An Ensemble Neural Network Architecture with Fuzzy Response Integration for Complex Time Series Prediction

Martha Pulido, Alejandra Mancilla, and Patricia Melin

Tijuana Institute of Technology, Tijuana, México
marthapulido_84@hotmail.com, alejandra.mancilla@gmail.com, epmelin@hafsamx.org

Abstract. In this paper we describe the application of an architecture for an ensemble neural network for Complex Time Series Prediction. The times series we are considering are: the Mackey-Glass, Dow Jones and Mexican Stock Exchange and we show the results of a set of trainings with the ensemble neural network, and its integration with the methods of average, weighted average and Fuzzy Integration. Simulation results show very good prediction of the ensemble neural network with fuzzy logic integration.

1 Introduction

Time series predictions are very important because we can analyze past events to know the possible behavior of futures events and thus can take preventive or corrective decisions to help avoid unwanted circumstances.

The choice and implementation of an appropriate method of prediction has always been a major issue for enterprises that seek to ensure the profitability and survival of business. The predictions give the company the ability to make decisions in the medium and long term, and due to the accuracy or inaccuracy of data could mean predicted growth or profits and financial losses.

It is very important for companies to know the behavior that will be the future development of their business, and thus be able to make decisions that can improve the company's activities, and avoid unwanted situations which in some cases can lead to the company's failure.

2 Time Series and Prediction

A time-series is defined as a sequence of observations on a set of values that takes a variable (quantitative) at different points in time. The time series are widely used today because organizations need to know the future behavior of certain phenomena in order to plan, prevent, and so on, their actions. That is, to predict what will happen with a variable in the future from the behavior of that variable in the

O. Castillo et al. (Eds.): Evolutionary Design of Intelligent Systems, SCI 257, pp. 85–110.
springerlink.com

past [1]. The data can behave in different ways over time, this may be a trend, which is the component that represents a long-term growth or decline in value over a period of time high. You can also have a cycle, which refers to the wave motion that occurs around the trend, or may not have a defined or random manner; there are seasonal variations (annual, biannual, etc.). , which is a behavior pattern that is repeated year after year at a particular time. [2].

The word "prediction" comes from the Latin prognosticum, which means I know in advance. Prediction is to issue a statement about what is likely to happen in the future, based on analysis and considerations of experiments. Making a forecast is to obtain knowledge about uncertain events that are important in decision-making [3]. Time series prediction tries to predict the future based on past data, it take a series of real data $x_t - n, \ldots x_t - 2, x_t - 1, x_t$ and then obtains the prediction of data $x_t + 1, x_t + 2 \ldots x_t + n$. The goal of time series prediction or a model is to observe the series of real data, so that future data may be accurately predicted. [4]

3 Neural Networks

Neural networks are composed of many elements (Artificial Neurons), grouped into layers and are highly interconnected (with the synapses), this structure has several inputs and outputs, which are trained to react (or give values) in a way you want to input stimuli (R values). These systems emulate in some way, the human brain. Neural networks are required to learn to behave (Learning) and someone should be responsible for the teaching or training (Training), based on prior knowledge of the environment problem [5].

Artificial neural networks are inspired by the architecture of the biological nervous system, which consists of a large number of relatively simple neurons that work in parallel to facilitate rapid decision-making [6].

A neural network is a system of parallel processors connected as a directed graph. Schematically each processing element (neuron) of the network is represented as a node. These connections establish a hierarchical structure that is trying to emulate the physiology of the brain as it looks for new ways of processing to solve real world problems. What is important in developing the techniques of NN is if its useful to learn behavior, recognize and apply relationships between objects and plots of real-world objects themselves. In this sense, artificial neural networks have been applied to many problems of considerable complexity. Its most important advantage is in solving problems that are too complex for conventional technologies, problems that have no solution or that the algorithm of the solution is very difficult to find [5].

4 Methods of Integration

There exists a diversity of methods of integration or aggregation of information, and we mention some of these methods below:

Integration by average: this method is used in the ensembles of networks. This integration method is the simplest and most straightforward, consists in the sum of the results generated by each module divided by the number of modules, and the disadvantage is that there are cases in which the prognosis is not good.

Integration of Weighted Average: this method is an extension of the integration by average, with the main difference that the weighted average assigns importance weights to each of the modules. These weights are assigned to a particular module based on several factors; the most important is the knowledge product of experience. This integration method belongs to the well known aggregation operators.

Fuzzy logic was proposed for the first time in the mid-sixties at the University of California Berkeley by the brilliant engineer Lotfi A. Zadeh. Who proposed what it's called the principle of incompatibility: "As the complexity of a system increases, our ability to give precise instructions and build on their behavior decreases to the threshold beyond which the accuracy and meaning are mutually exclusive characteristics." Then introduced the concept of a fuzzy set (Fuzzy Set), under which lies the idea that the elements on which to build human thinking are not numbers but linguistic labels. Fuzzy logic can represent the common knowledge that natural of language is mostly qualitative and not necessarily quantitative in a mathematical language by means of fuzzy set theory and function characteristics associated with them. [7].

Fuzzy Set: Let X be a space of objects and x be generic element of X. A classical set$A$, $A \subseteq X$, is defined as a collection of elements or objects $x \in X$, such that each x can either belong or not belong to the set A. By defining a characteristic function for each element x in X, we can represent a classical set A by a set of ordered pairs $(x, 0)$ or $(x, 1)$, which indicates $x \notin A$ or $x \in A$, respectively. Unlike the aforementioned conventional set, a fuzzy set expresses the degree to which an element belongs to a set. Hence the characteristic function of a fuzzy set is allowed to have values between 0 and 1, which denotes the degree of membership of an element in a given set. The basic structure of a fuzzy inference system consists of three conceptual components: a rule base, which contains a selection of fuzzy rules; (or dictionary), which defines the membership functions used in the fuzzy rules; and a reasoning mechanism, which performs the inference produce (usually the fuzzy reasoning). [8].

5 Genetic Algorithms

Genetic algorithms were introduced by the first time by a professor of the University of Michigan named John Holland [9]. A genetic algorithm, is a mathematical highly parallel algorithm that transforms a set of mathematical individual objects with regard to the time using operations based on evolution. The Darwinian laws of reproduction and survival of the fittest can be used, and after having appeared of

natural form a series of genetic operations between(among the object) that stands out the sexual recombination [10,11]. Each of these mathematical objects is in the habit of being a chain of characters (letters or numbers) of fixed length that adjusts to the model of the chains of chromosomes, and one associates to them with a certain mathematical function that reflects the fitness.

6 Problem Statement and Proposed Method

This paper is concerned with the study of a fuzzy integration method that can be applied to ensemble neural networks with applications to complex time series, in addition to developing alternative methods for the integration of ensemble network, such as the average and weighted average. Figure 1 shows the general architecture used in this work.

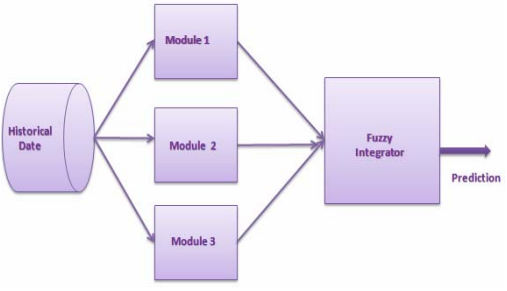

Fig. 1. General Architecture of the ensemble neural network.

Historical data of the Mackey-Glass time series was used for the ensemble neural network trainings, where each module was fed with the same information, to find a suitable architecture for a module of the ensemble will be same or very similar to the other modules, unlike the modular networks, where each module is fed with different data, which leads to architectures that are not uniform. Integration by the average method was very easy to implement, just joining the results of each module and the result was divided by the number of elements, the main problem of this method is that if one of the modules produces an unfortunate result, it can greatly affect the result the integration. Integration by weighted average includes assigning values from 0 to 1, where the module that has the best prediction is the one that will have a greater weight. The network consists of three modules, the allocation of weights we used is 0.50 for the module that produces better results, 0.30 for second best and 0.20 for the worst module, we do this only if the three modules meet the above conditions. Fuzzy integration for the Mackey-Glass time series was implemented in a system of traditional Mamdani fuzzy inference, which consists of three input variables (the results of each module of our ensemble network) and one output variable (the result of integration), is shown in Figure 2.

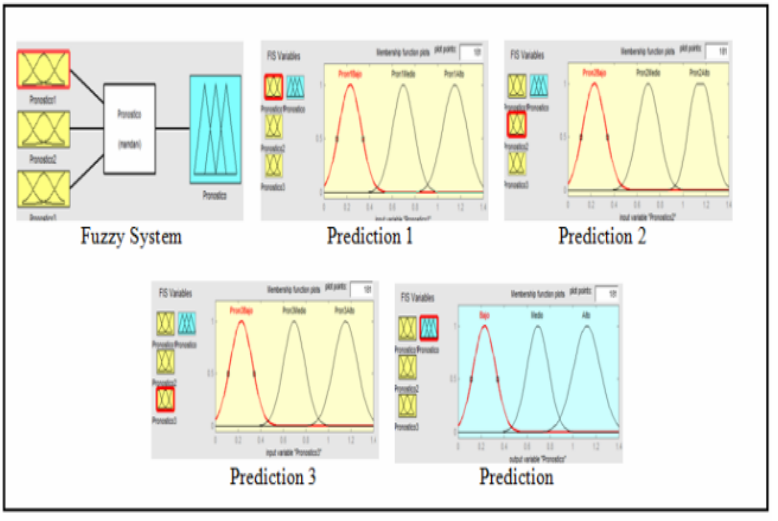

Fig. 2. Fuzzy Inference System.

To Optimize the Fuzzy System for Integration of the Mackey-Glass, Dow Jones and the Mexican Stock Exchange time series we implemented a genetic algorithm to optimize the membership functions and rules of the fuzzy system. Where the objective function is defined to minimize the prediction error:

$$f_1 = \left(\sum_{i=1}^{297} |a_i - X_i| \right) / 297$$

$$f_2 = \left(\sum_{i=1}^{297} |b_i - X_i| \right) / 297 \quad (1)$$

$$f_3 = \left(\sum_{i=1}^{297} |c_i - X_i| \right) / 297$$

$$f = \left(\left((f_1 + f_2 + f_3) /3 \right) + \left((R /27) /100 \right) \right)$$

Where $a, b, and\ c,$ correspond to prediction1, prediction2 and prediction3, respectively; these are used as inputs for the fuzzy integration system, X represents real data, f_1, f_2 and f_3 calculates the error of module1, module2, module3 respectively and f is the total prediction error, R is the number of rules generated by the genetic algorithm**,** the possible number of **27** rules and **100** to assign more weight to the error of prediction.

The corresponding chromosome structure is shown in Figure 3.

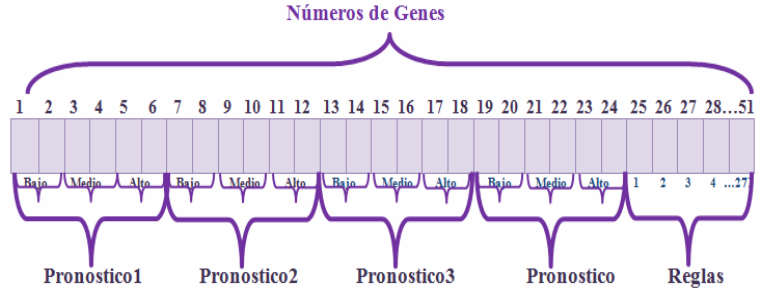

Fig. 3. Chromosome Structure

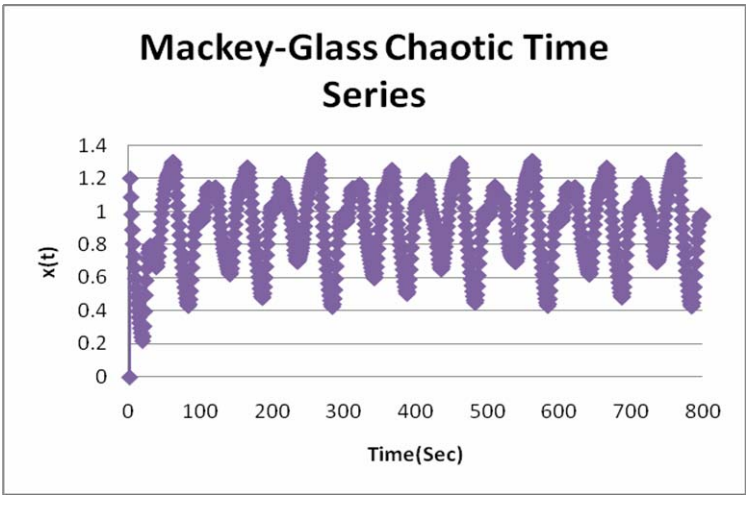

Fig. 4. Series data Mackey-Glass.

Data of the Mackey-Glass time series was generated using equation (1). We are using 800 points. We use 70% of the data for the ensemble neural network trainings and 30% to test the network.

The Mackey-Glass Equation is defined as follows:

$$\dot{x}(t) = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{2}$$

Where it is assumed x (0) = 1.2, t = 17, and x (t) = 0 for t <0. Figure 4 shows a plot of the time series for these parameter values.

This time series is chaotic, and there is no clearly defined period over time. The series does not converge or diverge, and the trajectory is extremely sensitive to the initial conditions. The time series is measured in number of points, and we apply the fourth order Runge-Kutta method to find the numerical solution of the equation [12].

Data of the Dow Jones time series: We are using 800 points that correspond from 11/03/05 to 01/08/09. We used 70% of the data for the ensemble neural network trainings and 30% to test the network [13].

Figure 5 shows a plot of the time series for these parameter values.

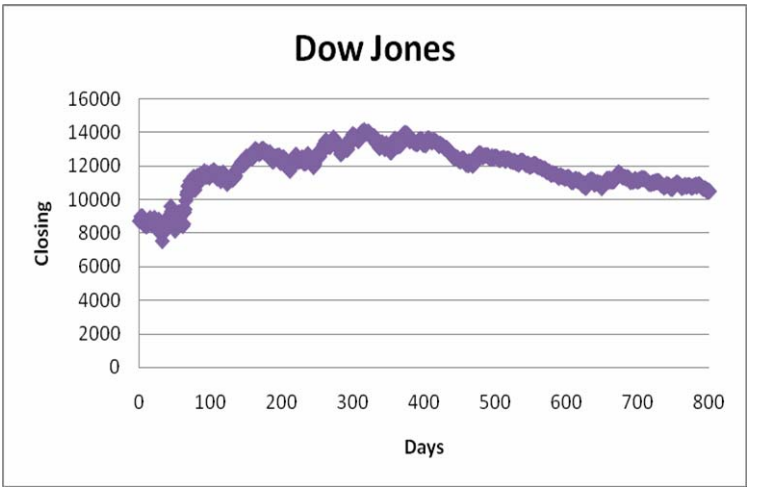

Fig. 5. Series data Dow Jones.

Data of the Mexican Stock Exchange time series: We are using 800 points that correspond from 11/09/05 to 01/15/09. We used 70% of the data for the ensemble neural network trainings and 30% to test the network [14]. We show in Figure 6 the plot of this time series.

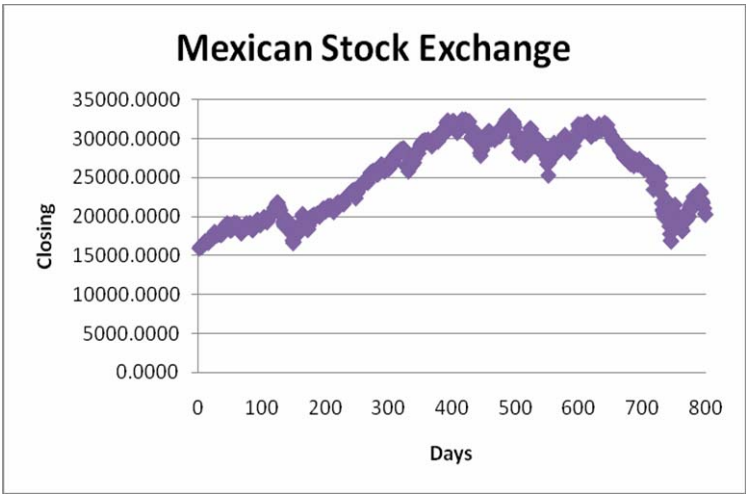

Fig. 6. Mexican Stock Exchange.

7 Simulation Results

In this section we show results for the three time series using the Ensemble NN with fuzzy Integration.

7.1 Simulation Results for the Mackey-Glass Time Series

The architecture of the ensemble network that produced the best results is shown in Figure 7 for the Mackey-Glass Time Series [14].

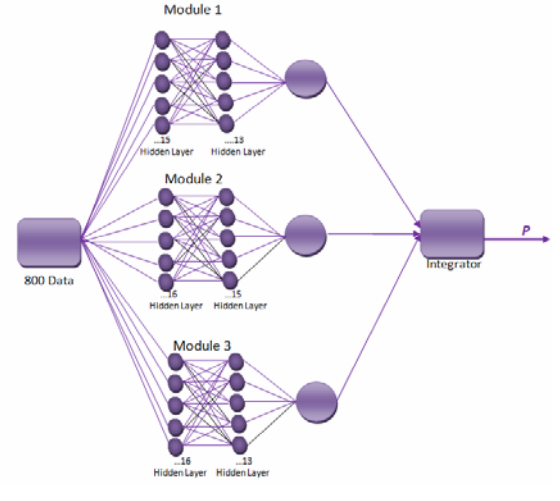

Fig. 7. The best Network Architecture for Mackey-Glass.

In this architecture we used two layers in each module. In module 1, in the first layer we used 15 neurons and 13 neurons in second layer. In module 2 we used 15 neurons in the first layer and 13 neurons in the second, and in module 3 we used 16 neurons in the first layer and 13 neurons in the second. The training method used was the Levenberg-Marquardt (LM); we also applied 3 delays to the network.

The best training was the one shown in row number 3, using 1 layer in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 1) where the total error of this training was: 0.1296.

Table 1. Results of Training using 1 layer for Mackey-Glass.

	Number of Layers	Number of Neurons	Number of Delays	**Target Error**	Number Max Epoch	Epoch	Time	**Error by Module**	Error Total
Training1	1	37 35 35	3	0.00000001 0.00001 0.00001	16000 20000 20000	4 1 1	1 Seg 1 Seg 1 Seg	0.0144 0.0617 0.0417	0.2703
Training2	1	37 35 35	3	0.00000001 0.00001 0.00001	16000 20000 20000	4 2 1	1 Seg 1 Seg 1 Seg	0.0983 0.074 0.024	0.1803
Training3	1	29 30 31	3	0.000001 0.00001 0.00001	6000 2000 5000	3 1 1	1 Seg 1 Seg 1 Seg	0.0777 0.0347 0.0517	0.1296
Training4	1	25 28 29	3	0.0000001 0.0000001 0.00000001	6000 3000 5000	2 2 4	1 Seg 1 Seg 1 Seg	0.1072 0.0356 0.0973	0.1752
Training5	1	27 30 28	3	0.0000001 0.0000001 0.00000001	6000 3000 5000	2 2 7	1 Seg 1 Seg 1 Seg	0.0894 0.0577 0.058	0.1664

The best training was the one shown in row number 5 using 2 layers in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 2) where the total error of this training was: 0.01396.

Table 2. Results of Training using 2 layers for Mackey-Glass.

	Number of Layers	**Number of Neurons**	**Number of Delays**	**Target Error**	**Number Max Epoch**	**Epoch**	**Time**	**Error by Module**	**Total Error**
Training1	2	29,10 18,18 22,12	3	0.00000001 0.00000001 0.000001	11000 20000 10000	7 8 5	1 Seg 1 Seg 1 Seg	0.076 0.0425 0.1157	0.1571
Training2	2	29,12 18,16 20,17	3	0.00000001 0.00000001 0.000001	11000 20000 10000	7 9 8	1 Seg 1 Seg 1 Seg	0.0533 0.0479 0.0795	0.1277
Training3	2	18,12 18,17 21,19	3	0.0000001 0.000000001 0.000001	7000 1000 9000	9 11 7	1 Seg 1 Seg 1 Seg	0.0924 0.0543 0.1672	0.2025
Training4	2	18,13 15,18 15,18	3	0.0000001 0.000000001 0.000000001	7000 1000 1000	10 26 9	1 Seg 1 Seg 1 Seg	0.0579 0.0798 0.073	0.162
Training5	2	15,13 16,15 11,13	3	0.00000001 0.00000001 0.0000001	6000 3000 10000	11 8 10	1 Seg 1 Seg 1 Seg	0.0196 0.0116 0.0098	0.0136

The best training was the one shown in row number 5 varying between 1 and 2 layers in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 3) where the total error of this training was: 0.01396.

Table 3. Results of Training using 1and 2 layers for Mackey-Glass.

	Number of Layers	Number of Neurons	Number of Delays	**Target Error**	Number Max Epoch	Epoch	Time	**Error by** Module	Total Error
Training1	1 y 2	10,17 30 18,11	3	0.0000001 0.0000001 0.00000001	6000 3000 5000	6 1 10	1 Seg 1 Seg 1 Seg	0.0622 0.0295 0.0255	0.1002
Training2	1 y 2	13,12 30 10,17	3	0.00000001 0.00000001 0.00000001	6000 2000 5000	8 11 9	1 Seg 1 Seg 1 Seg	0.027 0.1237 0.0613	0.1712
Training3	1 y2	13,15 31 15,10	3	0.00000001 0.00000001 0.0000001	6000 2000 3000	11 4 11	1 Seg 1 Seg 1 Seg	0.1134 0.1016 0.0407	0.2285
Training4	1 y 2	31 31 15,10	3	0.00000001 0.00000001 0.00000001	2000 2000 5000	55 4 9	1 Seg 1 Seg 1 Seg	0.0848 0.1058 0.0926	0.2215
Training5	1 y 2	23 25 10,14	3	0.000000001 0.0000001 0.0000001	6000 1000 3000	44 1 8	1 Seg 1 Seg 1 Seg	0.0144 0.0617 0.417	0.0899

The result is training number 3 (as shown in Table 4), where the integration method used was average integration; and we obtained an error of 0.025, with weighted average integration we obtained an error of 0.0461, with fuzzy integration we obtained an error of 0.0789 and with optimized fuzzy integration we obtained an error of 0.038131.

The best method of integration for this architecture was the average integration with an error of 0.0205.

Table 4. Results of the Integration Methods with 1 layer for Mackey-Glass.

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0417	0.0558	0.1151	0.05127
Training2	0.317	0.0436	0.0715	0.037832
Training3	0.025	0.0461	0.0789	0.038131
Training4	0.0205	0.0245	0.0694	0.028319
Training5	0.0261	0.0326	0.0787	0.036614

The best result is training 5 (as shown in Table 5), where the integration method used was average integration; we obtained an error of 0.0106, with weighted average integration we obtained an error of 0.0126, with fuzzy integration we

Table 5. Results of the Integration Methods with 2 layers for Mackey-Glass.

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0396	0.0737	0.0813	0.040998
Training2	0.0388	0.0439	0.0815	0.033412
Training3	0.0583	0.075	0.0826	0.052223
Training4	0.0272	0.0355	0.0899	0.04853
Training5	0.0106	0.0126	0.0708	0.023317

obtained an error of 0.0708 and with optimized fuzzy integration we obtained an error of 0.023317.

The best result is training 5 (as shown in Table 6), where the integration method used was average integration; we obtained an error of 0.0241, with weighted average integration we obtained an error of 0.0348, with fuzzy integration we obtained an error of 0.0751 and with optimized fuzzy integration we obtained an error of 0.023317.

Where the best method for this architecture was the average integration with an error of 0.0241.

Table 6. Results of the Integration Methods with 1 and layers for Mackey-Glass

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0264	0.037	0.0681	0.040998
Training2	0.0258	0.053	0.0728	0.033412
Training3	0.03	0.0307	0.0738	0.052223
Training4	0.0312	0.0297	0.0963	0.048869
Training5	0.0241	0.0348	0.0751	0.026237

The best result obtained for the Mackey-Glass time series was number 5 using 2 layers in each of the modules of the ensemble network, implementing 3 delays, the training method used was the Levenberg-Marquardt (LM), (as shown in Table 2 and in Figure 8).

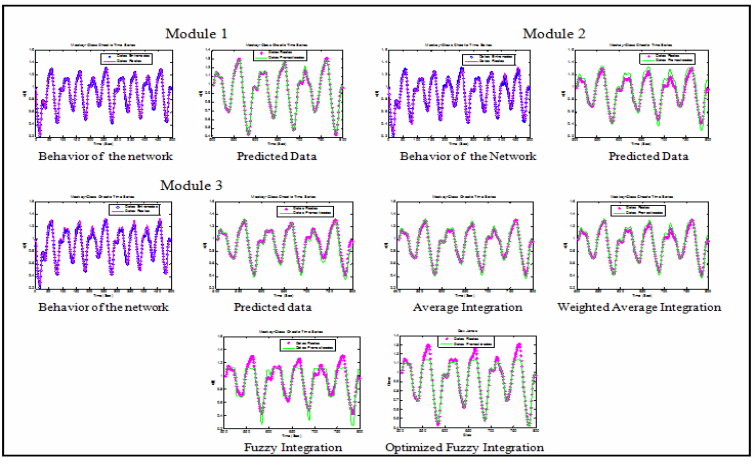

Fig. 8. Simulation results of training 5 for Mackey-Glass.

Table 7. Genetic algorithm results for training 5 for Mackey-Glass.

Num.	Ind.	Gen.	GGP	selection	Mutation	Pm	Crossover	Pc	Pm Rules	Num. Rules	Duration	GA Error	Prediction Error
1	100	100	0.85	rws	mutbga	0.07	xovdprs	0.5	0.002	22	01:56:46	0.031767	0.023619
2	200	100	0.85	rws	mutbga	0.03	xovdprs	0.01	0.002	22	03:10:08	0.030951	0.022803
3	250	100	0.85	rws	mutbga	0.04	xovdprs	0.02	0.002	22	04:17:40	0.033902	0.025754
4	150	100	0.85	rws	mutbga	0.06	xovdprs	0.8	0.002	27	02:36:06	0.035352	0.025352
5	100	100	0.85	rws	mutbga	0.1	xovdprs	1	0.002	27	01:36:32	0.03835	0.02835
6	200	400	0.85	rws	mutbga	0.05	xovdprs	0.75	0.002	20	13:13:53	0.030779	0.0233716
7	100	200	0.85	rws	mutbga	0.05	xovdprs	0.9	0.002	27	03:42:17	0.060882	0.050882
8	100	200	0.85	rws	mutbga	0.06	xovdprs	1	0.002	27	03:36:33	0.045956	0.035956
9	100	150	0.85	rws	mutbga	0.08	xovdprs	0.05	0.002	27	02:19:40	0.046778	0.035778
10	100	150	0.85	rws	mutbga	0.5	xovdprs	0.7	0.002	27	02:23:40	0.051033	0.041033
11	100	150	0.85	rws	mutbga	0.06	xovdprs	0.09	0.002	27	02:23:18	0.057214	0.047214
12	100	150	0.85	rws	mutbga	0.06	xovdprs	0.5	0.002	27	02:17:30	0.049478	0.039478
13	100	150	0.85	rws	mutbga	0.05	xovdprs	0.65	0.002	27	02:22:37	0.043053	0.033053

Table7 shows the genetic algorithm results for training 5 (as shown in Table 2 and in Figure 9). Where the prediction error is of 0.023372.

The fuzzy integration system generated by the genetic algorithm is represented in figure 10:

Comparing Figure 2, which belongs to the base fuzzy integrator (not optimized) with Figure 9, which belongs to the fuzzy integrator generated by the genetic algorithm, it can be seen that the parameters of membership functions are different, the number of rules obtained are 20 and this helps to improve the prediction error.

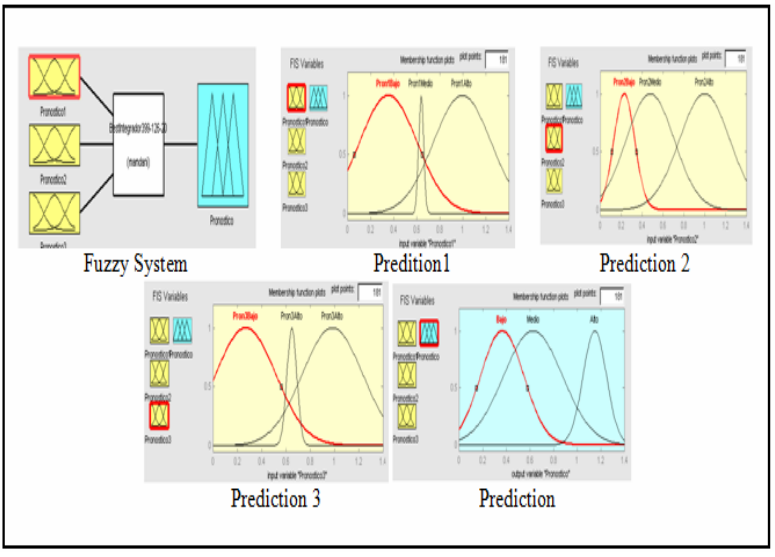

Fig. 9. Fuzzy system generated by genetic algorithm for Mackey-Glass.

1. If (pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto) then (Pronostico is Bajo)
2. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Bajo) then (Pronostico is Bajo)
3. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Alto) then (Pronostico is Bajo)
4. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
5. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
6. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
7. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
8. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
9. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
10. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Medio) then (Pronostico is Medio)
11. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
12. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
13. If (Pronostico1 is Pron1Alto) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
14. If (Pronostico1 is Pron1Alto) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto then (Pronostico is Medio)
15. If (Pronostico1 is Pron1Alto) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
16. If (Pronostico1 is Pron1Alto) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
17. If (Pronostico1 is Pron1Alto) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
18. If (Pronostico1 is Pron1Alto) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Medio) then (Pronostico is Alto)
19. If (Pronostico1 is Pron1Alto) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Alto) then (Pronostico is Alto)
20. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)

Fig. 10. Rules Fuzzy system generated by genetic algorithm for Mackey-Glass.

7.2 Simulation Results for the Dow Jones Time Series

The architecture of the ensemble network that produced the best results for Dow Jones Time Series is shown in Figure 11.

In this architecture we used one layer in each module. In module 1, in the first layer we used 37 neurons in module 2 we used 38 neurons in the first layer and in module 3 we used 38 neurons in first layer. The training method used was the Levenberg-Marquardt (LM); we also applied 3 delays to the network.

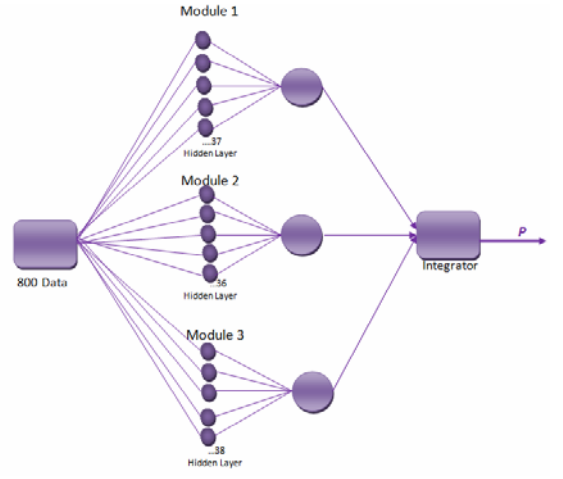

Fig. 11. The Best Network Architecture for Dow Jones.

The best training was the one shown in row number 4, using 1 layer in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 8) where the total error of this training was: 0.00000886.

Table 8. Results of Training using 1 layer for Dow Jones.

	Number of Layers	Number of Neurons	Number of Delays	Target Error	Number Max Epoch	Epoch	Time	Error by Module	Error Total
Training1	1	25	3	0.1	22000	12	00:00:01	0.0005738	
		22		0.1	30000	4	00:00:01	0.00004885	0.00016782
		21		0.1	3000	12	00:00:01	0.00039722	
Training2	1	26	3	0.1	2000	8	00:00:01	0.00003269	
		25		0.1	3000	7	00:00:01	0.00027969	0.00024572
		24		0.1	3500	10	00:00:01	0.00042478	
Training3	1	27	3	0.1	2000	5	00:00:01	0.00024953	
		26		0.1	3000	6	00:00:01	0.00003122	0.00013306
		27		0.1	3500	4	00:00:01	0.00011843	
Training4	1	37	3	0.1	2000	6	00:00:01	0.00000242	
		36		0.1	3000	7	00:00:01	0.00002334	0.00000886
		38		0.1	3500	5	00:00:01	0.00000084	
Training5	1	25	3	0.1	2000	8	00:00:01	0.00001953	
		31		0.1	3000	7	00:00:01	0.00009723	0.00009188
		33		0.1	3500	6	00:00:01	0.00016338	

The best training was the one shown in row number 1 using 2 layers in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 9) where the total error of this training was: 0.00018046.

Table 9. Results of Training using 2 layers for Dow Jones.

	Number of Layers	Number of Neurons	Number of Delays	Target Error	Number Max Epoch	Epoch	Time	Error by Module	Error Total
Training1	2	19,17	3	0.1	5000	8	00:00:02	0.00001458	
		19,17		0.01	5000	7	00:00:02	0.00013627	0.00018046
		15,19		0.1	3500	7	00:00:01	0.00039054	
Training2	2	20,18	3	0.1	5000	7	00:00:01	0.00019284	
		19,18		0.1	5000	6	00:00:01	0.00026851	0.00030629
		18,19		0.1	3500	6	00:00:01	0.00045751	
Training3	2	20,22	3	0.1	2000	6	00:00:03	0.00016737	
		21,22		0.1	5000	9	00:00:03	0.00072676	0.00032187
		21,23		0.1	3500	8	00:00:03	0.00007148	
Training4	2	12,14	3	0.1	2000	8	00:00:02	0.00028912	
		15,13		0.1	5000	7	00:00:01	0.00053855	0.00036097
		16,15		0.1	3500	4	00:00:01	0.00025523	
Training5	2	13,14	3	0.1	2000	6	00:00:01	0.00020369	
		15,16		0.1	5000	5	00:00:01	0.00047477	0.00028602
		16,18		0.1	3500	6	00:00:01	0.00017961	

The best training was the one shown in row number 4 varying between 1 and 2 layers in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 10) where the total error of this training was: 0.00008543.

Table 10. Results of Training using 1 y 2 layers for Dow Jones.

	Number of Layers	Number of Neurons	Number of Delays	Target Error	Number Max Epoch	Epoch	Time	Error by Module	Error Total
Training1	1 y 2	21,21	3	0.1	2000	12	00:00:12	0.000155	
		38		0.1	3500	10	00:00:01	0.000412	0.000357
		21,21		0.1	3500	16	00:00:16	0.000504	
Training2	1 y 2	17,15	3	0.1	2000	7	00:00:03	0.0001679	
		32		0.1	3500	4	00:00:01	0.0000573	0.0001957
		15,16		0.1	3500	6	00:00:02	0.0001334	
Training3	1 y 2	17,19	3	0.1	2000	6	00:00:02	0.00019984	
		34		0.1	3500	5	00:00:00	0.00007986	0.00010505
		17		0.1	3500	5	00:00:01	0.00003545	
Training4	1 y 2	30	3	0.1	2000	7	00:00:01	0.00007017	
		19,17		0.1	3500	7	00:00:01	0.00013854	0.00008543
		32		0.1	3500	6	00:00:00	0.00004757	
Training5	1 y 2	34	3	0.1	2000	5	00:00:01	0.00017922	
		18,16		0.1	3500	7	00:00:01	0.00009751	0.00009281
		34		0.1	3500	7	00:00:00	0.0000017	

The result is training number 4 (as shown in Table 11), where the integration method used was average integration; and we obtained an error of 0.0053, with weighted average integration we obtained an error of 0.0060, with fuzzy integration

we obtained an error of 0.0194 and with optimized fuzzy integration we obtained an error of 0.006863.

The best method of integration for this architecture was the average integration with an error of 0.0053.

Table 11. Results of the Integration Methods with 1 layer for Dow Jones.

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0215	0.0290	0.0207	0.007044
Training2	0.0408	0.0466	0.0225	0.007108
Training3	0.0271	0.0336	0.0190	0.007176
Training4	0.0053	0.0060	0.0194	0.006863
Training5	0.0085	0.0117	0.0257	0.007879

The best result is training 1 (as shown in Table 12), where the integration method used was average integration; we obtained an error of 0.0208, with weighted average integration we obtained an error of 0.0406, with fuzzy integration we obtained an error of 0.0315 and with optimized fuzzy integration we obtained an error of 0.008987.

The best method of integration for this architecture was the optimized fuzzy Integration with an error of 0.008987.

Table 12. Results of the Integration Methods with 2 layers for Dow Jones.

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0208	0.0406	0.0226	0.008987
Training2	0.0205	0.0338	0.0315	0.006189
Training3	0.0588	0.0656	0.0282	0.009745
Training4	0.0419	0.0537	0.0356	0.006242
Training5	0.0214	0.0460	0.0378	0.012783

The best result is training 4 (as shown in Table 13), where the integration method used was average integration; we obtained an error of 0.0069, with weighted average integration we obtained an error of 0.0098, with fuzzy integration

we obtained an error of 0.0455 and with optimized fuzzy integration we obtained an error of 0.008387.

The best method of integration for this architecture was the average integration with an error of 0.008387.

Table 13. Results of the Integration Methods with 1and 2 layers for Mackey-Glass.

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0095	0.0146	0.0305	0.008766
Training2	0.0124	0.0157	0.0312	0.007788
Training3	0.0097	0.0174	0.0241	0.010623
Training4	0.0069	0.0098	0.0455	0.008387
Training5	0.0102	0.0132	0.0365	0.009729

The best result obtained for the Dow Jones time series was number 4 using 1 layer in each of the modules of the ensemble network, implementing 3 delays, the training method used was Levenberg-Marquardt (LM), (as shown in Table 13 and in Figure 12).

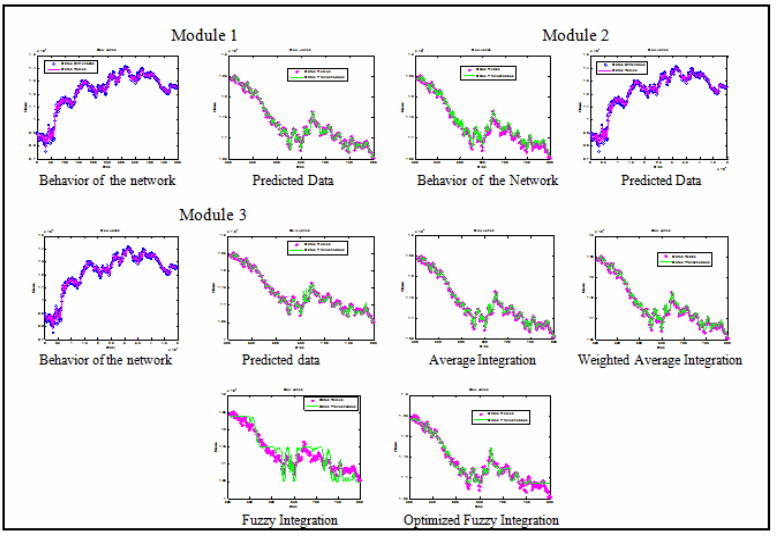

Fig. 12. Simulation results of training 4 for Dow Jones time series.

Table 14. Genetic algoritm results for training 4 for the Dow Jones time series.

Num.	Ind.	Gen.	GGP	Selection	Mutation	Pm	Crossover	Pc	Pm Rules	Núm. Rules	Duration	GA Error	Prediction Error
1	100	100	0.85	rws	mutbga	0.8	xovdprs	1	0.002	27	02:06:04	0.018556	0.017556
2	100	100	0.85	rws	mutbga	0.07	xovdprs	0.5	0.002	22	02:07:44	0.015011	0.006863
3	100	100	0.85	rws	mutbga	0.6	xovdprs	1	0.002	22	02:05:34	0.016055	0.007907
4	100	100	0.85	rws	mutbga	0.06	xovdprs	0.8	0.002	20	02:06:33	0.01828	0.01051
5	100	100	0.85	rws	mutbga	0.5	xovdprs	1	0.002	22	02:06:08	0.018319	0.010171
6	100	100	0.85	rws	mutbga	0.05	xovdprs	0.8	0.002	21	02:07:49	0.015718	0.007941
7	100	100	0.85	rws	mutbga	0.06	xovdprs	0.9	0.002	20	02:07:49	0.016283	0.008876
8	100	100	0.85	rws	mutbga	0.06	xovdprs	0.5	0.002	22	02:09:36	0.016539	0.008391
9	100	100	0.85	rws	mutbga	0.7	xovdprs	0.5	0.002	22	02:34:40	0.021852	0.013704
10	100	100	0.85	rws	mutbga	0.09	xovdprs	0.5	0.002	22	02:39:27	0.020939	0.012791

Table14 shows the genetic algorithm results for training 4 (as shown in Table 13 and in Figure 12) where the prediction error is of 0.006863.

The fuzzy integration system generated by the genetic algorithm is represented in Figure 13 and the fuzzy rules in Figure 14.

Comparing Figure 2, which belongs to the base fuzzy integrator (not optimized) with Figure 13, which belongs to the fuzzy integrator generated by the genetic algorithm, it can be seen that the parameters of membership functions are different, the number of rules obtained are 22 and this helps to improve the error prediction.

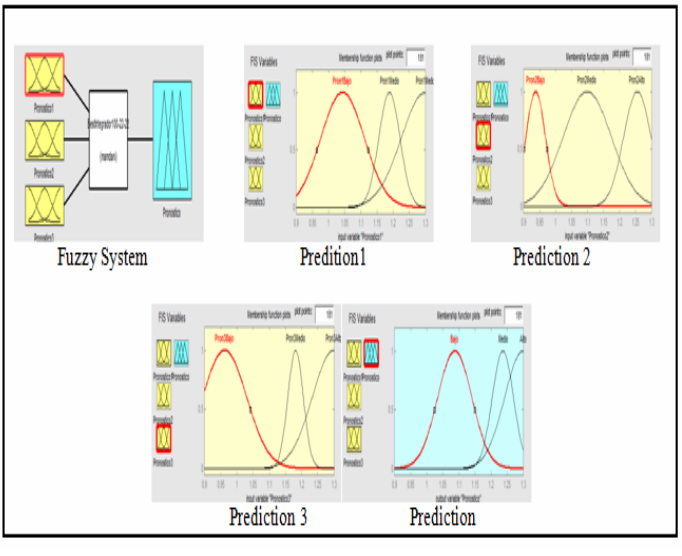

Fig. 13. Fuzzy system generated by the genetic algorithm for the Dow Jones time series.

1. If (pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
2. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto) then (Pronostico is Bajo)
3. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Bajo) then (Pronostico is Bajo)
4. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
5. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Bajo) then (Pronostico is Bajo)
6. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
7. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
8. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
9. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
10. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Medio) then (Pronostico is Alto)
11. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
12. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
13. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Medio) then (Pronostico is Medio)
14. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Bajo) then (Pronostico is Medio)
15. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
16. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
17. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
18. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
19. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
20. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Medio) then (Pronostico is Alto)
21. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Alto) then (Pronostico is Alto)
22. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)

Fig. 14. Rules of the Fuzzy system generated by genetic algorithm for Mexican Stock Exchange.

7.3 Simulation Results for the Mexican Stock Exchange Time Series

The architecture of the ensemble network that produced the best results for Mexican Stock Exchange Time Series is shown in Figure 15.

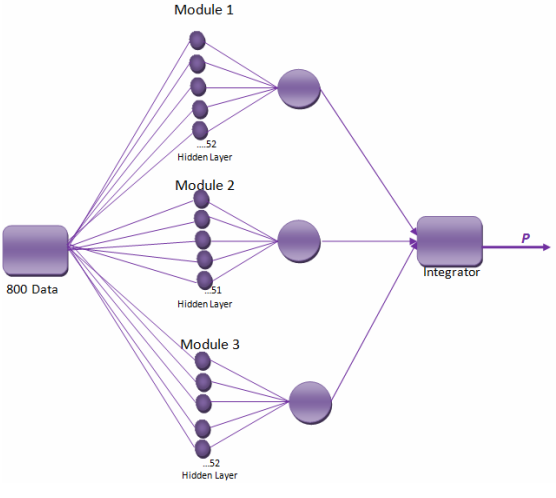

Fig. 15. The Best Network Architecture for Mexican Stock Exchange.

In this architecture we used one layer in each module. In module 1, in the first layer we used 52 neurons in module 2 we used 51 neurons in the first layer and in module 3 we used 52 neurons in first layer. The training method used was the Levenberg-Marquardt (LM); we also applied 3 delays to the network.

The best training was the one shown in row number 4, using 1 layer in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 15) where the total error of this training was: 0.00015.

Table 15. Results of Training using 1 layer for Mexican Stock Exchange.

	Number of Layers	Number of Neurons	Number of Delays	Target Error	Number Max Epoch	Epoch	Time	Error by Module	Error Total
Training1	1	40	3	0.1	22000	82	00:00:09	0.00019	0.00029
		65		0.1	30000	17	00:00:03	0.00046	
		70		0.1	30000	33	00:00:06	0.00022	
Training2	1	42	3	0.1	22000	31	00:00:03	0.00007	
		67		0.1	30000	16	00:00:02	0.00047	0.00021
		74		0.1	30000	80	00:00:11	0.00009	
Training3	1	50	3	0.1	22000	27	00:00:03	0.00022	0.00021
		50		0.1	30000	69	00:00:06	0.00009	
		50		0.1	30000	10	00:00:01	0.00033	
Training4	1	52	3	0.1	22000	23	00:00:03	0.00022	0.00015
		51		0.1	30000	38	00:00:03	0.00012	
		52		0.1	30000	7	00:00:01	0.00011	
Training5	1	53	3	0.1	2000	6	00:00:02	0.00067	0.00031
		52		0.1	3000	18	00:00:02	0.00006	
		54		0.1	3000	57	00:00:05	0.00022	

The best training was the one shown in row number 1 using 2 layers in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 16) where the total error of this training was: 0.000195.

Table 16. Results of Training using 2 layers for Mexican Stock Exchange.

	Number of Layers	Number of Neurons	Number of Delays	Target Error	Number Max Epoch	Epoch	Time	Error by Module	Error Total
Training1	2	26,26	3	0.1	2000	17	00:00:30	0.000407	0.000195
		32,32		0.1	5000	16	00:00:16	0.000086	
		22,22		0.1	3500	9	00:00:09	0.000093	
Training2	2	27,27		0.1	2000	15	00:00:30	0.000479	
		25,25	3	0.1	5000	12	00:00:16	0.000427	0.000389
		27,27		0.1	3500	15	00:00:09	0.000270	
Training3	2	28,28	3	0.1	2000	10	00:00:08	0.0000045	
		29,29		0.1	5000	14	00:00:09	0.0006578	0.000646
		28,27		0.1	3500	12	00:00:07	0.0013	
Training4	2	27,28	3	0.1	2000	7	00:00:06	0.000348	
		26,27		0.1	5000	9	00:00:07	0.000214	0.000220
		30,30		0.1	3500	832	00:08:55	0.000997	
Training5	2	29,29	3	0.1	2000	10	00:00:09	0.000613	
		28,28		0.1	5000	12	00:00:10	0.000310	0.000552
		30,30		0.1	3500	9	00:00:09	0.000732	

The best training was the one shown in row number 5 varying between 1 and 2 layers in each of the modules of the ensemble network, implementing 3 delays in the network, the training method used was Levenberg-Marquardt (LM), (as shown in Table 17) where the total error of this training was: 0.00016.

Table 17. Results of Training using 1and 2 layers for Mexican Stock Exchange.

	Number of Layers	Number of Neurons	Number of Delays	Target Error	Number Max Epoch	Epoch	Time	Error by Module	Error Total
Training1	1 y 2	30,30	3	0.1	2000	12	00:00:12	0.0020	0.0009
		50		0.1	5000	10	00:00:01	0.0002	
		52,52		0.1	3500	16	00:02:37	0.0006	
Training2	1 y 2	54	3	0.1	2000	6	00:00:01	0.0003	
		52		0.1	5000	9	00:00:01	0.0006	0.00052
		30,28		0.1	3500	15	00:00:13	0.0006	
Training3	1 y 2	54	3	0.1	2000	11	00:00:11	0.00034	
		32,32		0.1	5000	11	00:00:11	0.00046	0.00045
		55		0.1	3500	9	00:00:09	0.00056	
Training4	1 y 2	54	3	0.1	2000	10	00:00:02	0.00016	
		22,21		0.1	5000	10	00:00:03	0.00004	0.00016
		55		0.1	3500	8	00:00:02	0.00029	
Training5	1 y 2	55	3	0.1	2000	8	00:00:02	0.00015	
		19,19		0.1	5000	160	0:00:28	0.00089	0.00038
		54		0.1	3500	8	00:00:01	0.00012	

The result is training number 4 (as shown in Table 18), where the integration method used was average integration; and we obtained an error of 0.0479, with weighted average integration we obtained an error of 0.0519, with fuzzy integration we obtained an error of 0.1094 and with optimized fuzzy integration we obtained an error of 0.045848.

The best method of integration for this architecture was the optimized fuzzy integration with an error of 0.045848.

Table 18. Results of the Integration Methods with 1 layer for Mexican Stock Exchange

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0410	0.0619	0.1224	0.043856
Training2	0.0919	0.1017	0.1575	0.046531
Training3	0.0468	0.0500	0.1223	0.046123
Training4	0.0479	0.0519	0.1094	0.045848
Training5	0.0570	0.0701	0.1062	0.044292

The best result is training 1 (as shown in Table 19), where the integration method used was average integration; we obtained an error of 0.0815, with weighted average integration we obtained an error of 0.0926, with fuzzy integration we obtained an error of 0.1286 and with optimized fuzzy integration we obtained an error of 0.05337.

The best method of integration for this architecture was the optimized fuzzy integration with an error of 0.05337.

Table 19. Results of the Integration Methods with 2 layers for Mexican Stock Exchange

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0815	0.0926	0.1286	0.05337
Training2	0.0647	0.0831	0.1341	0.06387
Training3	0.0612	0.0830	0.1386	0.065554
Training4	0.0536	0.0693	0.1237	0.049567
Training5	0.0504	0.0696	0.1336	0.050638

The best result is training 4 (as shown in Table 20), where the integration method used was average integration; we obtained an error of 0.0458, with weighted average integration we obtained an error of 0.0529, with fuzzy integration we obtained an error of 0.1023 and with optimized fuzzy integration we obtained an error of 0.046131.

The best method for this architecture was the average integration with an error of 0.0458.

Table 20. Results of the Integration Methods with 1y 2 layers for Mexican Stock Exchange

	Average Integration	Weighted Average Integration	Fuzzy Integration	Optimized Fuzzy Integration
Training1	0.0883	0.1187	0.1491	0.050886
Training2	0.0474	0.0804	0.1250	0.052601
Training3	0.0561	0.0774	0.1094	0.051523
Training4	0.0458	0.0529	0.1023	0.046131
Training5	0.0731	0.0843	0.1270	0.051713

The best result obtained for the Mexican Stock Exchange Time Series. was number 4 using 1 layers in each of the modules of the ensemble network, implementing 3 delays, the training method used was Levenberg-Marquardt (LM), (as shown in Table 20 and in Figure16).

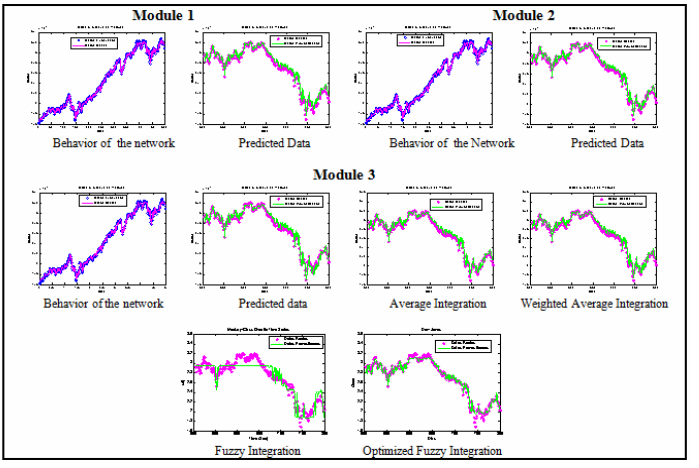

Fig. 16. Simulation results of training 4 for the Mexican Stock Exchange.

Table 21. Genetic algoritm results for training 4 for Mexican Stock Exchange.

Ind.	Gen.	GGP	Selection	Mutation	Pm	Crossover	Pc	Pm Rules	Núm. Rules	Duration	GA Error	Prediction Error
100	100	0.85	rws	mutbga	0.8	xovdprs	1	0.002	27	03:25:59	0.069886	0.059886
100	100	0.85	rws	mutbga	0.07	xovdprs	0.5	0.002	20	03:39:59	0.053679	0.046272
100	100	0.85	rws	mutbga	0.6	xovdprs	1	0.002	22	03:51:53	0.067592	0.059444
100	100	0.85	rws	mutbga	0.09	xovdprs	0.5	0.002	27	04:09:37	0.063615	0.053615
100	100	0.85	rws	mutbga	0.06	xovdprs	0.8	0.002	22	02:58:50	0.065992	0.057884
100	100	0.85	rws	mutbga	0.5	xovdprs	1	0.002	27	03:08:20	0.071177	0.061177
100	100	0.85	rws	mutbga	0.05	xovdprs	0.8	0.002	20	03:14:33	0.069961	0.062554
100	100	0.85	rws	mutbga	0.06	xovdprs	0.9	0.002	22	03:30:04	0.053996	0.045848
100	100	0.85	rws	mutbga	0.6	xovdprs	0.05	0.002	22	03:23:30	0.058665	0.050517
100	100	0.85	rws	mutbga	0.7	xovdprs	0.5	0.002	27	03:28:07	0.065534	0.055534

Table 21 shows the genetic algorithm results for training 4 (as shown in Table 20 and in Figure 16) where the prediction error is of 0.045848.

The fuzzy integration system generated by the genetic algorithm is represented in Figure 17 and the fuzzy rules in Figure 18.

Comparing Figure 2, which belongs to the base fuzzy integrator (not optimized) with Figure 17, which belongs to the fuzzy integrator generated by the genetic algorithm, it can be seen that the parameters of the membership functions are different , the number of rules obtained are 22 and this helps to improve the prediction error.

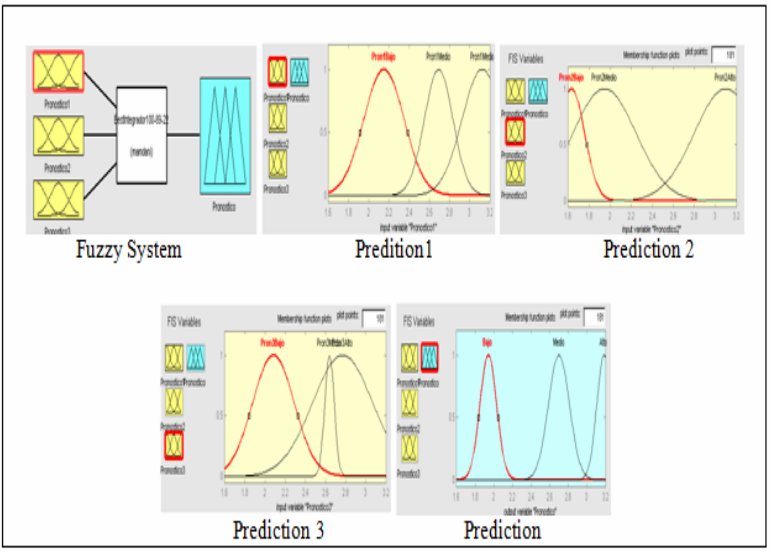

Fig. 17. Fuzzy system generated by genetic algorithm for Mexican Stock Exchange.

1. If (pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
2. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto) then (Pronostico is Bajo)
3. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Bajo) then (Pronostico is Bajo)
4. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
5. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
6. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
7. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
8. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
9. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
10. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
11. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
12. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Medio) then (Pronostico is Medio)
13. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Bajo then (Pronostico is Medio)
14. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
15. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Bajo) then (Pronostico is Alto)
16. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Medio) then (Pronostico is Bajo)
17. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Bajo) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
18. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Medio) then (Pronostico is Medio)
19. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Medio) and (Pronostico3 is Pron3Alto) then (Pronostico is Medio)
20. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Medio) then (Pronostico is Alto)
21. If (Pronostico1 is Pron1Medio) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Alto) then (Pronostico is Alto)
22. If (Pronostico1 is Pron1Bajo) and (Pronostico2 is Pron2Alto) and (Pronostico3 is Pron3Alto) then (Pronostico is Alto)

Fig. 18. Rules of the Fuzzy system generated by genetic algorithm for Mexican Stock Exchange.

8 Conclusions

The best result obtained for the Mackey-Glass series training was using the ensemble neural network architecture with 2 layers using 3 delays. The error obtained by the average integration was 0.0106, the error obtained by the average weighted integration was 0.0126 and the error obtained by the fuzzy integration was 0.0708. The best result obtained for the Dow Jones series training was

using the ensemble neural network architecture with 1 layer using 3 delays. The error obtained by the average integration was 0.0053, the error obtained by the average weighted integration was 0.0060 and the error obtained by the fuzzy integration was 0.0194. The best result for the Mexican Stock Exchange series training was obtained using an ensemble neural network architecture with 1 layer using 3 delays. The error obtained by the average integration was 0.0479, by the average weighted integration was 0.0519 and by the fuzzy integration was 0.1094, respectively.

Since the results were not satisfactory with the fuzzy integration system, it was decided to implement an evolutionary approach to optimize the membership functions and rules of this system. The method chosen to optimize this integration system was a genetic algorithm. After applying the genetic algorithm we were able to obtain an error of 0.023317 for the Mackey-Glass Series, an error of 0.006863 for the Dow Jones time series and an error of 0.045848 for the Mexican Stock Exchange time series, with this improving the results obtained with respect to the non optimized system.

Acknowledgment

We would like to express our gratitude to the CONACYT, Tijuana Institute of Technology for the facilities and resources granted for the development of this research.

References

[1] Vásquez, G., Muñoz, Tapia, J.: "Series de Tiempo" Catholic University of the Holy Concepción, Chile (2001), http://zip.rincondelvago.com/?00033622 (December 05, 2008)

[2] Arellano, M.: Introduction to time series analysis (2001), http://ciberconta.unizar.es/LECCION/seriest/100.HTM (September 06, 2008)

[3] Davey, N., Hunt, S., Frank, R.: Time Series Prediction and Neural Networks, University of Hertfordshire, Hatfield, UK (1999) (December 04, 2008)

[4] Plummer, E.A.: Time series forecasting with feed-forward neural Networks: uidelines and limitations. University of Wyoming (July 2000), http://www.karlbranting.net/papers/plummer/Paper_7_12_00.htm (January 20, 2009)

[5] Neural Network Tutorial, http://www.answermath.com/neural-networks/tutorial-esp-2-concepto.htm (December 05, 2008)

[6] Multaba, I.M., Hussain, M.A.: Application of Neural Networks and Other Learning. Technologies in Process Engineering. Imperial Collage Press (2001) (December 03, 2008)

[7] Basic Concepts of Fuzzy Logic, http://www.tdx.cbuc.es/TESIS_UPC/AVAILABLE/TDX0207105105056//04Rpp04de11.pdf (September 10, 2008)

[8] Jang, J.S.R., Sun, C.T., Mizutani, E.: Neuro-Fuzzy and Soft Computing. Prentice Hall, New Jersey
[9] Holland, J.: Adaptation in natural and artificial systems. University of Michigan Press (1975)
[10] Golberg, D.: Genetic Algorithms in search, optimization and machine learning. Addison Wesley, Reading (1989)
[11] Yen, J., Langari, R.: Fuzzy Logic: intelligence, control and Information. Prentice Hall, New Jersey (1999)
[12] MATLAB, Historical Mackey Glass data (2007)
[13] Dow Jones Indexes, http://www.djindexes.com (September 5, 2008)
[14] Mexico Bank, http://www.banxico.org.mx (December 10, 2008)