Analítika

A Review of Artificial Neural Networks: How Well Do They Perform in Forecasting Time Series?

Una revisión de las Redes Neuronales Artificiales: ¿Cuán bien realizan el pronóstico de series temporales?

Elsy Gómez-Ramos y Francisco Venegas-Martínez

www.inec.gob.ec | www.ecuadorencifras.com

A Review of Artificial Neural Networks: How Well Do They Perform in Forecasting Time Series?

Una revisión de las Redes Neuronales Artificiales: ¿Cuán bien realizan el pronóstico de series temporales?

Elsy Gómez-Ramos† y Francisco Venegas-Martínez‡

Escuela Superior de Economía, Instituto Politécnico Nacional, D. F. México, México

†elsygomez@msn.com, ‡ fvenegas1111@yahoo.com.mx

Recibido: 11 de septiembre de 2012 **Aceptado:** 4 de junio de 2013

Abstract

At the beginning of the 90's, Artificial Neural Networks (ANNs) started their applications in finance. The ANNs are data-drive, self-adaptive and non-linear methods that do not require specific assumptions about the underlying model. In general, there are five groups of networks used as forecasting tools: 1) Feedforward Networks, like the Multilayer Perceptron (MLP), 2) Recurrent Networks, 3) Polynomial Networks, 4) Modular Networks, and 5) Support Vector Machine. This paper carries out a review of the specialized literature on ANNs and makes a comparative analysis according to their performance in forecasting stock indices and exchange rates. The objective is to assess the performance when applying different types of networks in relation to MLP. It is shown that the MLP is the best network in forecasting time series. However, it is shown that the MLP has important delimitations in several respects: network architecture, basic functions and initialization weights.

Keywords: Artificial neural networks, Multilayer Perceptron, Forecasting time series.

Resumen

A principios de la década de los 90, las Redes Neuronales Artificiales (RNAs) comenzaron sus aplicaciones en finanzas. Las redes neuronales son dirigidas por datos, auto-adaptativas y los métodos no lineales que no requieren supuestos específicos sobre el modelo subyacente. En general, hay cinco grupos de redes que se utilizan como herramientas de pronóstico: 1) Redes Feedforward, como el perceptrón multicapa (MLP), 2) Redes recurrentes, 3) Redes polinómicas, 4) Redes modulares, y 5) Apoyo Vector Machine. En este trabajo se realiza una revisión de la literatura especializada sobre las RNA y hace un análisis comparativo de acuerdo a su desempeño en la predicción de índices bursátiles y tipos de cambio. El objetivo es evaluar el rendimiento cuando la aplicación de diferentes tipos de redes en relación con la MLP. Se muestra que la MLP es la mejor red en las series temporales de previsión. Sin embargo, está demostrado que la MLP tiene delimitaciones importantes en varios aspectos: la arquitectura de red, las funciones básicas y los pesos de inicialización.

Palabras clave: Las redes neuronales artificiales, Perceptrón multicapa, series de tiempo Forecasting.

Código JEL: C45, C53, C22.

1 Introduction

The efficient market hypothesis states that stock prices come from a random walk, which implies that the stock returns are not predictable for the public. However, there exists significant empirical evidence that rejects such a hypothesis. For example, there are studies that focus on the persistence and long memory in the volatility of stock markets (Sharth and Medeiros, 2009; Venegas-Martínez and Islas-Camargo, 2005), as well as others that sustain calendar effects (McNelis, 2005). These studies leave the possibility open to predict the behavior of those markets and, surprisingly, the number of research papers supporting the possibility to forecast the prices of this kind of markets is vast and growing.

Traditionally, econometrics has provided a widely range of tools like the GARCH model for forecasting stock prices and exchange rates. However, the rigidity (linear in mean) and the violation of assumptions (non-negativity of the coefficients) of such symmetric models have been discussed in many studies; these models cannot account for leverage effects, although they can account for volatility clustering (volatility appears in groups), leptokurtosis (kurtosis excess), and fat tails (extreme values have a bigger probability than that obtained from the Normal distribution).

The above facts have motivated the use of more flexible models in order to capture in a better way the financial markets behavior (Brooks, 2006; McNelis, 2005). Some of these models come from Artificial Intelligence (AI) that is characterized by its flexibility and capability to integrate different methodologies that somehow try to emulate the biological systems behavior. Within this field, we can find the Artificial Neural Networks (ANNs) that attempt to emulate the human brain functions; see, for instance, Anderson (2007).

There are many potential advantages offered by the ANNs, for instance: i) non-linearity, that is, the neural processor is basically non-linear, ii) input-output mapping, in other words, through supervised learning the network learns according to the examples, iii) adaptability, that is to say, the network has the ability to adapt their synaptic weights even in real time, iv) response capacity, in other words, in the context of pattern classification the network not only provides a pattern selection but also the reliability of decision making, v) fault tolerance due to the massive interconnection, vi) integrated large scale, that is, its parallelism makes it potentially faster for certain tasks and thus capturing complex behaviors, vii) uniformity in the analysis and design, that is to say, the same notation is used in all fields engaged with networks, and viii) neurobiology analogy (Haykin, 1994). In general, the ANNs are data-drive, self-adaptive and non-linear methods that do not require specific assumptions about the underlying model.

Yet, there have been severe criticisms in applications of networks in finance, we may mention, for instant that: a) the estimated coefficients obtained by the network do not have a real interpretation, b) there are no specific tests available in order to consider that a model is adequate, and c) the results are satisfactory inside the sample, but outside the sample are poor (Brooks, 2006). Despite of these critiques, the ANNs have been successfully applied in some specific finance areas. For example, the classification of areas proposed by Mender *et al*. (1996) is based on the decision-making (credit analysis, mortgage risk, project management, investment portfolios, price analysis, and corporate bankruptcy). While in Burrell and Folarin (1997) are mentioned applications in other specific areas (financial analysis, corporate bankruptcy, risk assessment, stock markets forecasting). There is also another available simplified classification in three groups: credit assessment (credit rating, credit risk, and bond pricing), portfolio management (optimal portfolio selection, and portfolio selection) and forecast and planning (predicting corporate bankruptcies; see Bahrammirzae, 2010). Yet, one of the most attractive applications in finance is forecasting financial time series, especially stock indices and exchange rates; in this regard, several investigations consider stock indices and exchange rates as indicator for the future conditions of the economic and financial system.

Since their application in finance in the early 90's, the ANNs have become popular, partly because they are considered as non-parametric models from a statistical point of view. This feature makes them quite flexible in modeling real-world phenomena where observations are generally available, but there is not a theoretical relationship or specification, especially for non-linear functions (Haykin, 1994; Mehrotra *et al*. 2000).

One of the most known networks is the MLP, which is characterized for being a universal approximator and classifier. The construction of the MLP for financial and economic series forecasting is described in Kaastra and Boyd (1996) and Mehrotra *et al*. (2000). Also, the ANNs performance has been compared with traditional models in finance in Burrell and Folarin (1997), Hamid and Iqbal (2004), Khashei and Bijari (2011), McNelis (2005), and Paliwal and Kumar (2009). When performance focuses on the various fields of AI in finance applications, see, for instance, Bahrammirzaee (2010) and Rada (2008).

Unlike previous studies, which analyze the performance of various networks with traditional models in many areas, this research makes a comparison among different types of networks to forecast particularly stock indices (or stocks) and/or the exchange rates. In general, we can identify about five groups of networks used as approximators and/or classifiers: (1) Feedforward Networks, like MLP, (2) Recurrent Networks, (3) Polynomial Networks, (4) Modular Networks, and (5) Support Vector Machine. In this paper, we shall analyze several research works that apply the MLP and other type of network for the stock indices and/or the exchange rate. The objective is to assess the performance when applying different types of networks in relation to MLP.

The research is organized as follows. In section 2, we introduce the MLP. In section 3, we present an overview of each networks group. In section 4, we carry out a comparative analysis of the ANNs. In section 5, we assess the performances of ANN in terms of their result in applications taking as a benchmark the MLP. In section 6, we present an application. Finally, in section 7, we conclude.

2 The Multilayer Perceptron

The neuron (or node) is the basic unit of a neural network. In the case of the MLP, it includes an input layer (that does not do any processing), one output layer and at least one hidden layer. The layers consist of a set of nodes; in the case of the hidden layer its inputs come from units in the previous layer and send its outputs to the next layer. The input and output layers indicate the flow of information during the training phase where the learning algorithm is implemented. The MLP generally learns by means of a backpropagation algorithm, which is basically a gradient technique. It has also been implemented variants of the algorithm to work on the problem of slow convergence (for example, momentum term, see Haykin, 1994). Once the trained process is carried out, the network weights are frozen and can be used to compute output values for new input samples. In what follows, we provide a brief explanation of the backpropagation algorithm.

The network learning is a process in which the weights, w, are adapted by a continuous interaction (k) with the environment, in such a way that

$$w_{nj}(k+1) = w_{nj}(k) + \Delta w_{nj}(k)$$

where $w(k)$ is the previous value of the weight vector and $w(k+1)$ is the updated value. The learning algorithm is a set of rules to solve the learning problem and determine the values $w_{nj}(k)$.

One of the most important algorithms is that of the error correction. Consider the n-th neuron in the iteration. Let y_n be the response of this neuron; $x(k)$ is the vector of environment stimuli, and $\{x(k), d_n(k)\}$ is the pair of training. Define the following error signal equation:

$$e_n(k) = d_n(k) - y_n(k)$$

The objective is to minimize the cost function (criterion) which takes into account this error. After selecting the criteria, the problem of error correction learning becomes one of optimization. Consider a function $\epsilon(w)$, which is a continuously differentiable function of a weight vector. The function $\epsilon(w)$ transforms the elements from w to real numbers. We need to find an optimal solution w^* that satisfies the condition:

$$\epsilon(w^*) \leq \epsilon(w).$$

Then it is necessary to solve an optimization problem without constraints posed as: the cost function minimization $e(w)$ with respect to the weight *vector*. The necessary condition for optimality is given by:

$$\nabla\epsilon(w^*) = 0$$

where ∇ is the gradient operator. An important class of optimization algorithms without constraints is based on the idea of iterative descent (gradient descent method and Newton's method). Starting with an initial condition $w(0)$, it generates a sequence $w(1),\ w(2), \ldots$, such that the cost function $\epsilon(w)$ decreases in every algorithm iteration. It is desirable that the algorithm eventually converge in to the optimal solution in such a way that

$$\epsilon(w(k+1)) < \epsilon(w(k))$$

In the descent gradient method, the successive adjustments are applied to the weight vector, in the direction of the gradient descent. For convenience, we will use the following notation:

$$g = \nabla\epsilon(w).$$

The gradient descent algorithm can be written formally as:

$$w(k+1) = w(k) - \eta g(k)$$

where η is a positive constant called the learning rate, and $g(k)$ is the gradient vector evaluated at $w(k)$. Therefore, the correction applied to the weight vector can be written as:

$$\Delta w(k) = w(k+1) - w(k) = -\eta g(k).$$

This method converges slowly to an optimal solution w^*. However, the learning rate has a larger impact on this convergent behavior. When η is small, the path of $w(k)$, over the plane W is smooth. When η is large, the path of $w(k)$ over the plane W is oscillatory, and when η exceeds a certain critical value, the path $w(k)$ over the plane W becomes unstable. Thus, the backpropagation algorithm is a technique to implement the method of descent gradient in a weight space for a multilayer network. The basic idea is to efficiently calculate the partial derivatives of an approximate function of the behavior by the neural network with respect to all the elements of the adjustable vector of parameters w for a given value of the input vector x.

3 Types of ANNs

The specialized literature identifies several groups of networks used as approximators and/or classifiers. This section provides a classification in terms of the general characteristics of the ANNs.

1. In the first group, we can find the Feedforward Networks (FFNs), like MLP. Its main feature is that their connection is forward so they do not establish any connections between the nodes on the same layer or previous nodes. The networks that share this feature are: The Radial Basic Function (RBF) (Bildirici *et al.* 2010; Dhamija and Bhalla, 2011; Cheng, 1996); the Generalized Regression Neural Network (GRNN) (Enke and Thawornwong, 2005; Mostafa, 2010); the Group Method of Data Handling Network (GMDHN) (Pham and Lui, 1995); the Probabilistic Neural Network (PNN) (Enke and Thawornwong, 2005; Thawornwong and Enke, 2004); the Dynamic Neural Network (DNN) (Guresen, Kayakutlu and Daim, 2010) and the Cerebellar Model Articulation Controller (CMAC) (Chen, 1996).

2. In the second group, we can locate the Recurrent Networks (RCNs) that are characterized by the dynamism of their connectivity, so these networks stores information that will be used later. The networks that share this feature are: Elman Network (ELN) (Kuan and Liu, 1995; Selvaratnam and Kirley, 2006; Sitte, 2002; Yumlu *et al*. 2005); the modifications to Elman network (Kodogiannis and Lolis, 2002; Perez-Rodriguez *et al*. 2005); Partially Recurrent Networks (PRN) (Kodogiannis and Lolis, 2002; Perez-Rodriguez *et al*. 2005) and Autoregressive Networks (ARN) (Kodogiannis and Lolis, 2002).

3. Through the third group, we can find the Polynomial Networks (PLNs) which typically offer efficient processing of polynomial input variables, otherwise if we would apply the sigmoidal or gaussian functions in the training, although it would be exhaustive. The networks that share this feature are: Pi-sigma networks such as Ridge Polynomial Networks (RPN) and its dynamic version (Ghazali *et al*. 2007, 2009 and 2011), as well as the Function Link Network (FLN) (Hussain *et al*. 2008).

4. In the fourth group are the Modular Networks (MNs) that consists of various modules (networks) which allow solving tasks separately and then combining the answers in a logical manner. One possibility is to use different network architectures (Zhang and Berardi, 2001) and another alternative is to apply different initialization weights leaving the same network architectures (Adeodato *et al*. 2011; Zhang and Berardi, 2001).

5. Through the fifth group, we can find the Support Vector Machine (SVM), this network belongs to the kernel base models or nucleus. The idea is to construct a hyperplane as a decision surface which maximizes the margin of separation (Carpinteiro *et al*. 2011; Kara *et al*. 2011; Shen *et al*. 2011).

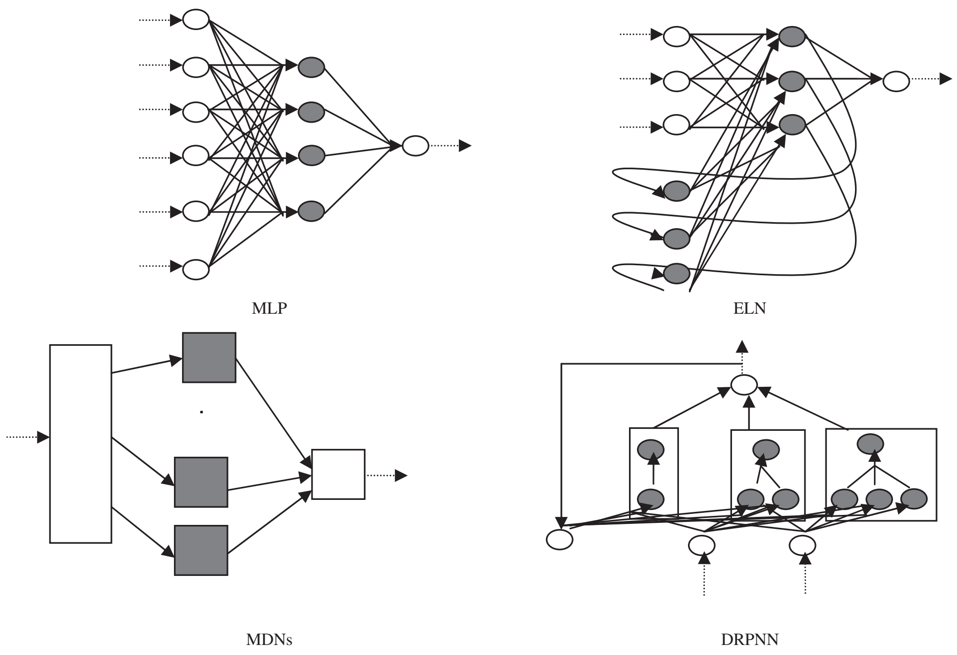

Figure 1. Some neural networks applied in stock market and exchange rate forecasting. Source: author elaboration.

4 A comparative Analysis of ANN

In this section is exposed the main characteristics of the ANNs. All these characteristics (o properties) take as a reference point the MLP:

a) Because MLP does not have a dynamic structure, RCNs are proposed as an alternative. Therefore, the ELN could have a better performance than that of the MLP (Perez-Rodriguez *et al*. 2005; Selvaratnam and Kirley, 2006; Sitte, 2002). However, in the ELN all nodes are connected to other nodes can make the training difficult, another proposal is the PARN (Kodogiannis and Lolis, 2002; Perez-Rodriguez *et al*. 2005) or the ARN in which case a more efficient training is expected (Kiani and Kastens, 2008; Kodogiannis and Lolis, 2002) this is so because the nodes are connected by themselves.

b) To find the best network is usually based on trial and error criteria, that is why this kind of methods waste information and time (this usually reflects in unstable forecasts) so using MDNs with different sizes of networks would avoid these selection process (Adeodato *et al*. 2011; Zang and Berardi, 2001). Another proposal is to apply the GMDHN which increases in size during the training (Pham and Lui, 1995), or apply the DNN, which increases the number of hidden layers dynamically (Guresen, Kayakutlu and Daim, 2010).

c) The sample is typically divided into two stages (training and testing). Furthermore, it is necessary to establish the number of nodes in the hidden layer before starting the training stage. An alternative to overcome these problems is to apply the GRNN, because it does not require estimating the number of nodes in the hidden layer and all the available information can be used for the network training, therefore no early stopping technique is required during its training (Enke and Thawornwong, 2005; Leung *et al*. 2000; Mostafa, 2010). Also the DNN or the PRNN do not need training or early stopping techniques (Enke and Thawornwong, 2005; Thawornwong and Enke, 2004).

d) In the MLP the processing nodes are located in the hidden and output layers sharing the same type of processor (using it as a classifier, the processing nodes are non-linear, but as an approximator the output node is linear), while in the RBF the nodes in the hidden layer have certain properties that help to different learning purposes, which could provide a more accurate forecast (Hutchinson, Lo and Poggio, 1994), or the SVM, in which the choice of kernel function is a critical decision for prediction efficiency (Kara *et al*. 2011).

e) During the training all weights are modified, and therefore, learning is slow. In contrast, RPNN have only one set of weights in the layer to train, which facilitates the learning process (Ghazali *et al*. 2007, 2009 and 2011). Another possibility could be the FLN (Hussain *et al*. 2008) or the Partially Connected Network (PCN), which selects the connection between nodes randomly (Chang *et al*. 2012).

f) In the MLP, it is necessary to define the range in the initialization weights (usually very small) however, there is no consensus for specific applications so it is usually chosen at the designer's discretion or having as a reference similar applications. An alternative to this limitation is to apply MDNs that share the same network architecture but different ranges in the initialization weights (Adeodato *et al*. 2011; Zang and Berardi, 2001).

g) The MLP employs an algorithm which is basically a gradient technique. It implies that the problem is nonconvex and its solution is a local minimum. SVM uses the structural risk minimization theory so the problem is a convex optimization problem, which means that the optimal solution is global (Ince and Trafalis, 2006).

h) The MLP is characterized by more learning interference for inputs distant from any training vector. A solution for this problem is to use CMACN, which can get one-step learning where MLP cannot (Lu and Wu, 2011).

5 Applications and Performance of ANN

In what follows, Table 1 summarizes the information according to the previous classification. On the basis of the obtained results in thirty reviewed papers, we observe that more than 40% of the analyzed researches support the idea that the MLP is the best network or at least it has the same performance with respect to the proposal networks. With regard to the investigations that were in favor of other models (e.g. econometrics models) (Kodogiannis and Lolis, 2002; Yumlu and Gurense, 2005), we excluded them, and analyzed only the performance of the different proposed networks, but none of the cases the MLP stands.

The main idea of this review is to point out the advantages and delimitations of the MLP with respect to other available networks by comparing not only the learning process, but also the architecture design. The issue of the type of connections between the nodes (like RCNs suggest) could not be so successful in several applications. The main drawback associated with RCNs is that they need more time to learn than the standard networks because their outputs pass through the network more than once (depending on the type of the RCNs) before the final output. Another issue is if apply or not in a network some optimization technique, like in SVM, instead of gradient technique.

The types of networks that have shown superiority over the MLP are the RBF, GRNN, MDNs and DRPNN. Both RBF and GRNN are FFNs. In these types of networks the training may be in terms of global or local basis functions. The MLP applies a global basic function (usually sigmoidal), and this function have non-negligible values throughout all measurement space, so many iterations are required to find a combination that has an acceptable error in all parts of the measurement space for which training data are available. On the other hand, GRNN and RBF are based on a localized basic function, which provides an important advantage of instant learning. The GRNN is based on the estimation of probability density functions, and RBF is based on iterative function approximation. Although, PRNN and GRNN are based on the estimation of probability density functions, the reason that GRNN and RBF outperform in compared with PRNN could be the used of the regression method (Chen, 1996).

The fact that the MDNs perform better than the MLP is because they are more precise techniques for the initialization weights, but when MDNs mix different architectures sharing the weights range, the result is poor. In the case of the DRPNN with respect to MLP, they have only a single layer of learnable weights, so it will reduce the network complexity. Therefore, PLNs are appropriated when the number of inputs to the model and the training becomes extremely large, so the training procedure for ordinary networks like MLP becomes very slow. The fact that some dynamic versions succeed (although it implies more connections) is because their architecture is very simple. Another case (isolated) is the CMAC that performs better than MLP or RBF, so MLP cannot elude the problem of slow learning.

Table 1. Application of different networks for the stock market and the exchange rate. Source: author elaboration.

Year	Author(s)	ANNs							Results[a]
		1			2	3	4	5	
		MLP	RBF	Others	RCNs	PLNs	MDNs	SVM	
Exchange rate									
2011	Dhamija & Bhalla	×	×						RBF performs better
2010	Bildirici *et al.*	×	×		×				RBF performs better
2009	Ghazali *et al.*	×				×			PLNs & MLP perform better
2008	Kiani & Kastens	×			×				RCNs perform better
2008	Hussain *et al.*	×				×			Same performance
2007	Ghazali *et al.*	×				×			PLNs performs better
2007	Portela *et al.*	×	×						MLP performs better
2006	Ince & Trafalis	×						×	SVM performs better
2002	Kodogiannis & Lolis	×	×		×				RCNs perform better
2001	Zhang & Berardi	×					×		MDNs perform better
2000	Leung *et al.*	×		×					Others performs better
1995	Kuan & Liu	×			×				Small differences
1994	Pham & Liu	×		×	×				MLP & others perform better
Stock market									
2012	Chang *et al.*	×		×					Others perform better
2011	Shen *et al.*	×	×					×	RBF performs better
2011	Lu & Wu	×	×	×				×	Others perform better
2011	Kara *et al.*	×						×	MLP performs better
2011	Carpinteiro *et al.*	×						×	SVM performs better
2011	Guresen *et al.*	×		×					MLP performs better
2010	Mostafa	×		×					Others performs better
2005	Enke & Thawornwong	×		×					MLP performs better
2006	Selvaratnam & Kirley	×			×				Same performance
2005	Yumlu *et al.*	×			×				RCNs perform better
2005	Pérez *et al.*	×			×				MLP performs better
2004	Thawornwong & Enke	×		×					MLP performs better
2002	Sitte	×			×				Same performance
2000	Leung *et al.*	×		×					Others perform better
Both									
2011	Adeodato *et al.*	×					×		MDNs perform better
2011	Ghazali *et al.*	×				×			PLNs performs better
1994	Hutchinson, Lo, & Poggio	×	×						MLP & RBF perform better

[a]The results are based on the author's criterion for the multilag forecast

The above results suggest that, in general, the basic functions, the initialization weights, and the network architecture will produce a path that concentrates great efforts. These issues have already been established by other authors who seek through intelligent methods (e.g. genetic algorithms) to improve the performance of the MLP, especially with the network architecture and initialization weights (Hansen and Nelson, 2003; Karathanasopoulos *et al*. 2010). With respect to the basic function, the literature in finance does not discuss this task with respect to GA or another methodology (or we did not find it). As a conclusion, we should not suggest complicated networks, in some cases, the simplicity is the best.

6 The IBM case

In this section we apply the MLP to the IBM case. To do this, we will focus on the IBM daily common stock returns from January 02, 2003 to May 05, 2013 (see Figure 2). Data consisting of 2592 days will be used for training and the last 10 days for testing (or forecast period). The software to be used is *Mathematica 6.0*.

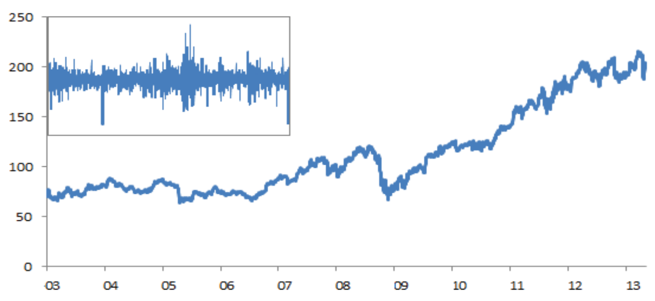

Figure 2. IBM daily closing prices and returns. Source: Economatica.

Step 1: Parameter values. The optimal values of the parameters depend on the application and they are not easy to determine *a priori*. In this case, they are chosen according to similar studies $\eta = 0.3$ and the momentum term $= 0.2$ (Pérez-Rodríguez *et al*. 2005; Theofilatos *et al*. 2010).

Step 2: Size of the training. According to the desired accuracy on the test set, it is suggested that

$$P \geq \frac{|w|}{1-a} \log \frac{n}{1-a}$$

where P denotes the size of the training set, $|w|$ denotes the numbers of weights to be trained, a stands for the expected accuracy on the test (in our case, the value is given by 0.95), and is the number of nodes (Mehrotra *et al*. 2000).

Step 3: We now analyze the error among the different network architectures. The empirical evidence suggests that when the size of the networks is increasing, the training performance improves; however, if the one-lag and multilag performances deteriorate, the networks are oversized. The process may stop (e.g. establish an acceptable error) or continue until it is found the "best" network. If the last procedure is chosen, we can compare the training error with the forecast error (one-lag testing + multilag testing) and choose the network just before the forecast error increases according to the training error.

In Table 2 is presented the mean squared error from different network architectures. The first column contains the training error which decreases when the size of the network increases. The last column contains the forecast error which is always bigger than the training error but at some point the forecast error begins to have a greater difference in relation to the training error, which is shown in Figure 3. At this point the training error continues decreasing because the network is larger than required so the network memorize information and gradually lose the ability to respond to new information (forecast error). According to the obtained results the "best" network architectures is 6-8-1 (the series is roughly explained from the lag number 6 with 8 hidden layers) with a training error of .000486 and a forecast error of .000507 while the remainder of the results are discarded (a lot of time and information is wasted).

Table 2. Mean Squared Errors. Source: author elaboration.

Network architectures[a]		Training	1 One-lag Testing	2 Multilag Testing	1+2 Forecast error
3	2-1	.006402	.007236	.005764	0.006500
	3-1	.004078	.004721	.003574	0.004148
	4-1	.002949	.003478	.002522	0.003000
	5-1	.002322	.002794	.001945	0.002370
	6-1	.001907	.002342	.001567	0.001955
	7-1	.001619	.002010	.001306	0.001658
4	6-1	.001457	.001838	.001161	0.001500
	7-1	.001243	.001591	.000971	0.001281
5	6-1	.001182	.001520	.000917	0.001219
	7-1	.000735	.000996	.000529	0.000763
6	**8-1**	**.000486**	**.000692**	**.000322**	**0.000507**
7	9-1	.000336	.000505	.000321	0.000413
	10-1	.000268	.000414	.000319	0.000367
8	10-1	.000254	.000397	.000308	0.000353
10	9-1	.000250	.000391	.000276	0.000334
	10-1	.000189	.000307	.000264	0.000286
	11-1	.000165	.000274	.000235	0.000255

[a]Input-hidden-output nodes

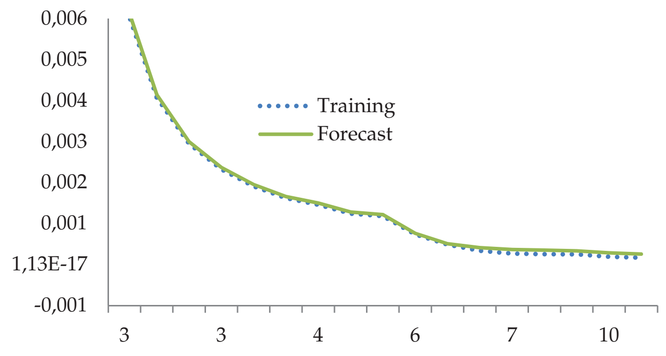

Figure 3. Training and forecast errors. Source: author elaboration.

7 Conclusions

This paper carried out an exhaustive review of the specialized literature on ANNs and made a comparative analysis according to their performances in forecasting stock indices (or stocks) and exchange rates. In this regard, it is important to point out that the MLP is one of the most used networks in finance, because it is a feedforward multilayer network with non-linear node functions. In order to support this, we have reviewed thirty applications in the literature. We found that more than 40% of the analyzed researches support the idea that the MLP is the best network or at least it has the same performance with respect to the proposal networks. However, it is shown that the MLP has important delimitations in several respects: network architecture, basic functions and initialization weights. One way to improve the performance of the MLP is to apply intelligent methods. As a result we get a hybrid network which is expected to provide a more accurate forecast.

References

Adeodato, P., A. Arnaud, G. Vasconcelos, R. Cunha y D. Monteiro. 2011. *Mlp ensembles improve long term prediction accuracy over single networks*. International Journal of Forecasting, 27(3): 661–671.

Anderson, J. A. 2007. *An Introduction to Neural Networks*. The IMT Press.

Bahrammirzaee, A. 2010. *A comparative survey of artificial intelligence applications in finance: Artificial neural networks, expert system and hybrid intelligent systems.* Neural Computing and Applications, 19(8): 1165–1195.

Bildirici, M., E. Alp y O. Ersin. 2010. *Tar-cointegration neural network model: An empirical analysis of exchange rates and stock return.* Expert Systems with Applications, 37(1): 57–69.

Brooks, C. 2006. *Introductory Econometrics for Finance*. Cambridge University Press, 7 edición.

Burrell, P. y B. Folarin. 1997. *The impact of neural networks in finance.* Neural Computing and Applications, 6(4): 193–200.

Carpinteiro, O., J. Leite, C. Pinheiro y I. Lima. 2011. *Forecasting Models for Prediction in Time Series. Artificial Intelligence Review.* Springer.

Chang, P., D. Wang y Z. D. 2012. *A novel model by evolving partially connected neural network for stock price trend forecasting*. Expert Systems with Applications, 39(1): 1–15.

Cheng, C. 1996. *Fuzzy Logic and Neural Network Handbook*. IEEE Press.

Dhamija, A. y V. Bhalla. 2011. *Exchange rate forecasting: Comparison of various architectures of neural networks.* Neural Computation and Applications, 20(3): 355–363.

Enke, D. y S. Thawornwong. 2005. *The use of data mining and neural networks for forecasting stock market returns*. Expert Systems with Applications, 29(4): 927–940.

Ghazali, R., A. J. Hussain, D. Al-Jumeily y M. Merabti. 2007. *Dynamic Ridge Polynomial Neural Networks in Exchange Rates Time Series Forecasting*. ICANNGA.

Ghazali, R., A. J. Hussain y P. Liatsis. 2011. *Dynamic ridge polynomial neural network: Forecasting the univariate non-stationary and stationary trading signals*. Expert Systems with Applications, 38(4): 3765–3776.

Ghazali, R., A. J. Hussain, N. M. Nawi y B. Mohamad. 2009. *Non-stationary and stationary prediction of financial time series using dynamic ridge polynomial neural network*. Neurocomputing, 72(10-12): 2359–2367.

Guresen, E., G. Kayakutlu y T. U. Daim. 2010. *Using artificial neural network models in stock market index prediction*. Expert Systems with Applications, 38(8): 10389–10397.

Hamid, Z., S. A. y Iqbal. 2004. *Using neural networks for forecasting volatility of s&p 500 index futures prices*. Journal of Business Research, 57(10): 1116–1125.

Hansen, J. V. y R. D. Nelson. 2003. *Forecasting and recombining time-series components by using neural networks*. Journal of the Operational Research Society, 54(3): 307–317.

Haykin, S. 1994. *Neural Networks a Comprehensive Foundation*. IEEE Computer Society Press.

Hussain, A., J. A. Anderson, A. Knowles, P. J. G. Lisboa y W. El-Deredy. 2008. *Financial time series prediction using polynomial pipelined neural networks*. Expert Systems with Applications, 35(3): 1186–1199.

Hutchinson, J. M., A. W. Lo y T. Poggio. 1994. *A nonparametric approach to pricing and hedging derivative securities via learning networks*. Journal of Finance, 49(3): 851–889.

Ince, H. y T. B. Trafalis. 2006. *A hybrid model for exchange rate prediction*. Decision Support Systems, 42(2): 1054–1062.

Kaastra, I. y M. Boyd. 1996. *Designing a neural network for forecasting financial and economic time series*. Neurocomputing, 10(3): 215–236.

Kara, Y., M. A. Boyacioglu y O. K. Baykan. 2011. *Predicting direction of stock price index movement using artificial neural networks and support vector machines: the sample of the istanbul stock exchange*. Expert Systems with Applications, 38(5): 5311–5319.

Karathanasopoulos, A., K. Theofilatos, P. Leloudas y S. Likothanassis. 2010. *Modeling The ASE 20 Greek Index Using Artificial Neural Networks Combined With Genetic Algorithms*. 6352, Artificial Neural Networks, Lecture Note in Computer Science.

Khashei, M. y M. Bijari. 2011. *A novel hybridization of artificial neural networks and arima models for time series forecasting*. Applied Soft Computing Journal, 11(2): 2664–2675.

Kiani, K. M. y T. L. Kastens. 2008. *Testing forecast accuracy of foreign exchange rate: Predictions from feedforward and various recurrent neural network architecture*. Computational Economics, 32(4): 383–406.

Kodogiannis, V. y A. Lolis. 2002. *Forecasting financial time series using neural network and fuzzy system-based techniques*. Neural Computing and Applications, 11: 90–102.

Kuan, C. M. y T. Liu. 1995. *Forecasting exchange rate using feedforward and recurrent neural networks*. Journal of Applied Econometrics, 10(4): 347–364.

Leung, M., A. Chen y H. Daouk. 2000a. *Forecasting exchange rate using general regression neural networks*. Computers and Operations Research, 27(11): 1093–1110.

Leung, M. T., H. Daouk y A. S. Chen. 2000b. *Forecasting stock indices: A comparison of classification and level estimation models*. International Journal of Forecasting, 16(2): 173–190.

Lu, C. y J. Wu. 2011. *An efficient cmac neural network for stock index forecasting*. Expert Systems with Applications, 38(12): 14357–15598.

McNelis, P. D. 2005. *Neural Networks in Finance: Gaining Predictive Edge in the Market*. Elsevier Academic Press, UK.

Medsker, L., E. Turba y R. R. Trippi. 1996. *Neural Networks in Finance and Investing: Using Artificial Intelligence to Improve Real-World Performance*. Neural Networks Fundamentals for Financial Analysts.

Mehrotra, K., C. K. Mohan y S. Ranka. 2000. *Elements of Artificial Neural Networks*. The MIT Press, USA.

Mostafa, M. M. 2010. *Forecasting stock exchange movements using neural networks: Empirical evidence from kuwai*. Expert systems with applications, 37(9): 6302–6309.

Paliwal, M. y U. A. Kumar. 2009. *Neural networks and statistical techniques: A review of application*. Expert Systems with Applications, 36(1): 2–17.

Perez-Rodriguez, J. V., S. Torra y J. Andrada-Felix. 2005. *Star and ann models: Forecasting performance on the spanish ibex-35 stock index*. Journal of Empirical Finance, 12(3): 490–509.

Pham, D. T. y X. Lui. 1995. *Neural Networks for Identification, Prediction and Control*. Springer, USA.

Portela, A., N. Affonso da Costa y L. Coelho. 2007. *Computational intelligence approaches and linear models in case studies of forecasting exchange rate*. Expert Systems with Applications, 33(4): 816–823.

Rada, R. 2008. *Expert systems and evolutionary computing for financial investing: A review*. Expert Systems with Applications, 34(4): 2232–2240.

Selvaratnam, S. y M. Kirley. 2006. *Predicting Stock Market Time Series Using Evolutionary Artificial Neural Networks with Hurst Exponent Input Window. Lecture Notes in Artificial Intelligence*. Springer.

Sharth, M. y M. C. Medeiros. 2009. *Asymmetric effects and long memory in the volatility of dow jones stock*. International Journal of Forecasting, 25(2): 198–212.

Shen, W., X. Guo, C. Wu y D. Wu. 2011. *Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm*. Knowledge-Based Systems, 24(3): 427–443.

Sitte, R. y J. Sitte. 2002. *Neural networks approach to the random walk dilemma of financial time series*. Applied Intelligence, 3(16): 163–171.

Thawornwong, S. y D. Enke. 2004. *The adaptive selection of financial and economic variables for use with artificial neural networks*. Neurocomputing, 56: 205–232.

Venegas-Martínez, F. y A. Islas-Camargo. 2005. *Volatilidad de los mercados bursátiles de américa latina: efectos de largo plazo*. Comercio Exterior, 55(1): 936–947.

Yumlu, S., F. S. Gurgen y N. Okay. 2005. *A comparison of global, recurrent and smoothed-piecewise neural models for istanbul stock exchange (ise) prediction*. Pattern Recognition Letter, 26(13): 2093–2103.

Zhang, G. P. y V. L. Berardi. 2001. *Time series forecasting with neural network ensembles: An application for exchange rate prediction*. Journal of the Operational Research Society, 52(6): 652–664.