Proceedings of the 2007 IEEE Symposium on
Computational Intelligence and Data Mining (CIDM 2007)

A New Evolutionary Approach for Time Series Forecasting

Tiago A. E. Ferreira, Germano C. Vasconcelos and Paulo J. L. Adeodato
Center for Informatics – Federal University of Pernambuco
Av. Prof. Luiz Freire, s/n, CDU, 50732-970, Recife - PE - Brazil
Emails: {taef, gcv, pjla}@cin.ufpe.br

***Abstract*— This work introduces a new method for time series prediction - Time-delay Added Evolutionary Forecasting (TAEF) - that carries out an evolutionary search of the minimum necessary time lags embedded in the problem for determining the phase space that generates the time series. The method proposed consists of a hybrid model composed of an artificial neural network (ANN) combined with a modified genetic algorithm (GA) that is capable to evolve the complete network architecture and parameters, its training algorithm and the necessary time lags to represent the series. Initially, the TAEF method finds the most fitted predictor model and then performs a behavioral statistical test in order to adjust time phase distortions that may appear in the representation of some series. An experimental investigation is conducted with the method with some relevant time series and the results achieved are discussed and compared, according to several performance measures, to results found with the multilayer perceptron networks and other works reported in the literature.**

I. Introduction

Many nonlinear approaches have been developed by independent researchers for the prediction of time series such as the bilinear models [1], the threshold autoregressive models [2], the exponential autoregressive models [3], the general state dependent models [4] and others. However, those nonlinear approaches usually involve high technical and mathematical complexities.

Alternative approaches based on artificial neural networks (ANNs) have been proposed for the same purpose [5]. In order to define a solution to a given problem, ANNs require the setting up of a series of system parameters, some of them not always easy to determine. The network topology, the number of processing units, the algorithm for network training (and its corresponding variables) are just some of the parameters that require definition. In addition to those, in the particular case of time series prediction, another crucial element necessary to determine is the relevant time lags to represent the series.

Many researches based on the evolutionary approach for the definition of neural network parameters have produced interesting results in the past [6]. Some of these works have focussed on the evolution of the network weights whereas others aimed at evolving the network architecture. The pioneering work of Stanley e Miikkulainen [7] shows an efficient evolutionary methodology (NEAT) that combines genetic algorithms and neural networks to define optimal solutions, making use of a evolutionary adjustment for the network topologies. Since then, many other works have been proposed such as the research of Miller [8] that investigates mutation operators biologicaly inspired in an evolutionary algorithm. More particularly, in the case of time series prediction, some previous works have provided different approaches for tackling the problem [9], [10], [6], [11], [12], [13].

In this paper, a systematic procedure based on a hybrid intelligent system is proposed with the capability of automatically defining both network weights and architecture, the most fitted algorithm for training the network and, in addition, the most important parameter for the prediction of time series: the relevant time lags that represent the series. The adopted method consists of a combination of a standard neural network with a modified genetic algorithm (GA) [9] which efficiently searches and defines 1. the best evolved neural network structure in terms of the number of processing units and network weights, 2. the most fitted training algorithm [14] that boosts the prediction performance, 3. the minimum number of (and the particular) temporal lags necessary to solve the problem, and 4. a behavioral statistical test carried out at the prediction model output to fix relative phase distortions in the series representation.

It is shown how this procedure can enhance prediction performance making use of a test bed composed of four relevant time series, according to a number of different performance metrics.

II. The Time Series Problem

A time series can be defined as a set of points, generally time equidistant, such as $X_t = \{x_t \in \mathbb{R} \mid t = 1, 2, 3 \ldots N\}$, where t is the temporal index and N is the number of observations. Therefore x_t is a sequence of temporal observations orderly sequenced and equally spaced.

The aim when applying prediction techniques to a given time series is to identify certain regular patterns present in historical data in order to create a model capable of generating the next temporal patterns. In this context, a crucial factor for a good forecasting performance is the correct choice of the time lags considered for representing the series. Such relational structures among historical data constitute a d-dimensional phase space, where d is the minimum dimension capable of representing such relationship. Therefore, a d-dimensional phase space can be built so that it is possible to unfold a time series in its interior. The work of F. Takens [15] has proved that if d is sufficiently large, such built phase space is

homeomorphic to the phase space which generated the time series. Thus Takens' theorem [15] has, firstly, provided the theoretical guarantee that it is possible to build a state space using the correct lags, and, secondly, that if this space is correctly rebuilt, the dynamics of this space is topologically identical to the dynamics of the original system state space.

The big problem in reconstructing the original state space is naturally the correct choice of the variable d, or, more specifically, the choice of the important time lags necessary for characterization of the system dynamics. In order to achieve such objective some tests for the verification of the dependence of the time lags such as the δ-Test method [16], and some other methods [17] based on Takens' theorem [15], can be applied. However, in general, these tests are based on the primary dependence between the variables and do not consider any possible induced dependencies. For example if $F(x_{t-1}) = F(F(x_{t-2}))$, it is said that x_{t-1} is the primary dependence, and the dependence induced on x_{t-2} is not considered (any variable without a primary dependence is denoted as irrelevant). The method proposed in this paper, conversely, does not make any prior assumption about the dependencies between the variables. In other words, it does not discard any possible correlation that can exist between the series parameters, even higher order correlations, since it carries out an iterative automatic search for finding the relevant time lags.

III. The TAEF Method

A. Introduction

The method proposed in this work — the Time-delay Added Evolutionary Forecasting (TAEF) method — is an evolution of the work first reported in [18] and tries to reconstruct the phase space of a given time series by carrying out a search for the minimum dimensionality necessary to reproduce, to a certain accuracy, the phenomenon generator of the times series.

The proposed procedure is a intelligent hybrid system based on a multilayer perceptron network (MLP) trained with a modified genetic algorithm (GA) [9] which not only searches for a number of the ANN parameters but also for the adequate embedded dimension represented in the lags. The minimal acceptable accuracy of the prediction generated is initially set by the user, but is automatically changed by the training algorithm if it finds a model with better accuracy (evolution process).

The scheme describing the proposed algorithm is based on the iterative definition of the three main elements necessary for building an accurate forecasting system: 1. the underlying information necessary to predict the series (the minimum number of time lags adequate for representing the series); 2. the structure of the model capable of representing such underlying information for the purpose of prediction (the number of units in the ANN structure); and 3. the appropriate algorithm for training the model.

It is important to consider the minimum possible number of time lags in the representation of the series because the larger the number of lags the larger the cost associated with the model training.

Following this principle, the important parameters defined by the algorithm are:

1) **The number of time lags to represent the series:** initially, a maximum number of lags (*MaxLags*) is defined by the user and a GA can choose any number of specific lags in the interval [1, *MaxLags*] for each individual of the population;
2) **The number of units in the ANN hidden layer:** the maximum number of hidden layer units (*NHiddenmax*) is determined by the user and the GA chooses, for each candidate individual, the number of units in the hidden layer (in the interval [1, *NHiddenmax*]);
3) **The training algorithm for the ANN:** RPROP, Levenberg-Marquardt, Scaled Conjugate Gradient, One Step Secant Conjugate Gradient [14] are candidates for the best algorithm for training the ANN and the GA defines one of these algorithms for each individual in the population.

B. Method Operation

The algorithm starts with the user defining a minimum initial fitness value (*MinFit*) which should be reached by at least one individual of the population in a given GA round. The fitness function is defined as,

$$Fitness = \frac{1}{1 + MSE} \tag{1}$$

where *MSE* is the Mean Squared Error of the ANN and will be formally defined in the next section.

In each GA round, a population of M individuals is generated, each of them being represented by a chromosome (in the experiments carried out here $M = 10$). Each individual is a three-layer ANN where the first layer is defined by the number of time lags, the second layer is composed of a number of hidden processing units (sigmoidal units) and the third layer consists of a linear processing unit (prediction horizon of one step ahead).

Each individual has distinct network initialization and cross validation. The stopping criteria for each one of the individual are:

- The number of Epochs: *NEpochs*;
- The increase in the validation error: *Gl*;
- The decrease in the training error: *Pt*.

The best repetition with the smallest validation error is chosen to represent the best individual. Following this procedure, the GA evolves towards a good fitness solution (which may not be the best solution yet), according to the stopping criteria:

- Number of generations created: *NGen*;
- Fitness evolution of the best individual: *BestFit*.

After this point, when the GA reaches a solution, the algorithm checks if the fitness of the best individual paired or overcame the initial value specified for the variable *MinFit* (minimum fitness). If this is not the case, the value of *MaxLags* (maximum number of lags) is increased by the unit and the

GA procedure is repeated to search for a better solution. The objective here is to increase the possible number of lags in the lag set until a solution of minimum fitness is reached.

However, if the fitness reached was satisfactory, then the algorithm checks the number of lags chosen for the best individual, places this value as *MaxLags*, sets *MinFit* with the fitness value reached by this individual, and repeats the whole GA procedure. In this case, the fitness achieved by the best individual was better than the fitness previously set and, therefore, the model can possibly generate a solution of higher accuracy with the lags of the best individual (and with the *MinFit* reached by the best individual as the new target). If, however, the new value of *MinFit* is not reached in the next round, *MaxLags* gets again the same value defined for it just before the round that found the best individual, increased by the unit (the maximum number of lags is increased by one). The idea is that if the time lags found by the best individual were not capable of producing a higher fitness than the one previously found then that may be because some important lag (or lags) was discarded. The state space for the lag search is then increased by one to allow a wider search for the definition of the lag set. This procedure goes on until the stop condition is reached. After that, the TAEF method chooses the best model found among all the candidates.

C. Method Improvement

During the development of the method, a peculiar behavior was observed in the prediction model. While the representations of some series were developed by the model with a very close approximation between the actual series and the predicted series ("in-phase" matching), the predictions of other series were always presented with a one step shift (delay) with respect to the original data ("out-of-phase" matching). This out-of-phase behavior was always found in the prediction of the financial series, whereas the in-phase matching was observed in all the other types of series (natural phenomena series). An interesting point to observe is that this one step delay behavior is similar to a random walk like model. Since it is a common sense in finance and economics that financial times series behave like random walks [19], as a first approximation, it is not strange that predictor models generated for them show this time delay distortion.

If this fact is analyzed in comparison to a random walk model, the prediction error minimization will be reached when $X_t = X_{t-1}$ — the value at the time t is equal to the value at the time $t-1$, once that the expected value of the noise (R_t) is zero.

This observation is also in accordance with some other results reported in the literature [20] which showed that predictions of financial time series represented by an ANN exhibit a characteristic one step shift with respect to the original data (out-of-phase matching). They argued that the financial series is represented by the ANN as if it were a random walk.

In any case, in order to make the TAEF method more robust for representation of any time series, another element was introduced in the method operation. After the best model is chosen when training is finished, a statistical test is employed to check if the network representation has reached an in-phase or out-of-phase matching. This is conducted by comparing the outputs of the prediction model with the actual series, making use of the validation data set. This comparison is a simple hypothesis test, where the null hypothesis is that the prediction corresponds to an in-phase matching and the alternative hypothesis is that the prediction does not correspond to an in-phase matching (or out-of-phase matching).

If this test (t-test) accepts the in-phase matching hypothesis, the elected model is ready for practical use. Otherwise, the method carries out a new procedure to adjust the relative phase between the prediction and the actual time series. The validation patterns are presented to the ANN and the output of these patterns are re-arranged to create new inputs that are both presented to the ANN and set as the output (prediction) target. Figure 1 illustrates this idea.

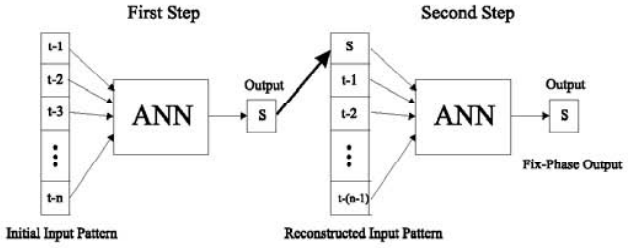

Fig. 1. Procedure to adjust the relative phase.

The approximation results for both the in-phase and out-of-phase models are measured and the best model (smaller MSE error) is elected as the final model. Figure 2 depicts the complete TAEF algorithm for the model construction.

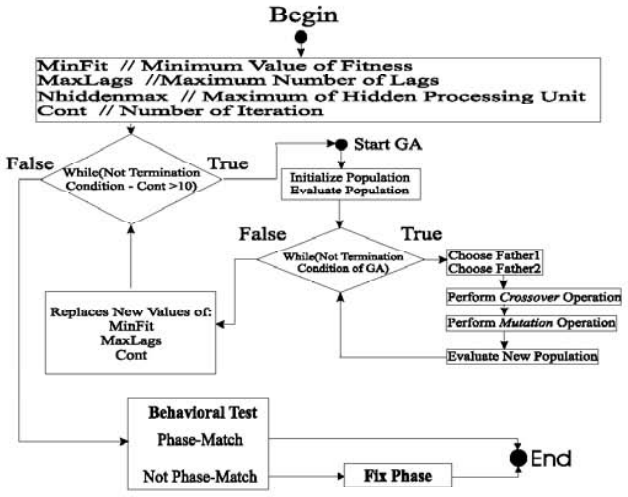

Fig. 2. The TAEF method's algorithm

The fase adjustment procedure does not assume that the ANN behave like a random walk model, but it behaves similarly to a random walk: the $t+1$ prediction is taken as

the t value.

IV. Performance Evaluation

Most of the works found in the literature of time series prediction employ only the mean squared error (MSE) as performance criterion for model evaluation:

$$MSE = \frac{1}{N}\sum_{j=1}^{N}(target_j - output_j)^2 \quad (2)$$

where N is the number of patterns, $target_j$ is the desired output for pattern j and $output_j$ is the predicted value for pattern j.

For allowing a more robust performance assertiveness[21], a set of 6 (six) additional criteria was taken into account for assessment of the method proposed. The second measure employed was the $MAPE$ (Mean Absolute Percentage Error), given by

$$MAPE = \frac{100}{N}\sum_{j=1}^{N}\left|\frac{target_j - output_j}{X_j}\right| \quad (3)$$

where N, $target_j$, and $ouytput_j$ are the same MSE parameters, and X_j is the time series at point j.

The third measure was the U of Theil Statistics, or NMSE (Normalized Mean Squared Error):

$$UofTheil = \frac{\sum_{j=1}^{N}(target_j - output_j)^2}{\sum_{j=1}^{N}(target_j - target_{j+1})^2}. \quad (4)$$

which associates the model performance with a random walk model. If the U of Theil Statistics is equal to 1, the predictor has the same performance of a random walk model. If the U of Theil Statistics is greater than 1, then the predictor has a worse performance than a Random Walk model, and if the U of Theil Statistics is less than 1, then the predictor is better than a random walk model. So, the predictor is usable if its U of Theil Statistics is less than 1, and tends to the perfect model if the U of Theil Statistics tends to zero.

The fourth metrics applied considers the correctness of Prediction of Change in Direction ($POCID$):

$$POCID = 100\frac{\sum_{j=1}^{N}D_j}{N} \quad (5)$$

where

$$D_j \begin{cases} 1 & if(target_j - target_{j-1})(output_j - output_{j-1}) > 0, \\ 0 & otherwise. \end{cases} \quad (6)$$

The fifth was the Average Relative Variance (ARV), which is given by

$$ARV = \frac{1}{N}\frac{\sum_{j=1}^{N}(output_j - target_j)^2}{\sum_{j=1}^{N}(output_j - \overline{target})^2} \quad (7)$$

where N, $target_j$, and $output_j$ are the same parameters of the other evaluation measures, and $\overline{target}$ is the time series mean. If the ARV value is equal to 1, the predictor has the same performance of calculating the mean over the series. If the ARV value is greater than 1, then the predictor performs worse than a mean time series prediction. Otherwise, if the ARV value is less than 1, then the predictor is better than a mean time series prediction. So, the predictor is usable if its ARV is less than 1, and tends to the perfect model if the ARV tends to zero.

The last two evaluation criteria corresponded to the Akaike (AIC) and the Bayesian (BIC) information which include the freedom degrees (penalizing the models with additional parameters) in the model evaluation. The AIC and BIC are approximated by

$$AIC = N\ln(MSE) + 2p \quad (8)$$

$$BIC = N\ln(MSE) + p + N\ln(p) \quad (9)$$

where N is the number of time series points, MSE is the Mean Squared Error and p is the number of freedom degrees.

V. Experimental Results

A set of four relevant times series was employed for evaluation of the method proposed. Two of these series are artificial — the series of Henon Map and GARCH Model series, and the other series were drawn from real world situations: Sunspot and Nasdaq Index.

All the series investigated were normalized to lie within the interval [0,1] and divided in training set (50% of the data), validation set (25% of the data) and test set (25% of the data). The GA parameters were the same for all the series with a mutation probability of 10%. For all the experiments carried out, the following system parameters were employed:

- Initialization parameters:
  1) $MinFit = 0.99$ ($\sim$ 1% of error);
  2) $MaxLags = 4$;
  3) $NHiddenmax = 20$.
- Stopping conditions for the GA:
  1) $NGen = 1000$;
  2) $BestFit =< 10^{-4}$.
- Stopping conditions for each individual:
  1) $NEpochs = 1000$;
  2) $Gl -< 5\%$;
  3) $Pt =< 10^{-6}$.

In addition, experiments with standard multi-layer perceptron (MLP) networks were used for comparison with the TAEF method. Several architectures were examined within these experiments, with 10 (ten) random initializations for each ANN architecture attempted. For all the cases, the Levenberg-Marquardt Algorithm was employed for network training. For all the series, the best ANN initialization was elected as the ANN model to be beaten. The statistical behavioral test for phase correction was also applied to the standard ANN model to guarantee a fair comparison between the models.

A. Henon Map Series

The Henon series is a very popular example of time series due to its complex nature and chaotic dynamics. An interesting work that employed this series was conducted by D.B.Murray [22]. Such as in the present work, Murray was interested in proposing a model to represent the phase space of the temporal lags. He developed his approach based on the idea of building this phase space of embedded dimensions from a metric tensor whose components are adjusted in order to the minimize the prediction error (root mean square error, RMSE). The best prediction results obtained by Murray corresponded to 3.7e-3 (RMSE), or 1.4e-5 (1.4e-3 %), if considered the mean squared error (MSE) .

The Henon series considered in this work was the same as that used by Murray, being composed of 10.000 points generated from Equation (10) with parameters $a = 1.4$ and $b = 0.3$. This series is generated without the inclusion of any noise (the r terms are null) .

$$X_t = 1 - a(X_{t-2} - r_{t-2})^2 + b(X_{t-4} - r_{t-4}) + r_t \quad (10)$$

For the prediction of Henon Map series (1 step ahead prediction), the TAEF method identified the lags 2, 3, 5 and 7 as the relevant to the problem, defined 14 processing units in the hidden layer of the network, elected the Levenberg-Marquardt algorithm as the most fitted for the ANN training and classified the prediction model as "in-phase" matching. In the ANN experiments, the architectures used were 4-1-1, 4-5-1 and 4-10-1, where the best model was 4-10-1. Table I shows the results (best standard ANN model and TAEF method) with all the performance measures presented in Section IV for both cases: "in-phase" matching and if the prediction model had been chosen as "out-of-phase" matching.

TABLE I
EXPERIMENTAL RESULTS FOR THE HENON SERIES

Performance Measure	ANN Model		TAEF Method	
	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase
MSE	4.0840e-9	0.1588	3.1678e-11	1.0445
MAPE (%)	1.0349e-10	1227.5241	0.0061	305.0857
Theil	3.9090e-10	0.9998	1.9836e-10	0.9996
POCID (%)	100.00	42.3823	100.00	41.5064
ARV	7.8006e-11	2.6363	1.3932e-10	1.3187
AIC	-59657.5	-8746.5570	-55250.9695	556.6590
BIC	-59241.3	-7062.8850	-53722.5625	2084.9763

Figure 3 shows a comparative graph of the actual Henon series (solid lines) and the prediction generated by the TAEF method (dash lines) for the last 100 points of the test set, for both cases of prediction hypotheses (in-phase matching and out-of-phase matching).

Observing the performance measures (Table I) and the prediction graphs (Figure 3), it is possible to notice that the TAEF method correctly classified the Henon series in the in-phase matching category.

B. Sunspot Series

The sunspot series used consisted of the total annual measures of the sun spots from the years 1700 to 1988, generating a database of 289 examples.

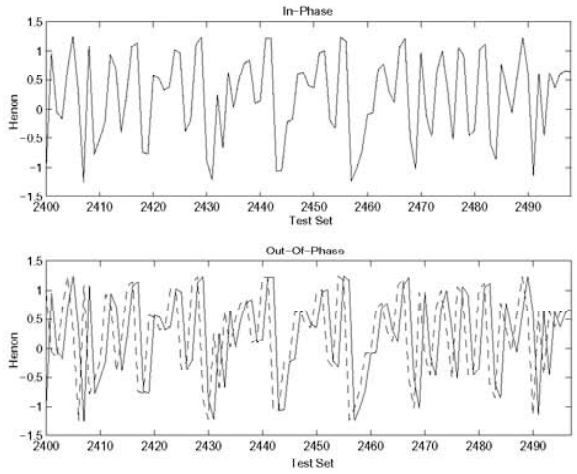

Fig. 3. Prediction results for the Henon series (last 100 points of the test set) – actual values (solid lines) and predicted values (dashed lines) for both cases: "in-phase" matching and "out-of-phase".

N.Terui and H.K.Van Dijk [23] developed a work where a combination of some linear and non-linear models were employed for times series prediction. Among the series investigated, Terui and Van Dijk employed the sunspot series from the years 1720 to 1989 to test their method based on the combination of the AR, TAR and ExpAR models [23]. The best experimental results reported with their proposed method (best model combination) corresponded to an MSE error of 0.0390.

For the prediction of the Sunspot series (1 step ahead prediction), the TAEF method identified again the lags 1 to 4 as the relevant to the problem, defined 4 processing units in the hidden layer of the network, elected the Levenberg-Marquardt algorithm as the most fitted for the ANN training and classified the model as "in-phase" matching. In the experimentes with ANN models were examined the architectures: 3-1-1, 3-5-1 and 3-10-1, where the best model was 3-5-1. The Table II shows the results (the best ANN model and TAEF method) with all the performance measures for both cases: "in-phase" matching and if the prediction model had been chosen as "out-of-phase" matching.

TABLE II
EXPERIMENTAL RESULTS FOR THE SUNSPOT SERIES

Performance Measure	ANN Model		TAEF Method	
	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase
MSE	0.9205	1.0163	0.0070	0.0307
MAPE (%)	2.41	133.5613	30.0661	82.5523
Theil	0.3443	1.3295	0.1763	1.2225
POCID (%)	90.00	61.7021	84.0580	65.2174
ARV	0.1418	0.3020	0.1233	0.4125
AIC	-280.9	76.9087	-321.5201	-195.8776
BIC	-196.0	81.9688	-305.5321	-117.9137

Figure 4 shows a comparative graph of the actual Sunspot series (solid lines) and the prediction generated by the TAEF method (dashed lines) for the test set, for both cases of

prediction hypotheses (in-phase matching and out-of-phase matching).

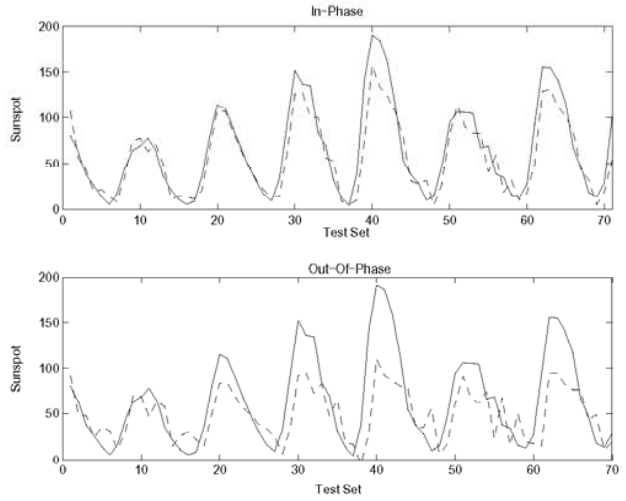

Fig. 4. Prediction results for the Sunspot series (test set) – actual values (solid lines) and predicted values (dashed lines) for both cases: "in-phase" matching and "out-of-phase".

Observing the Table II and the predictions graphs of Figure 4 it is possible to notice that the method correctly classified the Sunspot series in the in-phase matching category.

C. Nasdaq Index Series

The Nasdaq series (National Association of Securities Dealers Automated Quotation) corresponds to daily observations from 2nd February 1971 to 18th of June 2004 of the Nasdaq index (8428 points).

For the Nasdaq Index Series the TAEF Method identified the lags 3, 4, 6 and 8 as the relevant to the problem, defined 11 processing units in the hidden layer of the network, elected the Levenberg-Marquardt algorithm as the most fitted for the ANN training and classified the model as "out-of-phase" matching. In the experimentes with the standard ANN models the following architectures were examined : 3-1-1, 3-5-1 and 3-10-1, where the best model was 3-5-1. Table III shows the results with all the performance measures for both cases: "out-of-phase" matching and if the prediction model had been chosen as "in-phase" matching. Of particular interest to this financial series are the values shown by the Statistics U of Theil (0.17), by the POCID (89.6%) and by the ARV (0.0005) which show that the "out-of-phase" hypothesis produces far better results than those given by the "in-phase" hypothesis.

Figure 5 shows a comparative graph of the actual Nasdaq series (solid lines) and the prediction generated by the method (dash lines) for the last 100 points of the test set. Once again it is possible to notice that the prediction generated in the out-of-phase matching hypothesis is not delayed with respect to the original data and that the "out-of-phase" model pointed out by the method was the correct choice.

TABLE III
EXPERIMENTAL RESULTS FOR THE NASDAQ SERIES

Performance Measure	ANN Model		TAEF Method	
	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase
MSE	0.0022	0.0023	2.1449e-5	32374e-6
MAPE (%)	2.6988e-3	2.7001e-3	0.2012	0.0774
Theil	1.1728	1.1759	1.1441	0.1726
POCID (%)	53.0641	53.9458	52.7091	89.6338
ARV	3.4977e-3	3.5011e-3	0.0034	5.1500e-4
AIC	-22536.0	-22536.4	-22342.4331	-26310.0737
BIC	-22363.1	-22363.9	-21391.1870	-25358.8956

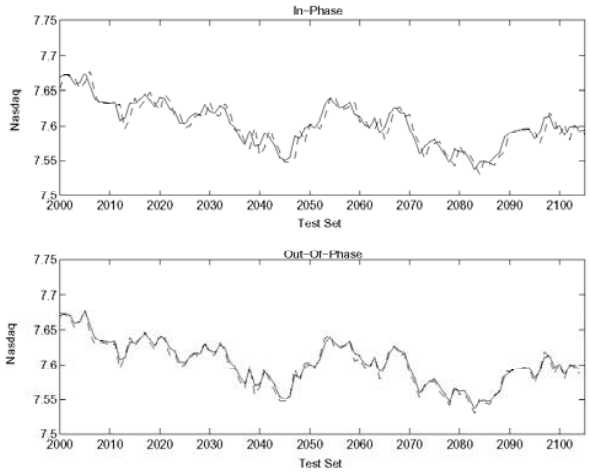

Fig. 5. Prediction results for the Nasdaq Index Series (test set) – actual values (solid lines) and predicted values (dashed lines) for both cases: "in-phase" matching and "out-of-phase" matching.

D. GARCH Model Series

The General Autoregressive Conditional Heteroscedasticity (GARCH) [24] is a model that, loosely speaking, is related to a time-varying variance function, i.e., volatility. The conditional term indicates a dependency on the observations of the immediate past, and the autoregressive term describes a feedback mechanism that incorporates past observations into the present. This model is a time-series technique that allows users to model the serial dependency of volatility.

Bollerslev [24] developed the GARCH Model as a generalization of Engle's [25] original ARCH volatility modeling technique. Bollerslev [24] designed the GARCH model to offer a more parsimonious model with less computational cost.

The GARCH models are usually applied to return series, where financial decisions are rarely based solely on expected returns and volatilities. The return series is simply the difference between the series value at time t and the series value at time $t-1$.

Let the return series be:

$$x_t = C + \varepsilon_t \tag{11}$$

where C is a constant and ε_t is a white noise disturbance. The conditional variance of this innovation (x_t) is, by definition,

$$Var_{t-1}(x_t) = E_{t-1}(\varepsilon_t^2) = \sigma_t^2 \tag{12}$$

where E_{t-1} denotes the expected value conditional on the past, of the process, along with any other information available at the time $t-1$.

The general GARCH(P,Q) model for the conditional variance of innovations is,

$$\sigma_t^2 = \kappa + \sum_{i=1}^{P} g_i \sigma_{t-i}^2 + \sum_{j=1}^{Q} a_j \varepsilon_{t-j}^2 \tag{13}$$

with the constraints,

$$\sum_{i=1}^{P} g_i + \sum_{j=1}^{Q} a_j < 1;$$

$$\kappa > 0;$$

$$g_i \geq 0 \qquad for \quad i = 1, 2, 3, \ldots, P;$$

$$a_j \geq 0 \qquad for \quad j = 1, 2, 3, \ldots, Q$$

Two GARCH models were set up for experimentation: GARCH(1,1) and GARCH(1,2). According to these models, a data set with 1000 points was created and times series were built based on the return series. The initial time series point is $x_0 = 0$, and other points are given by,

$$x_t = Return_t + x_{t-1} \tag{14}$$

where $Return_t$ is the point at time t of the return series.

In Table IV the parameters elected by the TAEF method are shown for each one of the GARCH series, where the first column presents the series and the next columns show the parameters chosen by the method. The last column shows the series classification (in-phase matching or out-of-phase matching).

TABLE IV
CONFIGURATIONS REACHED BY THE TAEF METHOD FOR THE GARCH MODEL TIMES SERIES.

Serie	Lags	Hidden Units	Algorithm	Classification
GARCH(1,1)	4,6,8,10,11	15	RProP	Out-Of-Phase
GARCH(1,2)	2,3,4,5	8	Scaled Conjugate Gradient	Out-Of-Phase

In Table V the results with all the performance measures are shown for all the GARCH series for both cases: out-of-phase matching and "in-phase"matching for the test set. Figures 6 and 7 present the comparative graphs of the actual GARCH series (solid lines) and the prediction generated by the TAEF method (dashed lines) for the test set, for both cases of prediction hypotheses (in-phase matching and out-of-phase matching).

VI. CONCLUSIONS

This paper has presented a hybrid system for application in time series forecasting problems which consists of an ANN combined with a modified genetic algorithm and a behavioral test of phase matching hypothesis carried out at the model's output.

The experimental results using seven different metrics (MSE, MAPE, U of Theil Statistics, POCID, ARV, AIC and BIC) showed that this system can boost the performance of time series prediction on both artificially generated time series and real world (financial market and natural phenomena) time series. The experimental validation of the method was carried out on some complex and relevant time series and were compared to standard MLPs in the same conditions: two real world time series, the Henon Map series (with its non-linear relations and chaotic characteristics), and the artificially generated GARCH Model series.

With the introduction of the behavioral test for identifying the best prediction model: "in-phase" or "out-of-phase", the TAEF method was able to classify if a given time series tends or not to a Random Walk like model, thus adjusting the model if necessary. Such adjustment is automatically

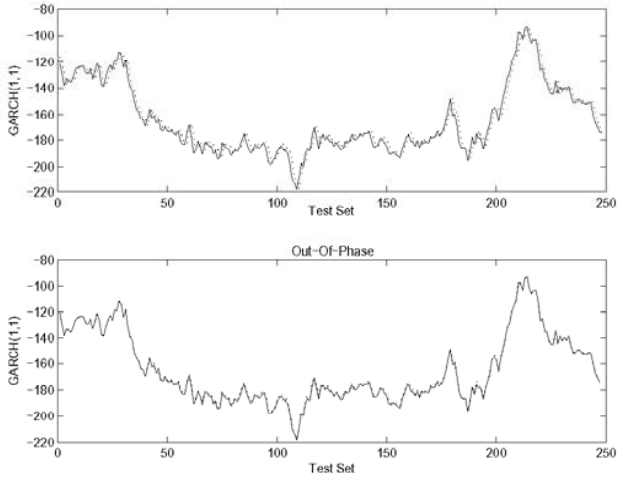

Fig. 6. GARCH(1,1) Model. The graph shows the In-Phase Hypothesis (top) and Out-Of-Phase Hypothesis (bottom) for the test set. Solid lines represent real data and dashed lines the predicted data.

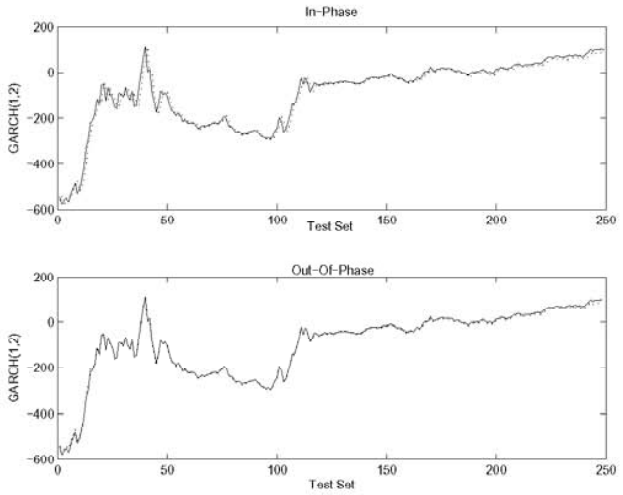

Fig. 7. GARCH(1,2) Model. The graph shows the In-Phase Hypothesis (top) and Out-Of-Phase Hypothesis (bottom) for the test set. Solid lines represent real data and dashed lines the predicted data.

TABLE V

RESULTS FOR THE GARCH MODEL TIMES SERIES

Series	Hypotheses	Performance Measure						
		MSE	MAPE	U of Theil	POCID	ARV	AIC	BIC
GARHC(1,1)	In-Phase	0.0005	10.96%	0.3187	51.82%	0.0424	-1653.4	-204.5
	Out-Of-Phase	$2.100 \cdot 10^{-5}$	2.58%	0.0403	95.53%	0.0017	-2447.4	-1969.4
GARHC(1,2)	In-Phase	0.0009	3.28%	3.2783	50.00%	0.0258	-1650.8	-1227.4
	Out-Of-Phase	$6.0000 \cdot 10^{-5}$	1.03%	0.0643	97.16%	0.0017	-2314.6	-2093.4

conducted without the use of any additional training phase nor the use of any additional training data (the same original validation data is employed). Only one additional epoch is used for presenting the original validation data and deciding which of the models generated (in-phase or out-of-phase) produces the best approximation.

The out-of-phase behavior reached by the ANN appears mostly when the time series is a financial series, an economical series, or (as a first approximation) a random walk like model. If the time series is generated by natural phenomena, the choice of in-phase matching is reached by the ANN. Although the out-of-phase behavior is characterized by a prediction shift (delay) with respect to original data (and this is a random walk like behavior), the ANN generated by TAEF method is not a random walk model (although behaves with the same behavior). This affirmation is supported by the phase fix procedure. If the ANN was a real random walk model, the phase fix procedure would generate the same result of the original prediction, because in the random walk model the $t+1$ value is always the t value. Why the ANN has this peculiar behavior is a mystery to us at the moment, and studies are being accomplished to explain such behavior.

When compared to the best results found with standard MLPs and in the literature, the TAEF method presented a superior performance in all the comparisons made. A further study is being conducted to determine any possible limitations of the method when dealing with other types of components found in other different real world time series such as trends, seasonality, impulses, steps, model exchange and other non-linearities.

REFERENCES

[1] T. S. Rao and M. M. Gabr, *Introduction to Bispectral Analysis and Bilinear Time Series Models*, ser. Lecture Notes in Statistics. Berlin: Springer, 1984, vol. 24.

[2] T. Ozaki, *Nonlinear Time Series Models and Dynamical Systems*, ser. Hand Books of Statistics, E. J. Hannan and P. R. K. eds, Eds. Amsterdam: Noth-Holland, 1985, vol. 5.

[3] M. B. Priestley, *Non-Linear and Non-Stacionary time Series Analysis*. London: Academic Press, 1988.

[4] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing*. The MIT Press, 1986, vol. 1: Foundations.

[5] G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," *International Journal of Forecasting*, vol. 14, pp. 35–62, 1998.

[6] X. Yao, "Evolving artificial neural networks," *Proceedings of IEEE*, vol. 87, no. 9, pp. 1423–1447, September 1999.

[7] K. O. Stanley and R. Miikkulainen, "Evolving neral networks though augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[8] D. A. Miller, R. Arquello, and G. W. Greenwood, "Evolving artificial neural ntwork structures: Experimental results for biologivally inspired adaptive mutations," in *Proceedings of CEC2004 - Congress on Evolutionary Computation*, vol. 2. IEEE, 2004, pp. 2114–2119.

[9] F. H. F. Leung, H. K. Lam, S. H. Ling, and P. K. S. Tam, "Tuning of the structure and parametrs of the neural network using an improved genetic algorithm," *IEEE Transaction on Neural Networks*, vol. 14, no. 1, pp. 79–88, January 2003.

[10] M. M. Islan, X. Yao, and K. Muraz, "A construtive algorithm for training cooperative neural networks ensembles," *IEEE Transactions on Neural Networks*, vol. 14, no. 4, pp. 820–834, july 2003.

[11] J. Binner, G. Kendall, and A. Gazely, "Evolving neural networks with evolutionary strategies: A new application to divisia money." *Advances in Econometrics*, vol. 19, pp. 127–143, 2004.

[12] A. S. Andreou, E. F. Georgopolulos, and S. D. Likothanassis, "Exchange-rates forecasting: A hybrid algorithm based on genetically optimized adaptive neural networks," *Computational Economics*, vol. 20, no. 3, pp. 191–210, 2002.

[13] M. Matilla-García and C. Argüello, "A hybrid approach based on neural networks and genetic algorithms to the study of profitability in the spanish stock market," *Applied Economics Letters*, vol. 12, no. 5, pp. 303–308, April 2005.

[14] S. Haykins, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Princete Hall, 1998.

[15] F. Takens, "Detecting strange attractor in turbulence," in *Dynamical Systems and Turbulence*, ser. Lect. Notes in Maths., A. Dold and B. Eckmann, Eds., vol. 898. New York: Springer-Verlag, 1980, pp. 366–381.

[16] H. Pi and C. Peterson, "Finding the embedding dimension and variable dependences in time series," *Neural Computation*, vol. 6, pp. 509–520, 1994.

[17] N. Tanaka, H. Okamoto, and M. Naito, "Estimating the active dimension of the dynamics in a time series based on a information criterion," *Physica D*, vol. 158, pp. 19–31, 2001.

[18] T. A. E. Ferreira, G. C. Vasconcelos, and P. J. L. Adeodato, "A hybrid intelligence system approach for improving the prediction of real world time series," in *IEEE Proceedings of the Congress on Evolutionary Computation*, vol. 1. Portland, Oregon: IEEE, 2004, pp. 736–743.

[19] T. C. Mills, *The Econometric Modelling of Financial Time Series*. Cambridge: Cambridge University Press, 2003.

[20] R. Sitte and J. Sitte, "Neural networks approach to the random walk dilemma of financial time series," *Applied Intelligence*, vol. 16, no. 3, pp. 163–171, May 2002.

[21] M. P. Clementes and D. F. Hendry, "On the limitations of comparing mean squared forecast erros," *Journal of Forecasting*, vol. 12, no. 8, pp. 617–637, 1993.

[22] D. B. Murray, "Forecasting a chaotic time series using an improved metric for embedding space," *Physica D*, vol. 68, pp. 318–325, 1993.

[23] N. Teru and H. K. V. Dijk, "Combined forecasts form linear and nonlinear time series models," *Intenational Journal of Forecasting*, vol. 18, pp. 421–438, 2002.

[24] T. Bollerslev, "Generalized autoregressive condicional heteroskedasticity," *Journal of Econometrics*, vol. 31, pp. 307–327, 1986.

[25] R. F. Engle, "Autoregressive conditional heteroskedasticity with estimates of the variance of uk onflation," *Econometrica*, vol. 50, pp. 987–1008, 1982.