

A Hybrid Approach with Modular Neural Networks and Fuzzy Logic for Time Series Prediction

Omar Blanchet, Martha Ramirez, Maribel Gutierrez, Jassiny Quintero, Alejandra Mancilla and Patricia Melin
Dept. of Computer Science, Tijuana Institute of Technology

Abstract-We describe in this paper the application of several neural network architectures to the problem of simulating and predicting the dynamic behavior of complex economic time series. We use several neural network models and training algorithms to compare the results and decide at the end, which one is best for this application. We also compare the simulation results with the traditional approach of using a statistical model. In this case, we use real time series of prices of consumer goods to test our models. Real prices of tomato in the U.S. and Mexico show complex fluctuations in time and are very complicated to predict with traditional approaches.

I. INTRODUCTION

Forecasting refers to a process by which the future behavior of a dynamical system is estimated based on our understanding and characterization of the system. If the dynamical system is not stable, the initial conditions become one of the most important parameters of the time series response, i.e. small differences in the start position can lead to a completely different time evolution. This is what is called sensitive dependence on initial conditions, and is associated with chaotic behavior [2, 16] for the dynamical system.

The financial markets are well known for wide variations in prices over short and long terms. These fluctuations are due to a large number of deals produced by agents that act independently from each other. However, even in the middle of the apparently chaotic world, there are opportunities for making good predictions [4,5]. Traditionally, brokers have relied on technical analysis, based mainly on looking at trends, moving averages, and certain graphical patterns, for performing predictions and subsequently making deals. Most of these linear approaches, such as the well-known Box-Jenkins method, have disadvantages [9].

More recently, soft computing [10] methodologies, such as neural networks, fuzzy logic, and genetic algorithms, have been applied to the problem of forecasting complex time series. These methods have shown clear advantages over the traditional statistical ones [12]. The main advantage of soft computing methodologies is that, we do not need to specify the structure of a model a-priori, which is clearly needed in the classical regression analysis [3]. Also, soft computing models are non-

linear in nature and they can approximate more easily complex dynamical systems, than simple linear statistical models. Of course, there are also disadvantages in using soft computing models instead of statistical ones. In classical regression models, we can use the information given by the parameters to understand the process, i.e. the coefficients of the model can represent the elasticity of price for a certain good in the market.

II. MONOLITHIC NEURAL NETWORK MODELS

A neural network model takes an input vector X and produces an output vector Y . The relationship between X and Y is determined by the network architecture. There are many forms of network architecture (inspired by the neural architecture of the brain). The neural network generally consists of at least three layers: one input layer, one output layer, and one or more hidden layers. Fig. 1 illustrates a neural network with p neurons in the input layer, one hidden layer with q neurons, and one output layer with one neuron.

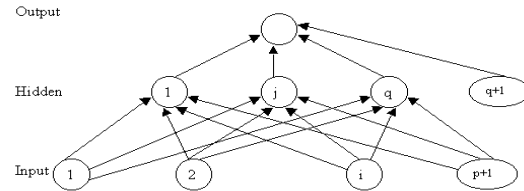


Fig. 1 Single hidden layer feed-forward network.

In the neural network we will be using, the input layer with $p+1$ processing elements, i.e., one for each predictor variable plus a processing element for the bias. The bias element always has an input of one, $X_{p+1}=1$. Each processing element in the input layer sends signals X_i ($i=1, \dots, p+1$) to each of the q processing elements in the hidden layer. The q processing elements in the hidden layer (indexed by $j=1, \dots, q$) produce an “activation” $a_j=F(\sum w_{ij}X_i)$ where w_{ij} are the weights associated with the connections between the $p+1$ processing elements of the input layer and the j th processing element of the hidden layer.

$$Y_t = \sum_{j=1}^{p+1} \pi_j x_{jt} + \sum_{j=1}^{p+1} \theta_j F \left(\sum_{i=1}^{p+1} w_{ij} x_{it} \right) \quad (1)$$

Here π_i are the weights for the connections between the input layer and the output layer, and θ_j are the weights for the connections between the hidden layer and the output layer. The main requirement to be satisfied by the activation function $F(\cdot)$ is that it be nonlinear and differentiable. Typical functions used are the sigmoid, hyperbolic tangent, and the sine functions. The weights in the neural network can be adjusted to minimize some criterion such as the sum of squared error (SSE) function:

$$E_1 = \frac{1}{2} \sum_{i=1}^n (d_i - y_i)^2 \quad (2)$$

Thus, the weights in the neural network are similar to the regression coefficients in a linear regression model. In fact, if the hidden layer is eliminated, (1) reduces to the well-known linear regression function. It has been shown [22] that, given sufficiently many hidden units, (1) is capable of approximating any measurable function to any accuracy. In fact $F(\cdot)$ can be an arbitrary sigmoid function without any loss of flexibility.

The most popular algorithm for training feedforward neural networks is the backpropagation algorithm [14,18]. As the name suggests, the error computed from the output layer is backpropagated through the network, and the weights are modified according to their contribution to the error function. Essentially, backpropagation performs a local gradient search, and hence its implementation does not guarantee reaching a global minimum. A number of heuristics are available to partly address this problem, some of which are presented below. Instead of distinguishing between the weights of the different layers as in Equation (1), we refer to them generically as w_{ij} in the following. After some mathematical simplification the weight change equation suggested by backpropagation can be expressed as follows:

$$\Delta w_{ij} = -\eta \frac{\partial E_1}{\partial w_{ij}} + \theta \Delta w_{ij} \quad (3)$$

Here, η is the learning coefficient and θ is the momentum term. One heuristic that is used to prevent the neural network from getting stuck at a local minimum is the random presentation of the training data. Another heuristic that can speed up convergence is the cumulative update of weights, i.e., weights are not updated after the presentation of each input-output pair, but are accumulated until a certain number of presentations are made, this number referred to as an “epoch”.

III. MODULAR NEURAL NETWORKS

There exists a lot of neural network architectures in the literature that work well when the number of inputs is relatively small, but when the complexity of

the problem grows or the number of inputs increases, their performance decreases very quickly. For this reason, there has also been research work in compensating in some way the problems in learning of a single neural network over high dimensional spaces.

In the literature there is also mention of the terms “ensemble” and “modular” for this type of neural network. The term “ensemble” is used when a redundant set of neural networks is utilized, as described in Hansen and Salamon [8]. In this case, each of the neural networks is redundant because it is providing a solution for the same task, as it is shown in Fig. 2. On the other hand, in the modular approach, one task or problem is decompose in subtasks, and the complete solution requires the contribution of all the modules, as it is shown in Fig. 3.

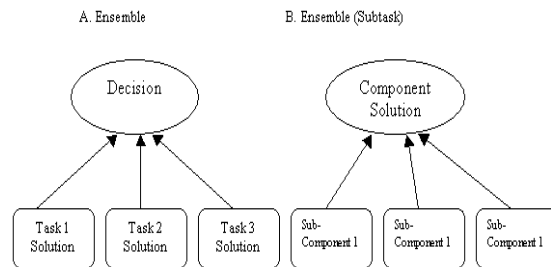


Fig. 2 Ensembles for one task and subtask.

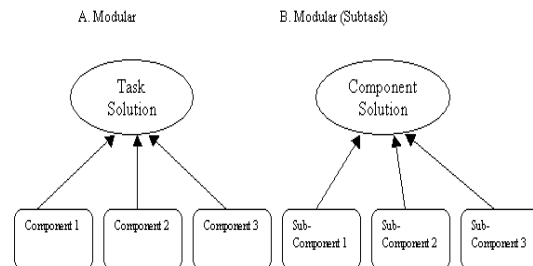


Fig. 3 Modular approach for task and subtask.

A. Multiple Neural Networks

In this approach we can find networks that use strongly separated architectures. Each neural network works independently in its own domain. Each of the neural networks is build and trained for a specific task. The final decision is based on the results of the individual networks, called agents or experts.

One example of this decision is shown by Schmidt [19], as shown in Fig. 4, where a multiple architecture is used, one module consists of a neural network trained for recognizing a person by the voice, while the other module is a neural network trained for recognizing a person by the image.

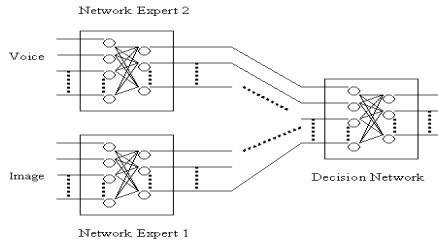


Fig. 4 Multiple networks for voice and image.

The outputs by the experts are the inputs to the decision network, which is the one making the decision based on the outputs of the expert networks.

B. Main Architectures with Multiple Networks

Within multiple neural networks we can find three main classes of this type of networks [7]:

- **Mixture of Experts (ME):** The mixture of experts can be viewed as a modular version of the multi-layer networks with supervised training or the associative version of competitive learning. In this design, the local experts are trained with the data sets to mitigate weight interference from one expert to the other.
- **Gate of Experts:** In this case, an optimization algorithm is used for the gating network, to combine the outputs from the experts.
- **Hierarchical Mixture of Experts:** In this architecture, the individual outputs from the experts are combined with several gating networks in a hierarchical way.

C. “Modular” Neural Networks

The term “Modular Neural Networks” is very fuzzy. It is used in a lot of ways and with different structures. Everything that is not monolithic is said to be modular. In the research work by Boers and Kuiper [1], the concept of a modular architecture is introduced as the development of a large network using modules.

One of the main ideas of this approach is presented in [19], where all the modules are neural networks. The architecture of a single module is simpler and smaller than the one of a monolithic network. The tasks are modified in such a way that training a subtask is easier than training the complete task. Once all modules are trained, they are connected in a network of modules, instead of using a network of neurons. The modules are independent to some extent, which allows working in parallel. Another idea about modular networks is presented by Boers and Kuiper [1], where they used an approach of networks not totally connected. In this model, the structure is more difficult to analyze, as shown in Fig. 5. A clear separation between modules can't be

made. Each module is viewed as a part of the network totally connected.

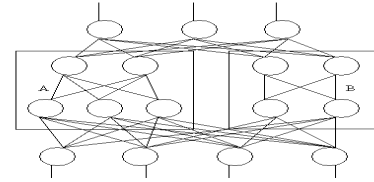


Fig. 5 One specific type of modular network.

In this figure, we can appreciate two different sections from the monolithic neural network, namely A and B. Since there are no connections between both parts of the network, the dimensionality (number of weights) is reduced. As a consequence the required computations are decreased and speed of convergence is increased.

D. Main Task Decomposition into Subtasks

Task Decomposition can be performed in three different ways, as mentioned by Lu and Ito [11]:

- **Explicit Decomposition:** In this case, decomposition is made before learning and requires that the designer has deep knowledge about the problem. Of course, this maybe a limitation if there isn't sufficient knowledge about the problem.
- **Automatic Decomposition:** In this case, decomposition is made as learning is progressing.
- **Decomposition into Classes:** This type of decomposition is made before learning, a problem is divided into a set of sub-problems according to the intrinsic relations between the training data. This method only requires knowledge about the relations between classes.

IV. METHODS FOR RESPONSE INTEGRATION

The importance of this part of the architecture for pattern recognition is due to the high dimensionality of this type of problems. As a consequence in pattern recognition is good alternative to consider a modular approach. This has the advantage of reducing the time required of learning and it also increases accuracy. In our case, we consider dividing the images of a human face in three different regions. We also divide the fingerprint into three parts, and applying a modular structure for achieving pattern recognition.

Yager [23] mentions in his work, that fuzzy measures for the aggregation criteria of two important classes of problems. In the first type of problems, we have a set $Z = \{z_1, z_2, \dots, z_n\}$ of objects, and it is desired to select one or more of these objects

based on the satisfaction of certain criteria. In this case, for each $z_i \in Z$, it is evaluated $D(z_i) = G(A_i(z_i), \dots, A_j(z_i))$, and then an object or objects are selected based on the value of G . The problems that fall within this structure are the multi-criteria decision problems, search in databases and retrieving of documents. In the second type of problems, we have a set $G = \{G_1, G_2, \dots, G_q\}$ of aggregation functions and object z . Here, each G_k corresponds to different possible identifications of object z , and our goal is to find out the correct identification of z . For achieving this, for each aggregation function G , we obtain a result for each z , $D_k(z) = G_k(A_1(z), A_2(z), \dots, A_n(z))$. Then we associate to z the identification corresponding to the larger value of the aggregation function.

Fuzzy integrals can be viewed as non-linear functions defined with respect to fuzzy measures. In particular, the “ $g\lambda$ -fuzzy measure” introduced by Sugeno [21] can be used to define fuzzy integrals. The ability of fuzzy integrals to combine the results of multiple information sources has been mentioned in previous works.

Definition 1. A function of sets $g: 2^X \rightarrow [0,1]$ is called a fuzzy measure if:

- 1) $g(\emptyset) = 0$ $g(X) = 1$
- 2) $g(A) \leq g(B)$ if $A \subset B$
- 3) if $\{A_i\}_{i=1}^{\infty}$ is a sequence of increments of the measurable set then

$$\lim_{i \rightarrow \infty} g(A_i) = g\left(\lim_{i \rightarrow \infty} A_i\right) \quad (4)$$

From the above it can be deduced that g is not necessarily additive, this property is replaced by the additive property of the conventional measure.

From the general definition of the fuzzy measure, Sugeno introduced what is called “ $g\lambda$ -fuzzy measure”, which satisfies the following additive property: For every $A, B \subset X$ and $A \cap B = \emptyset$,

$$g(A \cup B) = g(A) + g(B) + \lambda g(A)g(B), \quad (5)$$

for some value of $\lambda > -1$.

This property says that the measure of the union of two disjoint sets can be obtained directly from the individual measures. Using the concept of fuzzy measures, Sugeno [21] developed the concept of fuzzy integrals, which are non-linear functions defined with respect to fuzzy measures like the $g\lambda$ -fuzzy measure.

Definition 2 let X be a finite set and $h: X \rightarrow [0,1]$ be a fuzzy subset of X , the fuzzy integral over X of function h with respect to the fuzzy measure g is defined in the following way,

$$\begin{aligned} h(x) \circ g(x) &= \max_{E \subseteq X} [\min_{x \in E} (h(x), g(E))] \quad (6) \\ &= \sup_{\alpha \in [0,1]} [\min(\alpha, g(h_\alpha))] \end{aligned}$$

where h_α is the level set α of h ,

$$h_\alpha = \{x \mid h(x) \geq \alpha\}. \quad (7)$$

We will explain in more detail the above definition: $h(x)$ measures the degree to which concept h is satisfied by x . The term $\min(h_x)$ measures the degree to which concept h is satisfied by all the elements in E . The value $g(E)$ is the degree to which the subset of objects E satisfies the concept measure by g . As a consequence, the obtained value of comparing these two quantities in terms of operator \min indicates the degree to which E satisfies both criteria g and $\min(h_x)$.

V. SIMULATION AND FORECASTING PRICES OF CONSUMER GOODS IN THE U.S. MARKET

We will consider the problem forecasting the prices of tomato in the U.S. market. The time series for the prices of this consumer good shows very complicated dynamic behavior, and for this reason it is interesting to analyze and predict the future prices for this good. We show in Fig. 6 the time series of monthly tomato prices in the period of 1960 to 1999, to give an idea of the complex dynamic behavior of this time series. We will apply both the modular and monolithic neural network approach and also the linear regression method to the problem of forecasting the time series of tomato prices. Then, we will compare the results of these approaches to select the best one for forecasting.

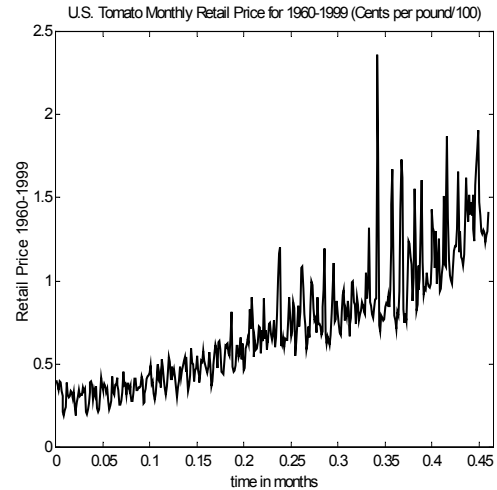


Fig. 6 Prices in US Dollars of tomato from January 1960 to December 1999.

VI. EXPERIMENTAL RESULTS

We describe, in this section, the experimental results obtained by using neural networks to the problem of forecasting tomato prices in the U.S. Market. We show results of the application of several architectures and different learning algorithms to decide on the best one for this problem. We also

compare at the end the results of the neural network approach with the results of linear regression models, to measure the difference in forecasting power of both methodologies. First, we will describe the results of applying modular neural networks to the time series of tomato prices. We used the monthly data from 1960 to 1999 for training a Modular Neural Network with four Modules, each of the modules with 80 neurons and one hidden layer. We show in Fig. 7 the result of training the modular neural network with this data. In Fig. 7, we can appreciate how the modular neural network approximates very well the real time series of tomato prices over the relevant period of time.

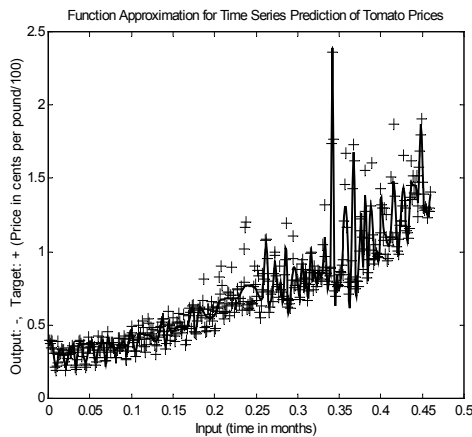


Fig. 7 Modular Neural network for tomato prices with the Levenberg-Marquardt algorithm.

We have to mention that the results shown in Fig. 7 are for the best modular neural network that we were able to find for this problem. We show in Fig. 8 the comparison between several of the modular neural networks that we tried in our experiments. From Fig. 8 we can appreciate that the modular neural network with one time delay and Leverberg-Marquardt (LM) training algorithm is the one that fits best the data and for this reason is the one selected.

We show in Fig. 9 the comparison of the best monolithic network against the best modular neural network. The modular network clearly fits better the real data of the problem.

Now we will show in the following figures the results for forecasting tomato prices in different regions of Mexico. In Fig. 10 we show the forecast for the Sinaloa region in Mexico. In Fig. 11 we show the forecast for the San Luis Potosi region. In Fig. 12 we show the forecast for the Morelos region. In Fig. 13 we show the forecast for the Nayarit region. In Fig. 14 we show the forecast of the Jalisco region. In Fig. 15 the Michoacan region is shown. In Fig. 16 the Sonora region is shown. Finally, we show in Fig. 17 the Puebla region.

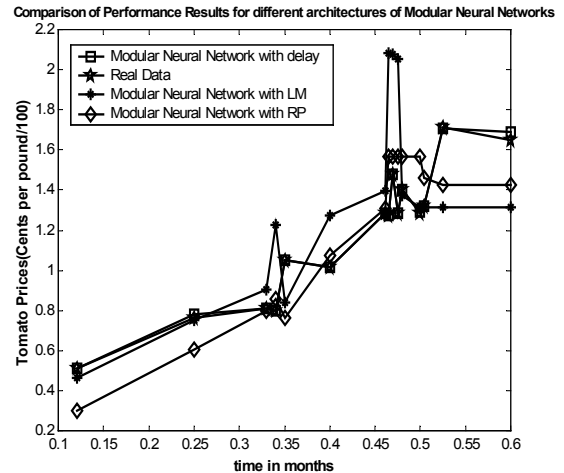


Fig. 8 Comparison of performance results for several modular neural networks.

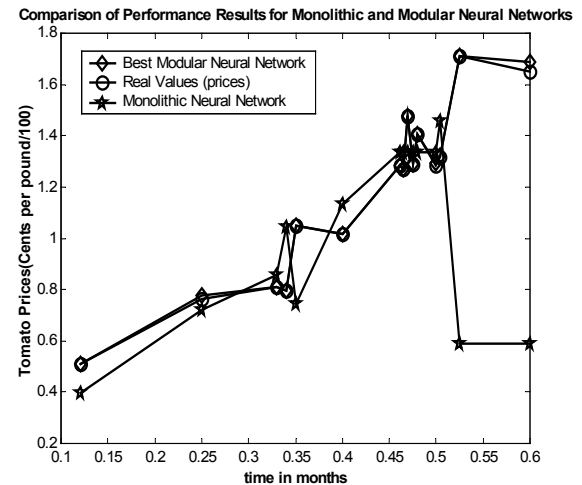


Fig. 9 Comparison of monolithic and modular neural networks.

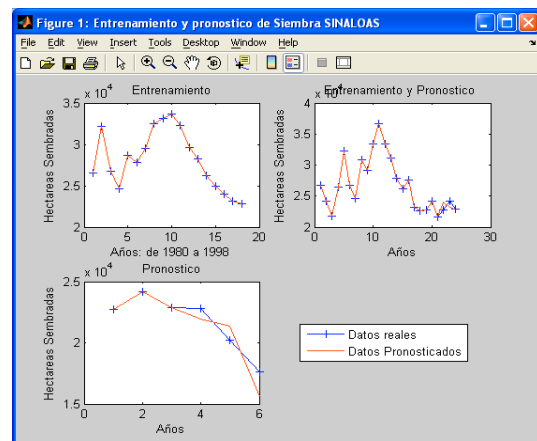


Fig. 10 Forecast for the Sinaloa region in Mexico.

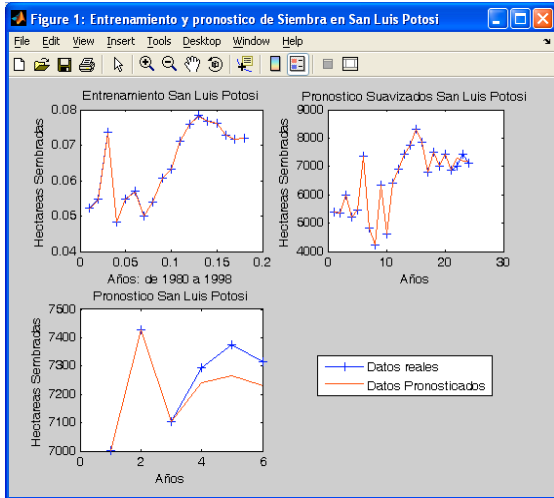


Fig. 11 Forecast for the San Luis Potosi region.

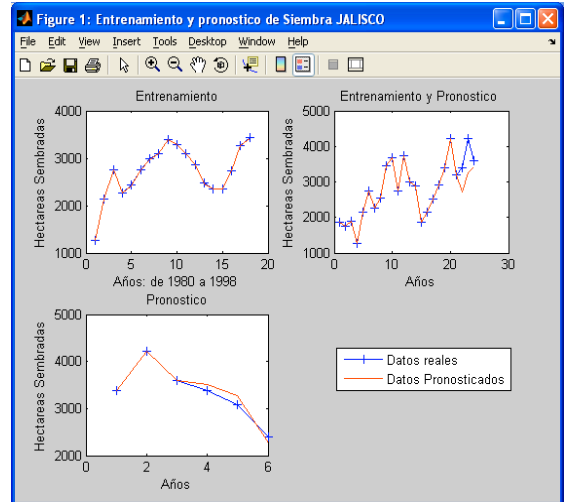


Fig. 14 Forecast for the Jalisco region in Mexico.

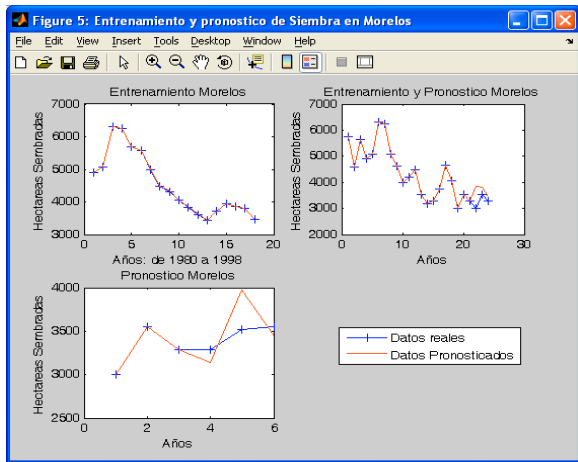


Fig. 12 Forecast for the Morelos region in Mexico.

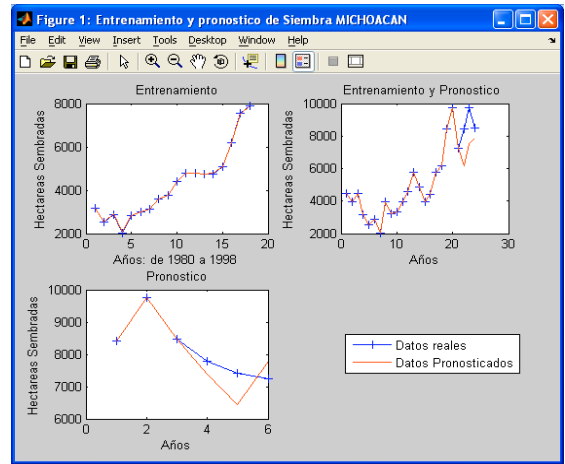


Fig. 15 Forecast for the Michoacan region in Mexico.

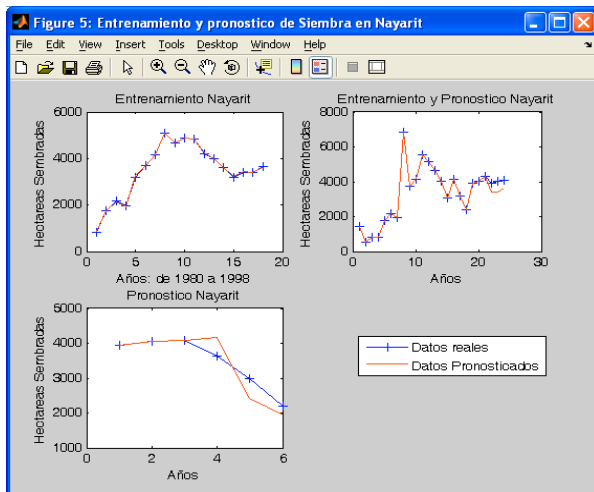


Fig. 13 Forecast for the Nayarit region in Mexico.

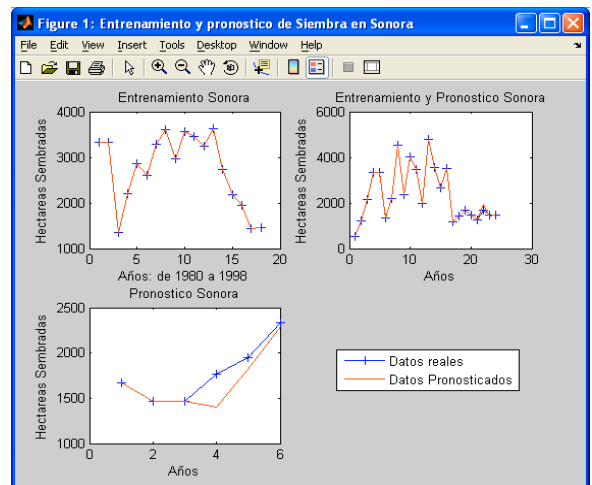


Fig. 16 Forecast for the Sonora region in Mexico.

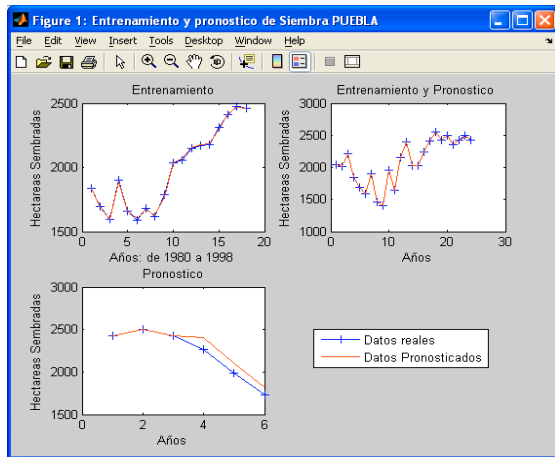


Fig. 17 Forecast for the Puebla region in Mexico.

We can appreciate in the forecasts of Figures 10 to 17 that in all cases the predicted values follow very accurately the real values. In conclusion, the modular neural network approach works very well in this application.

VII. CONCLUSIONS

We described in this paper the use of modular neural networks for simulation and forecasting time series of consumer goods in the U.S. Market. We have considered a real case to test our approach, which is the problem of time series prediction of tomato prices in the U.S. market. We have applied monolithic and modular neural networks with different training algorithms to compare the results and decide which is the best option. The Levenberg-Marquardt learning algorithm gave the best results. The performance of the modular networks was also compared with monolithic networks. The forecasting ability of modular networks was clearly superior than the one of monolithic neural networks, and for this reason modular neural models should be preferred.

ACKNOWLEDGMENTS

This research work was supported in part by CONACYT and SAGARPA of Mexico under Grant 2003-002. The principal author (Patricia Melin) was also supported under the System of National Researchers of Mexico (SNI level II) and the students by scholarships of CONACYT.

REFERENCES

- [1] Boers, E. and Kuiper, H. (1992) Biological Metaphors and the Design of Modular Artificial Neural Networks. Departments of Computer Science and Experimental and Theoretical Psychology at Leiden University, the Netherlands.
- [2] Brock, W.A., Hsieh, D.A., and LeBaron, B. (1991). "Nonlinear Dynamics, Chaos and Instability", MIT Press, Cambridge, USA.

- [3] Castillo, O. and Melin, P. (1996). "Automated Mathematical Modelling for Financial Time Series Prediction using Fuzzy Logic, Dynamical System Theory and Fractal Theory", Proceedings of CIFE'96, IEEE Press, New York, NY, USA, pp. 120-126.
- [4] Castillo, O. and Melin P. (1998). "A New Fuzzy-Genetic Approach for the Simulation and Forecasting of International Trade Non-Linear Dynamics", Proceedings of CIFE'98, IEEE Press, New York, USA, pp. 189-196.
- [5] Castillo, O. and Melin, P. (1999). "Automated Mathematical Modelling for Financial Time Series Prediction Combining Fuzzy Logic and Fractal Theory", Edited Book "Soft Computing for Financial Engineering", Springer-Verlag, Heidelberg, Germany, pp. 93-106.
- [6] O. Castillo and P. Melin, Soft Computing and Fractal Theory for Intelligent Manufacturing. Springer-Verlag, Heidelberg, Germany, 2003.
- [7] Fu, H.-C., Lee, Y.-P., Chiang, C.-C., and Pao, H.-T. (2001). Divide-and-Conquer Learning and Modular Perceptron Networks in IEEE Transaction on Neural Networks, vol. 12, No. 2, pp. 250-263.
- [8] Hansen, L. K. and Salamon P. (1990). Neural Network Ensembles, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No. 10, pp. 993-1001.
- [9] Haykin, S. (1996). "Adaptive Filter Theory", Prentice Hall.
- [10] Jang, J.-S. R., Sun, C.-T., and Mizutani, E. (1997). "Neuro-fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence", Prentice Hall.
- [11] Lu, B. and Ito, M. (1998). Task Decomposition and module combination based on class relations: modular neural network for pattern classification. Technical Report, Nagoya Japan, 1998.
- [12] Maddala, G.S. (1996). "Introduction to Econometrics", Prentice Hall.
- [13] Murray-Smith, R. and Johansen, T. A. (1997). Multiple Model Approaches to Modeling and Control. Taylor and Francis, UK.
- [14] Parker, D.B. (1982). "Learning Logic", Invention Report 581-64, Stanford University.
- [15] Quezada, A. (2004). Reconocimiento de Huellas Digitales Utilizando Redes Neuronales Modulares y Algoritmos Genéticos. Thesis of Computer Science, Tijuana Institute of Technology, Mexico.
- [16] Rasband, S.N. (1990). "Chaotic Dynamics of Non-Linear Systems", Wiley Interscience.
- [17] Ronco, E. and Gawthrop, P. J. (1995). Modular neural networks: A State of the Art. Technical Report, Center for System and Control. University of Glasgow, Glasgow, UK, 1995.
- [18] Rumelhart, D.E., Hinton, G.E., and Williams, R.J. (1986). "Learning Internal Representations by Error Propagation", in "Parallel Distributed Processing: Explorations in the Microstructures of Cognition", MIT Press, Cambridge, MA, USA, Vol. 1, pp. 318-362.
- [19] Schdmit, A. and Bandar, Z. (1997). A Modular Neural Network Architecture with Additional Generalization Abilities for High Dimensional Input Vectors, Proceedings of ICANN'97, Norwich, England.
- [20] Sharkey, A. (1999). Combining Artificial Neural Nets: Ensemble and Modular Multi-Nets Systems, Ed. Springer-Verlag, London, England.
- [21] Sugeno, M. (1974). Theory of fuzzy integrals and its application, Doctoral Thesis, Tokyo Institute of Technology, Japan.
- [22] White, H. (1989). "An Additional Hidden Unit Test for Neglected Non-linearity in Multilayer Feedforward Networks", Proceedings of IJCNN'89, Washington, D.C., IEEE Press, pp. 451-455.
- [23] Yager, R. R. (1999). Criteria Aggregations Functions Using Fuzzy Measures and the Choquet Integral, International Journal of Fuzzy Systems, Vol. 1, No. 2.