80%).

In Table 4 we are presenting the comparison among the monolithic neural network (MNN), the neural network with type-1 fuzzy weights (NNT1FW) and the best performance of the neural network with type-2 fuzzy weights (Eps = 80%) (NNT2FW), which shows that the use of type-2 fuzzy weights has the best behavior.

We also performed an experiment applying noise in the range (0.1–1) in the test data to observe the behavior of the monolithic neural network, neural network with type-1 fuzzy weights and neural network with type-2 fuzzy weights. The obtained results for the experiments are shown on Table 5.

In Table 5, the row labeled MNN represents the results of the monolithic neural network, the row with Eps = 0% represents the results of the neural network with type-1 fuzzy weights and the rows with Eps = 1% to Eps = 90% represent...

---

Table 3

<table>
<thead>
<tr>
<th>No.</th>
<th>Eps = 1%</th>
<th>Eps = 5%</th>
<th>Eps = 10%</th>
<th>Eps = 20%</th>
<th>Eps = 30%</th>
<th>Eps = 40%</th>
<th>Eps = 50%</th>
<th>Eps = 60%</th>
<th>Eps = 70%</th>
<th>Eps = 80%</th>
<th>Eps = 90%</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>0.085</td>
<td>0.086</td>
<td>0.065</td>
<td>0.063</td>
<td>0.060</td>
<td>0.068</td>
<td>0.081</td>
<td>0.076</td>
<td>0.075</td>
<td>0.063</td>
<td>0.110</td>
</tr>
<tr>
<td>E2</td>
<td>0.087</td>
<td>0.087</td>
<td>0.081</td>
<td>0.080</td>
<td>0.061</td>
<td>0.049</td>
<td>0.054</td>
<td>0.081</td>
<td>0.049</td>
<td>0.044</td>
<td>0.064</td>
</tr>
<tr>
<td>E3</td>
<td>0.072</td>
<td>0.077</td>
<td>0.090</td>
<td>0.060</td>
<td>0.064</td>
<td>0.050</td>
<td>0.077</td>
<td>0.083</td>
<td>0.044</td>
<td>0.058</td>
<td>0.084</td>
</tr>
<tr>
<td>E4</td>
<td>0.069</td>
<td>0.073</td>
<td>0.088</td>
<td>0.077</td>
<td>0.042</td>
<td>0.077</td>
<td>0.058</td>
<td>0.079</td>
<td>0.057</td>
<td>0.072</td>
<td>0.097</td>
</tr>
<tr>
<td>E5</td>
<td>0.054</td>
<td>0.060</td>
<td>0.059</td>
<td>0.049</td>
<td>0.058</td>
<td>0.053</td>
<td>0.079</td>
<td>0.058</td>
<td>0.073</td>
<td>0.047</td>
<td>0.115</td>
</tr>
<tr>
<td>E6</td>
<td>0.082</td>
<td>0.080</td>
<td>0.078</td>
<td>0.073</td>
<td>0.080</td>
<td>0.083</td>
<td>0.065</td>
<td>0.063</td>
<td>0.048</td>
<td>0.067</td>
<td>0.074</td>
</tr>
<tr>
<td>E7</td>
<td>0.045</td>
<td>0.072</td>
<td>0.086</td>
<td>0.067</td>
<td>0.072</td>
<td>0.063</td>
<td>0.069</td>
<td>0.050</td>
<td>0.078</td>
<td>0.049</td>
<td>0.123</td>
</tr>
<tr>
<td>E8</td>
<td>0.076</td>
<td>0.075</td>
<td>0.055</td>
<td>0.071</td>
<td>0.046</td>
<td>0.058</td>
<td>0.049</td>
<td>0.065</td>
<td>0.058</td>
<td>0.039</td>
<td>0.066</td>
</tr>
<tr>
<td>E9</td>
<td>0.079</td>
<td>0.069</td>
<td>0.075</td>
<td>0.069</td>
<td>0.083</td>
<td>0.082</td>
<td>0.072</td>
<td>0.053</td>
<td>0.069</td>
<td>0.070</td>
<td>0.090</td>
</tr>
<tr>
<td>E10</td>
<td>0.063</td>
<td>0.083</td>
<td>0.069</td>
<td>0.084</td>
<td>0.067</td>
<td>0.074</td>
<td>0.067</td>
<td>0.071</td>
<td>0.063</td>
<td>0.053</td>
<td>0.129</td>
</tr>
<tr>
<td>Average</td>
<td>0.085</td>
<td>0.077</td>
<td>0.078</td>
<td>0.076</td>
<td>0.071</td>
<td>0.072</td>
<td>0.072</td>
<td>0.073</td>
<td>0.068</td>
<td>0.061</td>
<td>0.098</td>
</tr>
</tbody>
</table>

Table 4

<table>
<thead>
<tr>
<th>No.</th>
<th>Epoch</th>
<th>Network error</th>
<th>Time (s)</th>
<th>Prediction error</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNN</td>
<td>800</td>
<td>$1 \times 10^{-7}$</td>
<td>06</td>
<td>0.053</td>
</tr>
<tr>
<td>NNT1FW</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>12</td>
<td>0.055</td>
</tr>
<tr>
<td>NNT2FW</td>
<td>100</td>
<td>$1 \times 10^{-8}$</td>
<td>44</td>
<td>0.039</td>
</tr>
</tbody>
</table>

Fig. 21. Plot of real data against prediction data of the Mackey-Glass time series for the neural network with type-2 fuzzy weights (epsilon = ±80%).
the results of the neural network with type-2 fuzzy weights for different levels of uncertainty. The best performance was achieved with the neural network with type-2 fuzzy weights (Eps = 50%) with noise in the data test.

7. Conclusions

In the experiments, we observe that the neural network with type-2 fuzzy weights obtains better results than the monolithic neural network and the neural network with type-1 fuzzy weights for the Mackey-Glass time series. This conclusion is based on the prediction errors of 0.039, 0.053 and 0.055 respectively, and average errors (40 experiments) of 0.061, 0.077 and 0.094 respectively.

The neural network with type-2 fuzzy weights shows better behavior at different levels of uncertainty than the monolithic neural network and neural network with type-1 fuzzy weights, this conclusion is based on the fact that best values of Table 5 are with type-2 fuzzy weights. The results obtained for the prediction error obtained are smaller than the results of monolithic neural network and neural network with type-1 fuzzy weights, 0.053 and 0.055 respectively.

Applying different levels of noise we observe that the neural network with type-2 fuzzy weights has better behavior and tolerance to noise than monolithic neural network and neural network with type-1 fuzzy weights. This conclusion was found by observing that the type-2 fuzzy weights present lower prediction errors than the other methods; only the type-2 fuzzy weights with epsilon equal to 30% and 90% present worse error predictions than the others methods.

The results obtained in these experiments show that the neural network with type-2 fuzzy weights obtained better results, without noise and with noise, than the monolithic neural network and the neural network with type-1 fuzzy weights, in the prediction for the Mackey-Glass time series.

The proposed methods, neural network with type-1 and type-2 fuzzy weights, have more robustness and achieve better results than the monolithic neural network. Besides, the type-2 fuzzy weights provide the neural network less susceptibility to the occurrence of a significant increase in the prediction error when noise is applied to the real data.

References