Istanbul University Journal of the School of Business
İstanbul Üniversitesi İşletme Fakültesi Dergisi

Vol/Cilt: 46, No/Sayı:2, November/Kasım 2017, 147-160
ISSN: 1303-1732 – http://dergipark.ulakbim.gov.tr/iuisletme
DOI: 10.5152/iujsb.2017.001

Comparing and Combining MLP and NEAT for Time Series Forecasting

Serkan Aras[a, ✦], Anh Nguyen[b], Allan White[b], Shan He[b]

ARTICLE INFO
Yayın Bilgisi

Received/Başvuru:
17.08.2016

Revision Requested/ Revizyon Talebi:
14.03.2017

Revision Submitted/ Revizyondan Geliş:
17.09.2017

Accepted/Kabul:
17.09.2017

Keywords:
Forecasting
Time Series
NEAT
MLP
Combined Forecast
Neural Networks

ABSTRACT

Neural networks are one of the widely-used time series forecasting methods in time series applications. Among different neural network architectures and learning algorithms, the most popular choice is the feedforward Multilayer Perceptron (MLP). However, it suffers from some drawbacks such as getting trapped in local minima, human intervention during the stage of training, and limitations in architecture design. The aims of this study were twofold. The first was to employ NeuroEvolution of Augmenting Topologies (NEAT), which has many successful applications in numerous fields. In this paper, we applied it to time series forecasting for the first time and compared its performance with that of the MLP. The second aim was to analyse the performance resulting from the pairwise combination of these methods. In general, the results suggested that the forecasts from the NEAT algorithm were more accurate than those of the MLP. The results also showed that pairwise combined forecasts in general were better than single forecasts. The best forecasts of all were obtained by pairwise combination of MLP and NEAT.

1. Introduction

Time-series forecasting is an important and consistently growing area that includes many different areas of scientific, industrial, commercial and economic activity. Linear forecasting methods such as moving average, exponential smoothing and Box-Jenkins techniques had dominated the field of time series analysis and applied to numerous real-world applications for a long time. However, many real-world problems in fact exhibit nonlinear characteristics. In such cases, using nonlinear forecasting techniques would be reasonable. Hence,

✦ Corresponding author. Tel: +90 555 254 37 28, e-mail: serkan.aras@deu.edu.tr
e-mail addresses: serkan.aras@deu.edu.tr (S. Aras), ngducanh2009@gmail.com (A. Nguyen), apwhite@yahoo.co.uk (A. White), s.he@cs.bham.ac.uk (S. He)
[a]*Department of Econometrics, Dokuz Eylül University Faculty of Economics and Administrative Sciences, İzmir, Turkey*
[b]*University of Birmingham School of Computer Science, Birmingham, United Kingdom*

many nonlinear forecasting methods have been developed e.g., bilinear, threshold autoregressive (TAR), smooth-transition autoregressive (STAR), autoregressive conditional heteroscedasticity (ARCH) and generalised autoregressive conditional heteroscedasticity (GARCH) models in attempts to capture nonlinear patterns in the data (Enders, 2010). The problem with these model-based methods is the need to specify the nonlinear functional relationship between the variables. This property of model-based nonlinear techniques leads to their application being limited to specific problems (De Goojier and Kumar, 1992).

The need for flexible nonlinear modelling to overcome the necessity of specifying nonlinear functions before fitting the data in time series forecasting has led some modellers to focus on neural networks because of their valuable properties. Neural networks originated as a result of an attempt to find the answer to the question of how human brains process information and trying to mimic the behaviour of real neurons. The main feature of neural networks consists of the ability to perform parallel data processing using many distributed nodes over the net. Some of the properties of neural networks make them very appealing in the modelling process compared with traditional time series forecasting techniques. First of all, they are data-driven, which means that there is no assumed function of the relationship between the variables being modelled. From a theoretical point of view, it has been proved that if there are sufficient hidden nodes in a neural network, it can converge to any functional form to any desired level of precision (Cybenko, 1989; Hornik et al., 1989). Also, the large number of successful applications of neural networks to time series forecasting problems provides evidence for them being a valuable tool in the modeller's toolkit (Zhang et al., 1998). Neural networks learn from past examples and the only thing needed is to present the relevant data relating to the problem in hand. This situation is very useful for many of the problems in which obtaining data is easier than predicting the data generation process of the system theoretically.

Evolutionary Artificial Neural Networks (EANN) is the family of biologically-inspired computational models which use evolutionary algorithms to optimise the architecture or weights or some other important parameters of neural networks or which use joint evolution of the weights and architecture simultaneously. A large number of studies have been carried out on this topic and more information can be found in Yao's review paper (1999). One interesting paper dealing with the joint evolution of weights and architecture of neural networks is reported by Stanley and Miikkulainen (2002). They proposed the NEAT algorithm which attempts to gain an advantage from evolving neural network topologies together with the weights. In their study, they showed that the NEAT algorithm can handle difficult control task problems very well, compared with other EANN techniques. However, according to our knowledge, there is no direct application of the NEAT algorithm in time series forecasting up to the present time. Thus, we hope that this study will form an example of the forecasting capability of NEAT in the time series forecasting context and prepare the way for future work on this topic.

After the first published papers on the use of multiple forecasts (Reid, 1968; Bates and Granger, 1969) to improve forecast accuracy by using individual methods in combination, the research area of combined forecasts has come to be of interest to researchers from every forecasting field. Clemen (1989) has presented a comprehensive review and annotated bibliography of this area. It is a fact accepted by almost everyone working in the forecasting field that no single forecasting method is the best for every modelling task application (Makridakis et al., 1982). The expectation of combining models is that a composite forecasting method will be better at capturing different patterns in data by means of combining the unique features of each model. Both theoretical and empirical evidence show that the accuracy of forecasts can be improved by combining different methods (Newbold and Granger, 1974; Armstrong, 1989; Palm and Zellner, 1992; Ginzburg and Horn, 1994; Zhang, 2003).

The purpose of this paper is firstly to adapt the NEAT algorithm to time series forecasting applications and compare its performance with the MLP, which is the most popular type of neural network used for forecasting purposes (Lachtermacher and Fuller, 1995; Zhang et al., 1998), in order to evaluate its potential modelling power. Secondly, in an attempt to improve the overall prediction performance, the two techniques are combined and the resulting performance assessed and compared with that of the techniques used separately.

The paper is organised as follows: the next section describes the feedforward MLP. The NEAT algorithm is presented in the third section. Section four deals with combining forecasts. Section five provides information on the model estimation processes. Then, in section six, empirical results and their implications are presented. Conclusions arising out of the experiments are presented and possible directions for future investigation are reported in the final section.

2. Multilayer Perceptron

Many different kind of neural networks have been proposed in the past. However, most attention has been directed towards the MLP because of its relative simplicity and successful application in various fields. The MLP employs the backpropagation algorithm that is based on the steepest gradient descent method during training, which is the process of minimising the objective function of the network by means of changing the connection weights. However, this method has some disadvantages like slow convergence, linear searching, the tendency to get trapped in local minima, possible oscillations during searching and sensitivity to learning rate. With the intention of ameliorating these difficulties with the gradient descent method, second-order methods such as conjugate gradient and Levenberg-Marquardt (which are more efficient nonlinear optimisation techniques) are used. More details about these techniques and the MLP can be found in Hagan (1994).

The MLP is structured with a particular topology for the duration of each experimental run. Within each training run, only the connection weights are allowed to change. After finishing one experimental run, the modeller can change some components of the topology to observe the behaviour of neural networks with different architectures. Determining the best neural network architecture requires making a large number of experiments. Also, in order to avoid getting trapped in local minima, many experimental runs should be repeated with the same architecture but different initial weights. In summary, the success of MLP neural networks could be said to be dependent on a large number of experimental runs and much human intervention during the training process. Even though there are many rules of thumb to assist in the design of MLP architectures, there are no certain rules accepted by all researchers as being the best. Hence the best way to handle this problem is to employ empirical techniques.

3. NeuroEvolution of Augmenting Topologies (NEAT)

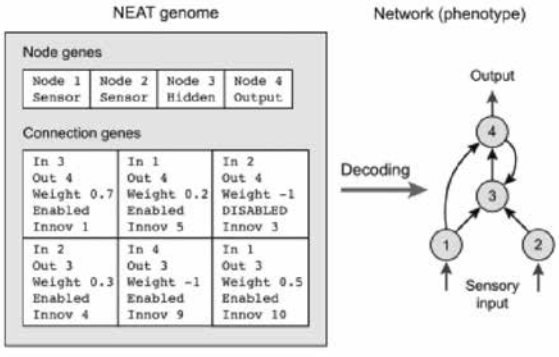
EANN algorithms are designed to utilise the principles of evolutionary algorithms to derive neural networks in an attempt to minimise human intervention during the training process. Numerous studies have been published in which evolutionary algorithms are used to evolve and optimise architecture design, connection weights, input feature selection, connection weight initialisation, transfer function optimisation etc. (Yao, 1999; Floreano et al., 2008). The focus of this paper is on the simultaneous evolution of the architecture and connection weights. The main advantage in using the evolution of the connection weights is to avoid the principal drawback of the backpropagation algorithm by taking advantage of properties of evolutionary algorithms. Evolutionary algorithms are population-based search strategies and, as such, are less likely to get trapped in local minima than search algorithms based on gradient information. When the evolution of architectures is carried out regardless of connection weights, the result is likely to be suboptimal because of poor fitness evaluation. To overcome this problem, the evolution of architectures with connection weights can be performed simultaneously. Different ideas have been proposed concerning the joint evolution of weights and architecture (Maniezzo, 1994; Angeline et al., 1994; Gruau et al., 1996; Yao and Liu, 1997; Leung et al., 2003; Pedrajas et al., 2003; Islam et al., 2003; Palmes et al., 2005).

Among these papers, the NEAT algorithm proposed by Stanley and Miikkulainen (2002) has received the most citations and has been applied to different problems such as pole balancing (Stanley and Miikkulainen, 2002), robot control (Stanley and Miikkulainen, 2004), computer games (Reisinger et al., 2007) and an automobile crash warning system (Stanley et al., 2005) but no studies have analyzed the behaviour of the algorithm for time series forecasting. NEAT automatically takes into account the number of hidden units and the particular connections that are necessary in order to be able to model the problem at hand during training while setting the values of the connection weights according

to the objective function (error measure). The modeller decides the form of the objective function. The NEAT method consists of three crucial components. These three components were developed in response to three different issues. Firstly, due to the competing conventions problem (also known as the permutations problem), using the crossover operator for the evolution of architectures has become a subject of discussion between researchers and some of them have put forward several suggestions that rely only on the mutation operator and don't employ crossover (Yao and Liu, 1997; Palmes et al., 2005). NEAT presents a genetic representation that allows meaningful crossover between diverse topologies through historical markings to overcome the problem of competing conventions. An example of the genetic encoding used by the NEAT algorithm is given in Figure 1. The second issue is how to protect structural innovation that needs a few generations to gain its full potential functionality. For this problem, a historical genetic record is used to protect similar individuals under speciation. Thus, structural innovations that may disappear because of their low fitness at the beginning of their evolution are protected and given some time to optimise themselves. The last issue is how to minimise error topologies without any fitness function that measures complexity. To address this problem, the algorithm starts with an initial population which consists of neural networks without any hidden nodes and, after that, the number of hidden nodes is increased incrementally. For a more detailed description of the NEAT algorithm see Stanley and Miikkulainen (2002).

Figure 1: The mapping from NEAT genome to a phenotype structure

Source: Floreano et al., 2008, (pp.50).



4. Combining Forecasts

A large number of papers dealing with combination methods for forecasting applications of neural networks can easily be found in the literature (Wedding and Cios, 1996; Luxhoj et al., 1996; Kodogiannis and Lois, 2002; Tseng et al., 2002; Zhang, 2003; Zou et al., 2007). Some papers have claimed that combining individual forecasts of dissimilar models or methods based on different information would lead to superior forecasts (Granger, 1989; Perrone and Cooper, 1992; Nikolopoulos et al., 2007). These findings encouraged the use of combining techniques for this study because the MLP and NEAT algorithms are somewhat different types of neural network. The MLP make use of gradient information to search for minima in the error surface, while NEAT takes advantage of an evolutionary algorithm and is therefore less likely to get trapped in local minima. However, even though the NEAT algorithm is more powerful in avoiding local minima, the MLP is able to do the search in a more finely tuned way. Sometimes, this property of the MLP can lead to finding lower points of the error surface than NEAT. The final model reached after spending much effort to model with neural networks cannot be claimed to be an optimal solution but it is more likely to be a suboptimal solution to the problem in question. Granger (1989) stated that in order to obtain better forecasts using combination methods, the component models should be suboptimal. Moreover, as expressed by Zhang (2003), the final selected model will still be unable to produce perfect forecasts for the future, because of factors like sampling variation, model uncertainty and structural changes in the model. After taking all of these comments into consideration, it is expected that the overall result of combined forecasts can lead to improved forecasting performance.

The combination approach implemented in this work is to use a simple averaging technique because of its robust properties and easily applied nature. Review articles of combined forecasts by Granger (1989), Clemen (1989) and de Menezes et al. (2000) show that many methods ranging from complex to simple ones are presented in the literature. The simplest method is to use an arithmetic average of the individual forecasts to produce combined forecasts. Elaborate combination schemes that depend on the estimation of many parameters have

often been found to underperform schemes that utilise a simple equal-weighted strategy (Clemen, 1989; Stock and Watson, 2001). Armstrong (2001) stated that when there is much uncertainty for the problem at hand, using simple methods would be a good strategy. Many studies have found that a simple mean of the two forecasts performs relatively well (de Menezes et al., 2000).

5. Model Building

When the literature is examined, it can be seen that some neural networks have two hidden layers, while most of them have just one hidden layer. In forecasting problems, many successful applications have just one hidden layer and use the logistic function for the hidden nodes and a linear function for the output nodes (Zhang, 1998; Haykin, 1999). Most neural network software provides default values for the learning rate and momentum parameters that typically work well and automatically decrease the learning rate and increase the momentum values as convergence is approached (Kaastra and Boyd, 1996). The last explanation is especially useful for MLP modelling. In this study, we also constructed neural networks with these features, i.e. one hidden layer and the logistic function for the hidden nodes and a linear function for the output node. One step-ahead forecasting was performed. Thus just one output node was employed in the output layer. The data were transformed to have zero mean and unit variance before the training process.

The data sets were divided into three subsets for training, validation and testing. The validation set was used to stop the training process, with aim of avoiding overfitting in order to determine the best neural network architecture structure. The final neural network model was selected according to the minimum error value reached on the validation set. The test set was used to evaluate the performance of the final neural network over the unseen data. It is a well-known fact that there is no consensus on which error measure is the best for forecasting (Makridakis et al., 1982; Armstrong and Fildes, 1995). Hence, two classical forecast evaluation statistics given in Equation 2 and 3, mean square error (MSE) and mean absolute error (MAE), were employed to compare the performances of neural network models on test data. The results obtained for these error measures were similar to those of other error measures which were not reported here to save space. All neural networks models in this study were implemented in the MATLAB software.

$$MSE=\sum_{t=1}^{n}\frac{(\hat{y}_t-y_t)^2}{n} \quad (1)$$

$$MAE=\frac{1}{n}\sum_{t=1}^{n}|\hat{y}_t-y_t| \quad (2)$$

where y_t and $\hat{y}_t$ represent the observation and forecast value at time t, respectively, and n denotes the number of data points in test set.

The aim of a forecasting application is to approximate the relationship between the output (y_t) and the inputs (y_{t-1}, y_{t-2}, ... , y_{t-p}). The mathematical form of the MLP model used in this study is as follows:

$$y_t=w_0+\sum_{j=1}^{a} w_j f(w_{0j}+\sum_{j=1}^{p} w_{ij}\, y_{t-i})+\varepsilon_t \quad (3)$$

where w_0 and w_{0j} (j=1,2, ..., q) represent the biases on the neurons, w_j (j=1,2, ..., q) and w_{ij} (i=1,2, ..., p; j=1,2, ..., q) denote the connection weights between the layers of the model, f(.) shows the logistic function for the hidden nodes, and p and q are the number of input and hidden neurons, respectively.

When the MLP model given in Equation 3 is considered, it can be said that the MLP model performs a nonlinear functional mapping (nonlinear autoregressive model) from the lagged values (y_{t-1}, y_{t-2}, ... , y_{t-p}) to the future value (y_t). While hidden nodes determine the degree of nonlinearity of the MLP model, the number of input nodes (or lagged observations) is the most important element in time series forecasting by the reason of identifying the autocorrelation structure of the time series at hand (Zhang et al., 2001; Aras and Kocakoç, 2016). There is no generally accepted rule to find out these substantial parameters. For this reason, a comprehensive experiment was conducted to determine the best number of hidden and input nodes in the scope of this study.

For the MLP experiments, the Levenberg-Marquardt optimisation method (which is frequently used for the backpropagation algorithm) was used to determine the connection weights for the MLP. In order to investigate

the best number of input and hidden nodes for the problem in hand, both the number of input nodes (i.e. lags of the time series) and the number of hidden units were each varied from one to ten and the resulting 10 x 10 grid subjected to experimental search in the following way. Each of the 100 different architectures was replicated 30 times with different initial weights. After that, the average validation set performance of the 300 neural networks corresponding to each number of input nodes was evaluated in order to determine the best number of input nodes. The process was repeated in order to select the best number of hidden units. This entire process was repeated 25 times, to yield 25 neural networks which were used to forecast the values in the test set.

With regard to the NEAT experiments, some further parameter settings should be mentioned. Fitness was evaluated according to a fitness function that depended on fitting error and two penalty terms. The first of these penalises the number of neurons in the input and hidden layers, in order to guard against overfitting. The second penalty term penalises architectures that are no longer improving their fitness, helping NEAT avoid stagnation or getting stuck at local minima. The fitness function f was defined as follows:

$$f = C - error - \alpha N - \beta G \quad (4)$$

where;

C: A constant that depends on the task to ensure a positive fitness value and error is the forecasting error used which, in these experiments, was either (MSE) or (MAE).

N: The number of neurons (N) in use.

G: The number of consecutive generations (G) for which the model did not make any improvement.

α: The penalty parameter for the number of neurons (N) in use.

β: The penalty parameter for the number of consecutive generations (G).

To create species, the NEAT algorithm used compatibility function given in Equation 5. By this equation, the distance between any two genomes was computed as follows:

$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \bar{W} \quad (5)$$

where E is the number of excess genes, D denotes the number of disjoint genes, and W is the average weight differences of matching genes. The coefficients for compatibility measurement were found experimentally as c_1=1.0, c_2=1.0, c_3=1.0 and the compatible initial threshold δ_t was set at 7.0. Species were created by speciation based on the distance between two genomes. If the distance was below δt, the two genomes were said to be compatible and belong to the same species. If the number of species was above 10, δt was increased by 0.2 to decrease the number of species. In addition, if the number of species was below 10, δt was decreased by 0.2 to make more room for new species. Also, a population size of 2000 NEAT networks was used. The probabilities of node mutation, weight mutation, and connection mutation were set as 0.6, 0.7, and 0.85, respectively. The probabilities of adding a new node and a new link were set at 0.1 and 0.2, respectively. These two probabilities were higher than in most of the studies using NEAT, because the fitness function had been modified so that the fitness evaluation could penalise those models using more neurons to get better results. However, if the model generated significantly better results, it could be accepted. The interspecies mating rate was 0.01. These parameter values were found by the experiments conducted. Finally, in order to prevent stagnation, the worst species in 50 consecutive generations was not allowed to reproduce. The same experimental settings were used in both forecasting problems. As with the MLP, the NEAT algorithm was repeated 25 times with the same experimental parameter settings so that the final result was 25 forecasting attempts by each algorithm on each dataset.

Two well-known real-world time series were used in these experiments to investigate the forecasting ability of the MLP and NEAT algorithms and their combined forecasts. The first data set was the lynx series which consists of the number of lynx trapped per year in the Mackenzie River district of Northern Canada and has 114 observations between 1821 and 1934. The data set is plotted in Figure 2. Prior to the modelling process, logarithms (to base 10) of the data were taken for this series, as specified by Priestley (1988) and others (Zhang, 2003;

Khashei and Bijari, 2010). This data set has been widely employed in the literature to compare performances of linear and nonlinear models (Wong and Li, 2000; Stone and He, 2007). The first 70 observations were used for training, the next 30 observations were used for validation and the last 14 observations were held out for testing purposes. The second data set was the sunspot series that is a record of the annual number of sunspots from 1700 to 1987. Figure 3 shows the sunspot series. The first 161 observations of the series were used for training, the next 60 observations of data were used for validation and the remainder of the series (consisting of 67 observations) were held out for testing. These data series were considered to be both nonlinear and non-Gaussian and were used in order to evaluate the effectiveness of nonlinear modelling procedures (De Groot and Würtz, 1991;Ghiassi and Saidane, 2005). They are available from Hyndman's Time Series Data Library (2012).

Three different combination schemes were employed in this study. Firstly, the 25 replicates from a particular method were employed. Secondly, for each method, the means of all possible pairs were selected from the 25 replicates, giving 300 means in total. Finally, pairs were selected by taking one of the 25 MLP replicates and one of the 25 NEAT replicates in all possible ways and calculating the means, yielding 625 altogether. The performance of the models on each dataset was evaluated in terms of two commonly employed measures: mean square error (MSE) and mean absolute error (MAE). Planned comparisons (Keppel, 1973) were employed to test following hypotheses:

1. MLP was tested against NEAT. NEAT was expected to perform better because of its greater resistance to getting trapped in local minima.

2. MLP was tested against MLP pairing. The pairing technique was expected to produce superior forecasting performance, as discussed earlier in Section 4.

3. NEAT was tested against NEAT pairing. As in the second comparison (above) the pairing method was expected to be better.

4. MLP pairing was tested against MLP+NEAT pairing. As discussed earlier in Section 4, the pairing of two different types of algorithm would be expected to produce

Figure 2: Canadian lynx series

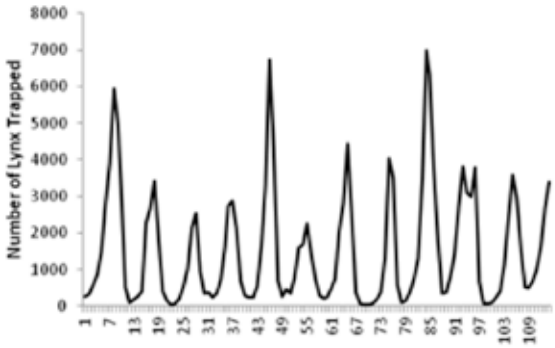
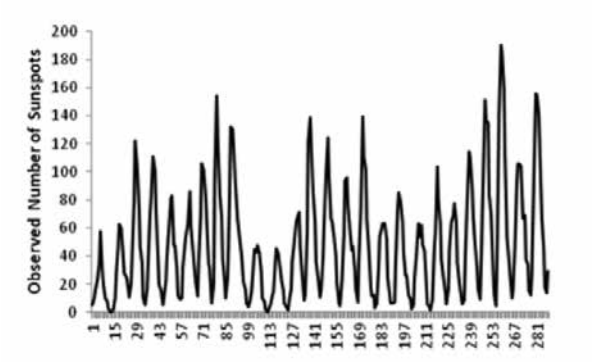


Figure 3: Sunspot series



better performance than pairing using the same technique for each member of the pair.

5. NEAT pairing was tested against MLP+NEAT pairing. As in the fourth comparison (above), the combination of two different techniques was expected to produce superior performance.

All comparisons were made twice for each dataset (once using MSE and once using MAE).

6. Empirical Results and Discussions

As can be seen in Table 1, which consists of the 25 performance results for the MLP and NEAT algorithms for the lynx data, their forecasting performance on the test data varied considerably.

Judging from Table 1, the optimum number of lags for the lynx series is two. However, in four of the replicates for the NEAT method, the best number of input nodes (lags)

Table 1. Initial results for NEAT and MLP with lynx data

NEAT				MLP			
Input	**Hidden**	**MSE**	**MAE**	**Input**	**Hidden**	**MSE**	**MAE**
2	11	0.00980	0.06941	2	7	0.01452	0.10944
2	7	0.01440	0.09198	2	8	0.02397	0.12574
3	8	0.01569	0.10169	2	7	0.00986	0.07813
2	2	0.01184	0.07914	2	3	0.00839	0.07767
2	5	0.01511	0.09697	2	8	0.01323	0.09367
3	4	0.01303	0.08644	2	8	0.00817	0.06681
3	6	0.01255	0.08593	2	4	0.00966	0.07143
2	9	0.01539	0.09183	2	8	0.01078	0.08593
3	6	0.01571	0.09848	2	7	0.02054	0.11987
2	3	0.01593	0.10276	2	5	0.00965	0.07865
2	2	0.01542	0.09825	2	8	0.01057	0.07867
2	8	0.01680	0.10413	2	7	0.01963	0.10849
2	11	0.01624	0.10433	2	6	0.01388	0.10188
2	1	0.01192	0.08529	2	4	0.02100	0.11256
2	7	0.01479	0.08736	2	9	0.01210	0.07514
2	9	0.00587	0.05489	2	6	0.01079	0.07127
2	9	0.01481	0.09763	2	4	0.01097	0.08276
2	6	0.01482	0.09175	2	3	0.01298	0.07522
2	7	0.01452	0.08882	2	8	0.01366	0.08498
2	3	0.01257	0.08812	2	3	0.01502	0.08745
2	2	0.01416	0.08793	2	9	0.01273	0.08933
2	4	0.01239	0.07594	2	7	0.01033	0.07441
2	5	0.01006	0.08328	2	7	0.01108	0.07440
2	8	0.01202	0.08194	2	8	0.01438	0.09198
2	16	0.00590	0.05585	2	9	0.00912	0.06690

was found to be three. We know from previous studies (Zhang, 2001; Zhang et al., 1998) that the effect of the number of hidden nodes on the performance of neural networks is lower than the effect of the number of input nodes. For this reason, we can expect that the variability in the number of hidden units to be higher than that of the input units and this was indeed found to be the case, with the optimum number ranging from two to 16 for the NEAT algorithm and from three to nine for the MLP.

Turning to Table 2, we find that the number of input nodes (lags) selected for the sunspot data is more variable than for the lynx data. The number ranged from two to seven for the NEAT algorithm and from three

Table 2. Initial results for NEAT and MLP with sunspot data

NEAT				MLP			
Input	**Hidden**	**MSE**	**MAE**	**Input**	**Hidden**	**MSE**	**MAE**
2	3	396.104	14.581	3	8	565.156	16.827
3	7	409.993	14.893	6	3	444.141	14.994
3	3	409.454	15.367	3	10	327.226	13.197
3	3	407.908	14.928	6	8	499.102	15.919
4	2	403.125	15.122	3	7	343.507	13.662
4	3	381.831	14.196	3	8	471.989	14.317
4	7	427.594	15.542	6	3	472.811	15.595
6	3	378.930	14.620	3	4	754.677	19.397
6	7	338.141	13.775	3	7	412.109	14.130
6	15	370.695	14.788	6	8	435.796	15.288
3	7	408.166	14.744	3	5	522.565	16.226
7	2	418.734	14.912	6	9	423.894	13.391
5	8	390.558	15.151	6	5	400.081	13.917
6	11	346.642	13.977	6	10	501.915	16.215
6	6	313.751	13.447	6	9	401.828	15.296
6	8	315.487	13.423	6	7	347.691	14.453
7	2	332.932	13.066	6	7	432.775	14.758
7	6	335.094	14.240	3	9	549.567	16.852
6	7	400.374	14.891	6	6	456.940	15.561
6	11	375.150	14.677	6	9	425.127	15.100
6	13	339.693	13.606	6	7	366.391	14.572
6	4	309.705	13.416	3	8	480.061	14.840
6	8	364.908	14.655	6	5	475.867	15.935
5	6	419.848	14.330	6	8	530.671	16.123
6	3	409.314	14.969	4	4	695.002	18.109

to six for the MLP. The most common number of input nodes was six. Again, the number of hidden nodes was found to be greater than the number of input nodes, ranging from two to 15 for the NEAT algorithm and from three to ten for the MLP.

Summary statistics for the various conditions are shown in Tables 3 and 4. Broadly speaking, the distribution of the results for each of the conditions was approximately Gaussian. However, the variances differed considerably between the various conditions. Therefore, the comparisons were made using the Welch-Satterthwaite version of the t-test (Satterthwaite, 1946; Welch, 1947) that uses separate variances for the groups rather than a pooled variance and also applies an adjustment to the number of degrees of freedom. The results of the various t-tests are displayed in Tables 5 and 6.

As regards the first hypothesis, the results are dataset dependent. They show that, compared with the MLP,

Table 3. Summary statistics for sunspot data

Method	No. of Scores	Mean of MSE	Std. Dev. of MSE	Mean of MAE	Std. Dev. of MAE
MLP	25	469.5	99.6	15.387	1.435
NEAT	25	376.17	37.12	14.453	0.676
MLP Pair	300	422.46	72.56	14.535	1.038
NEAT Pair	300	358.24	28.74	14.101	0.543
Both Pair	625	353.58	51.45	13.819	0.912

Table 4. Summary statistics for lynx data.

Method	No. of Scores	Mean of MSE	Std. Dev. of MSE	Mean of MAE	Std. Dev. of MAE
MLP	25	0.01308	0.00417	0.08731	0.01673
NEAT	25	0.01327	0.00290	0.08761	0.01313
MLP Pair	300	0.01132	0.00234	0.07908	0.01026
NEAT Pair	300	0.01202	0.00240	0.08265	0.01046
Both Pair	625	0.01122	0.00213	0.07803	0.00928

Table 5. t-test comparison of NEAT, MLP and Combined Forecasts for sunspot set

	MSE			MAE		
Comparison	t	df	p	t	df	p
MLP-NEAT	4.39	30	0.000	2.94	34	0.006
MLP-MLP Pair	2.31	26	0.029	2.91	26	0.007
NEAT-NEAT Pair	2.36	26	0.026	2.53	26	0.018
MLP Pair-Both Pair	14.76	448	0.000	10.20	526	0.000
NEAT Pair-Both Pair	1.76	902	0.078	5.87	881	0.000

the NEAT algorithm produced significantly better forecasting performance on the sunspot data, whichever measure of error was used. However, there was no corresponding significant benefit with the lynx data. It looks as if the MLP may have been more prone than the NEAT algorithm to getting stuck in local minima on the sunspot data than with the lynx data.

Table 6. t-test comparison of NEAT, MLP and Combined Forecasts for lynx set

	MSE			MAE		
Comparison	t	df	p	t	df	p
MLP-NEAT	-0.19	42	0.853	-0.07	45	0.945
MLP-MLP Pair	2.09	25	0.047	2.42	25	0.023
NEAT-NEAT Pair	2.10	26	0.046	1.84	26	0.077
MLP Pair-Both Pair	0.60	542	0.548	1.51	540	0.132
NEAT Pair-Both Pair	4.90	531	0.000	6.51	531	0.000

Comparing the performance of each algorithm with that of its pairwise counterpart, we find that the pairwise combination approach produces significantly better forecasting performance for the MLP on both datasets, whichever error measure was used. For the NEAT algorithm, a similar picture emerged, except that the result for the MAE did not reach statistical significance.

When we compare the forecasting performance of the pairwise MLP, with the combination technique, we find that the combination approach produces significantly better forecasting performance for the sunspot data but not for the lynx data, whichever error measure was employed.

The results for the final comparison were more mixed. The combination technique yielded significantly better forecasting performance on the lynx data than the pairwise version of the NEAT algorithm, for both error measures. For the sunspot data, the combination method gave significantly better results than the MLP as measured by the MAE but not according to the MSE.

In summary, 18 of the 20 comparisons yielded results in the expected direction, although not all of these were statistically significant. The two contrary results were for the simple algorithms operating on the lynx data were small and very far from statistical significance.

The NEAT algorithm showed superior forecasting performance to that of the MLP. Pairwise combinations of the algorithms produced better performance than that of their simple counterparts. Combining the two algo-

rithms produced superior performance to that of the pairwise algorithms.

After the performance comparison between the MLP, the NEAT, and the pairwise combination of these methods, we conducted an experiment by comparing the obtained results with the autoregressive integrated moving average (ARIMA) model and two well-known hybrid models by Zhang (2003) and Khashei and Bijari (2010). These studies used the same ARIMA models as a subset autoregressive model of order 12 for the lynx data and a subset autoregressive model of order 9, respectively. Hence, we used the same ARIMA models in this study. The results of the ARIMA models and the two mentioned hybrid models are presented in Table 7 for both series. To perform a statistical performance comparison, a one-sample z-test was exploited in this study. This test is robust to violations of the assumption of normal distribution when the sample size is sufficiently large (Newbold et al. 2009). For each comparison point, the null hypothesis (H_0) was that the error measure resulting from the ARIMA model or one of the hybrid methods is the mean of the population. The alternative hypothesis (H_1) is that the mean of the error measures obtained from the concerned methods (i.e. the MLP, the NEAT, or the pairwise combinations) is less than the population mean. In this case, the error measures in Table 7 show the assumed population means for each of the one-sample z-tests to be conducted.

Table 7. The results of the linear and hybrid models

	Lynx		Sunspot	
Models	**MSE**	**MAE**	**MSE**	**MAE**
ARIMA	0.02049	0.11225	306.082	13.034
Zhang's hybrid model	0.01723	0.10397	280.159	12.780
A hybrid (p,d,q) model by Khashei and Bijari	0.01361	0.08962	234.206	12.118

The results of the one-sample z-tests for the lynx and sunspot data sets are presented in Tables 8 and 9. In these tables, z-values and the corresponding p-values, shown in parentheses, are given for both error measures considered. As shown in Table 8, the methods used in

Table 8. The results of the one sample z-test for the lynx data

	MSE			MAE		
Methods	**ARIMA**	**Zhang**	**Khashei**	**ARIMA**	**Zhang**	**Khashei**
MLP	-8.88 (0.00)	-4.98 (0.00)	-0.63 (0.26)	-7.45 (0.00)	-4.98 (0.00)	-0.69 (0.25)
NEAT	-12.45(0.00)	-6.83 (0.00)	-0.58 (0.28)	-9.38 (0.00)	-6.23 (0.00)	-0.76 (0.22)
MLP Pair	-67.87(0.00)	-43.75(0.00)	-16.95(0.00)	-55.99(0.00)	-42.02(0.00)	-17.79(0.00)
NEAT Pair	-61.13(0.00)	-37.60(0.00)	-11.47(0.00)	-49.01(0.00)	-35.30(0.00)	-11.54(0.00)
Both Pair	-108.80(0.00)	-70.54(0.00)	-28.05(0.00)	-92.18(0.00)	-69.88(0.00)	-31.22(0.00)

Table 9. The results of the one sample z-test for the sunspot data.

	MSE			MAE		
Methods	**ARIMA**	**Zhang**	**Khashei**	**ARIMA**	**Zhang**	**Khashei**
MLP	8.20 (0.99)	9.50 (0.99)	11.81(0.99)	8.19 (0.99)	9.08 (0.99)	11.39(0.99)
NEAT	9.44 (0.99)	12.93(0.99)	19.12(0.99)	10.49(0.99)	12.37(0.99)	17.27(0.99)
MLP Pair	27.78(0.99)	33.97 (0.99)	44.94(0.99)	25.05(0.99)	29.28(0.99)	40.33(0.99)
NEAT Pair	31.43(0.99)	47.06(0.99)	74.75(0.99)	34.03(0.99)	42.14(0.99)	63.25(0.99)
Both Pair	23.07 (0.99)	35.67(0.99)	58.00(0.99)	21.52(0.99)	28.48(0.99)	46.62(0.99)

this study led to statistically better performances in all almost all the cases than the linear model, the hybrid models for the lynx data. However, for the sunspot data, neither the single methods nor the pairwise combinations produced forecasts that were statistically more accurate than the ARIMA and the hybrid methods under investigation. It can be said that the results achieved are highly dependent on the data sets used even if the pairwise combined forecasts tend to exhibit better forecasting performances compared to the single forecasts.

7. Conclusions

The main aim of this paper was to implement the NEAT algorithm for time series forecasting and to compare its performance with the MLP in order to evaluate its capability of forecasting. For this purpose, two nonlinear time series were used and the resulting performance was found to be similar to that obtained from the MLP for the one of the series and superior to the MLP for the other series investigated. As a result, it can be said that there is some evidence that NEAT is a promising tool for nonlinear forecasting.

The other interesting conclusion from this study is that combined forecasts resulting from simple averaging of pairwise selected MLP and NEAT forecasts produced superior performance, whichever algorithm was used. Possibly the most important finding in this study is that combining the MLP forecasts with those from the relatively novel NEAT technique can improve the accuracy of forecasts still further. The results obtained in this study suggest that combining the MLP and NEAT algorithms helps to capture nonlinear patterns in the data more effectively than either of the algorithms used separately.

For future work, it is intended to apply the NEAT algorithm to further nonlinear time series with a view to reaching more information about its usefulness. Although this study achieved interesting result with equal-weight combinations, other researchers have experimented with different schemes that might merit further investigation. Also, further work could be done on composition of the combinations, both in terms of the number of replicates and the particular combination scheme used.

References

Angeline, P. J., Saunders, G. M., & Pollack, J. B. (1994). An evolutionary algorithm that constructs recurrent neural networks. *IEEE transactions on Neural Networks*, 5(1), 54-65. **[CrossRef]**

Aras, S., & Kocakoç, İ. D. (2016). A new model selection strategy in time series forecasting with artificial neural networks: IHTS. *Neurocomputing*, 174, 974-987. **[CrossRef]**

Armstrong, J. S. (1989). Combining forecasts: The end of the beginning or the beginning of the end? *International Journal of Forecasting*, 5(4), 585-588. **[CrossRef]**

Armstrong, J. S. (Ed.). (2001). "Combining forecasts", Chapter 13 in Principles of forecasting: a handbook for researchers and practitioners (Vol. 30). *Springer Science & Business Media*. **[CrossRef]**

Armstrong, J. S., & Fildes, R. (1995). Correspondence on the selection of error measures for comparisons among forecasting methods. *Journal of Forecasting*, 14(1), 67-71. **[CrossRef]**

Bates, J. M., & Granger, C. W. (1969). The combination of forecasts. Journal of the *Operational Research Society*, 20(4), 451-468. **[CrossRef]**

Clemen, R. T. (1989). Combining forecasts: A review and annotated bibliography. *International Journal of Forecasting*, 5(4), 559-583. **[CrossRef]**

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 2(4), 303-314. **[CrossRef]**

De Gooijer, J. G., & Kumar, K. (1992). Some recent developments in non-linear time series modelling, testing, and forecasting. *International Journal of Forecasting*, 8(2), 135-156. **[CrossRef]**

De Groot, C., & Würtz, D. (1991). Analysis of univariate time series with connectionist nets: A case study of two classical examples. *Neurocomputing*, 3(4), 177-192. **[CrossRef]**

De Menezes, L. M., Bunn, D. W., & Taylor, J. W. (2000). Review of guidelines for the use of combined forecasts. *European Journal of Operational Research*, 120(1), 190-204. **[CrossRef]**

Enders, W. (2008). Applied econometric time series. John Wiley & Sons.

Floreano, D., Dürr, P., & Mattiussi, C. (2008). Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1), 47-62. **[CrossRef]**

Ghiassi, M., & Saidane, H. (2005). A dynamic architecture for artificial neural networks. *Neurocomputing*, 63, 397-413. **[CrossRef]**

Ginzburg, I., & Horn, D. (1994). Combined neural networks for time series analysis. *Advances in Neural Information Processing Systems*, 224-224.

Granger, C. W. (1989). Invited review combining forecasts—twenty years later. *Journal of Forecasting*, 8(3), 167-173. **[CrossRef]**

Gruau, F., Whitley, D., & Pyeatt, L. (1996, July). A comparison between cellular encoding and direct encoding for genetic neural networks. In Proceedings of the 1st annual conference on genetic programming (pp. 81-89). MIT Press.

Hagan, M. T., Demuth, H. B., Beale, M. H., & De Jesús, O. (1996). Neural network design (Vol. 20). Boston: PWS publishing company.

Haykin, S. S. (2001). Neural networks: a comprehensive foundation. Tsinghua University Press.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359-366. **[CrossRef]**

Hyndman, R. J. (2012). Time series data library. www.robjhyndman.com/TSDL/. Accessed on 12 June 2012.

Islam, M. M., Yao, X., & Murase, K. (2003). A constructive algorithm for training cooperative neural network ensembles. IEEE Transactions on neural networks, 14(4), 820-834. **[CrossRef]**

Kaastra, I., & Boyd, M. (1996). Designing a neural network for forecasting financial and economic time series. *Neurocomputing*, 10(3), 215-236. **[CrossRef]**

Khashei, M., & Bijari, M. (2010). An artificial neural network (p, d, q) model for timeseries forecasting. Expert Systems with applications, 37(1), 479-489. **[CrossRef]**

Keppel, G. (1973). Design and Analysis: A Researcher's Handbook. Prentice-Hall, New Jersey.

Kodogiannis, V., & Lolis, A. (2002). Forecasting financial time series using neural network and fuzzy system-based techniques. *Neural Computing & Applications*, 11(2), 90-102. **[CrossRef]**

Lachtermacher, G., & Fuller, J. D. (1995). Back propagation in time-series forecasting. *Journal of Forecasting*, 14(4), 381-393. **[CrossRef]**

Leung, F. H. F., Lam, H. K., Ling, S. H., & Tam, P. K. S. (2003). Tuning of the structure and parameters of a neural network using an improved genetic algorithm. *IEEE Transactions on Neural Networks*, 14(1), 79-88. **[CrossRef]**

Luxhøj, J. T., Riis, J. O., & Stensballe, B. (1996). A hybrid econometric—neural network modeling approach for sales forecasting. International *Journal of Production Economic*s, 43(2), 175-192. **[CrossRef]**

Makridakis, S., Andersen, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., & Winkler, R. (1982). The accuracy of extrapolation (time series) methods: Results of a forecasting competition. *Journal of Forecasting*, 1(2), 111-153. **[CrossRef]**

Maniezzo, V. (1994). Genetic evolution of the topology and weight distribution of neural networks. *IEEE Transactions on Neural Networks*, 5(1), 39-53. **[CrossRef]**

Newbold, P., & Granger, C. W. (1974). Experience with forecasting univariate time series and the combination of forecasts. *Journal of the Royal Statistical Society*. Series A (General), 131-165. **[CrossRef]**

Newbold, P., Carlson, W., & Thorne, B. (2009). Statistics for business and economics. Prentice Hall.

Nikolopoulos, K., Goodwin, P., Patelis, A., & Assimakopoulos, V. (2007). Forecasting with cue information: A comparison of multiple regression with alternative forecasting approaches. *European Journal Of Operational Research*, 180(1), 354-368. **[CrossRef]**

Palm, F. C., & Zellner, A. (1992). To combine or not to combine? Issues of combining forecasts. *Journal of Forecasting*, 11(8), 687-701. **[CrossRef]**

Palmes, P. P., Hayasaka, T., & Usui, S. (2005). Mutation-based genetic neural network. IEEE *Transactions on Neural Networks*, 16(3), 587-600. **[CrossRef]**

García-Pedrajas, N., Hervás-Martínez, C., & Muñoz-Pérez, J. (2003). COVNET: a cooperative coevolutionary model for evolving artificial neural networks. IEEE *Transactions on Neural Networks*, 14(3), 575-596. **[CrossRef]**

Perrone, M.P. & Cooper, L.N. (1993). When networks disagree: ensemble method for hybrid neural networks. in: R.J. Mammone (Ed.), Neural Networks for Speechand Image Processing, Chapman & Hall, London, pp.126-142.

Priestly, M.B. (1988). Non-linear and Non-Stationary Time Series Analysis. Academic Press.

Reid, D. J. (1968). Combining three estimates of gross domestic product. *Economica*, 35(140), 431-444. **[CrossRef]**

Reisinger, J., Bahceci, E., Karpov, I., & Miikkulainen, R. (2007, April). Coevolving strategies for general game playing. In 2007 IEEE Symposium on Computational Intelligence and Games (pp. 320-327). **[CrossRef]**

Satterthwaite, F. E. (1946). An approximate distribution of estimates of variance components. *Biometrics Bulletin*, 2(6), 110-114. **[CrossRef]**

Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2), 99-127. **[CrossRef]**

Stanley, K. O., & Miikkulainen, R. (2004). Competitive coevolution through evolutionary complexification. *J Artif Intell Res* (JAIR), 21, 63-100.

Stanley, K., Kohl, N., Sherony, R., & Miikkulainen, R. (2005, June). Neuroevolution of an automobile crash warning system. In Proceedings of the 7th annual conference on Genetic and evolutionary computation (pp. 1977-1984). **[CrossRef]**

Stock, J.H. & Watson, M. (1999). A Comparison of Linear and Nonlinear Uni-variate Models for Forecasting Macroeconomic Time Series. in Robert F. Engle and Halbert White eds., Cointegration, Causality, and Forecasting: A Festschrift in Honor of Clive W. J. Granger, Oxford University Press, Oxford, U.K., pp.1-44.

Stone, L., & He, D. (2007). Chaotic oscillations and cycles in multi-trophic ecological systems. Journal of Theoretical Biology, 248(2), 382-390. **[CrossRef]**

Tseng, F. M., Yu, H. C., & Tzeng, G. H. (2002). Combining neural network model with seasonal time series ARIMA model. *Technological Forecasting and Social Change*, 69(1), 71-87. **[CrossRef]**

Wedding, D. K., & Cios, K. J. (1996). Time series forecasting by combining RBF networks, certainty factors, and the Box-Jenkins model. *Neurocomputing*, 10(2), 149-168. **[CrossRef]**

Welch, B. L. (1947). The generalization ofstudent's' problem when several different population variances are involved. *Biometrika*, 34(1/2), 28-35. **[CrossRef]**

Wong, C. S., & Li, W. K. (2000). On a mixture autoregressive model. Journal of the Royal Statistical Society: *Series B* (Statistical Methodology), 62(1), 95-115.v

Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9), 1423-1447. **[CrossRef]**

Yao, X., & Liu, Y. (1997). A new evolutionary system for evolving artificial neural networks. *IEEE Transactions on Neural Networks*, 8(3), 694-713. **[CrossRef]**

Zhang, G., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial neural networks: The state of the art. *InternationalJournal of Forecasting*, 14(1), 35-62. **[CrossRef]**

Zhang, G. P. (2001). An investigation of neural networks for linear time-series forecasting. *Computers & Operations Research*, 28(12), 1183-1202. **[CrossRef]**

Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 50, 159-175. **[CrossRef]**

Zhang, G. P., Patuwo, B. E., & Hu, M. Y. (2001). A simulation study of artificial neural networks for nonlinear time-series forecasting. *Computers & Operations Research*, 28(4), 381-396. **[CrossRef]**

Zou, H. F., Xia, G. P., Yang, F. T., & Wang, H. Y. (2007). An investigation and comparison of artificial neural network and time series models for Chinese food grain price forecasting. *Neurocomputing*, 70(16), 2913-2923.**[CrossRef]**