



ELSEVIER

Expert Systems with Applications xx (0000) xxx–xxx

Expert Systems  
with Applications

www.elsevier.com/locate/eswa

# Predicting the exchange traded fund DIA with a combination of genetic algorithms and neural networks

Massimiliano Versace\*, Rushi Bhatt<sup>1</sup>, Oliver Hinds<sup>2</sup>, Mark Shiffer<sup>3</sup>

*Department of Cognitive and Neural Systems, Boston University, Boston, MA 02215, USA*

## Abstract

We evaluate the performance of a heterogeneous mixture of neural network algorithms for predicting the exchange-traded fund DIA. A genetic algorithm is utilized to find the best mixture of neural networks, the topology of individual networks in the ensemble, and to determine the features set. The Genetic Algorithm also determines the window size of the input time-series supplied to the individual classifiers in the mixture of experts. The mixtures of neural network experts consist of recurrent back-propagation networks, and Radial Basis Function networks. The application of Genetic Algorithm on the heterogeneous mixture of powerful neural network architectures shows promise for prediction of stock market time series. These highly non-linear, stochastic and highly non-stationary time series have been found to be notoriously difficult to predict using conventional linear statistical methods. In this paper, we propose a biologically inspired methodology to tackle such hard problems using a multi-faceted solution.

© 2004 Elsevier Ltd. All rights reserved.

*Keywords:* Time series prediction; Financial forecasting; Genetic algorithms; Neural networks

## 1. Introduction

During the last few years, financial institutions as well as individual investors have found a wide range of uses for Artificial Intelligence (AI) technologies such as expert systems, artificial neural networks (ANNs), genetic algorithms (GAs), and fuzzy logic. Financial engineering has therefore become the natural theatre where pattern classification algorithms can be contrasted and tested on highly demanding grounds. The main rationale motivating the use of these techniques is to identify and exploit the regularity that is hidden in the apparently chaotic price trend of a given security. Parallel to the development of AI technologies, the last decade has also witnessed the development of several nonlinear time series models (Granger & Anderson, 1978, Tong & Lim, 1980, Engle, 1982). These nonlinear models, although not suffering from a priori assumption of a linear relationship between data and time series, are limited by the fact that an a priori assumption of the nature of the nonlinear

relationship must be formulated, the latter task being even more demanding than the linear case.

Within the wider field of AI, ANNs have been demonstrated to be a natural solution to problems where an explicit description of the nature of the data is not available due to their capability to generate data-driven hypotheses. Historically, the development of these algorithms was inspired by investigations into the functioning of the nervous system. ANNs have subsequently been found to have sound theoretical basis from the perspective of statistical learning theory, and they usually yield good performance when used for real-world data analysis or in predicting nonlinear dynamical systems (Lapedes & Farber, 1987, 1988; Haykin, 1999; Duda, Hart, & Stork, 2000). ANNs have good generalization capabilities and are usually robust against noisy or missing data, all of which are highly desirable properties time series prediction. Importantly, the class of unsupervised neural networks can operate a much richer class of functions compared to linear regression based techniques. Therefore, unsupervised neural networks have the capability of discovering associations between features that may not have been expected or looked for.

There is an extensive literature on financial applications of ANNs (Trippi & Turban, 1993; Azoff, 1994; Refenes, 1995; Gately, 1996; Odom & Sharda, 1990; Coleman, Graettinger, & Lawrence 1991; Salchenker, Vinar, & Lash, 1992;

\* Corresponding author. Tel.: +617-353-6426; fax: +617-353-7755.

*E-mail addresses:* vrsace@cns.bu.edu (M. Versace); rushi@bu.edu (R. Bhatt); oph@bu.edu (O. Hinds); mshiffer@cns.edu (M. Shiffer).

<sup>1</sup> Tel.: +617-353-5235; fax: +617-353-7755.

<sup>2</sup> Tel.: +617-353-6426; fax: +617-353-7755.

<sup>3</sup> Tel.: +617-353-6181; fax: +617-353-7755.

113 Tam & Kiang, 1992; Wilson & Sharda, 1994, Weigend,  
 114 Rumelhart, & Hubermann, 1992; Zhang, 1998, for a review).  
 115 Although encouraging results have been reported in which  
 116 ANNs-based systems outperformed widely-used well-established  
 117 statistical methods, many inconsistent reports have  
 118 been undermining the robustness of these findings. Among  
 119 the reasons of these discrepancies are well-known problems  
 120 that characterize ANNs, in particular:

- 121
- 122 (1) Different network type (linear filters, multilayer percep-
- 123 trons, radial basis functions networks, or RBF, and self-
- 124 organizing maps, among others) can lead to different
- 125 results when trained and tested on the same database.
- 126 This is mainly due to the different classes of decision
- 127 boundaries that different ANN types prefer;
- 128 (2) For a given network type, ANNs are sensitive to the
- 129 choices topology and size for a given data set;
- 130 (3) ANNs are prone to overfitting, unless great care is taken
- 131 in choosing the size and connectivity of the network;
- 132 (4) The highly variable nature of financial time series can
- 133 prevent a single ANN from producing accurate fore-
- 134 casts for an extended trading period, even though it can
- 135 perform well above chance in a given testing set.

136  
 137 Combining classifiers and boosting methods often lead to  
 138 improvement in performance over single neural networks.  
 139 Several studies (Pelikan, de Groot, & Wurtz, 1992;  
 140 Ginzburg & Horn, 1994; Zhang, 1994; Chu & Widjaja,  
 141 1994) have shown improvement of performance in  
 142 combining different ANNs or training similar ANNs on  
 143 different features of the input data and then recombining  
 144 their output in a later stage. Our usage of a mixture of  
 145 experts is motivated by the hypothesis that a particularly  
 146 difficult problem can be broken into smaller sub-problems  
 147 which are easier to solve with a single classifier. This divide  
 148 et impera principle, so common in computer science, is  
 149 particularly suited for difficult problems, like modelling of  
 150 financial time series. The mixture of experts' framework  
 151 specifies that a prediction is made up of a series of  
 152 predictions from separate experts, or networks. In general,  
 153 the final output is decided by a function that combines  
 154 individual classification outputs:

$$155 \quad O = f(\vec{y}) \quad (1)$$

156  
 157 where  $\vec{y}$  is a vector of individual classifier output, and  $O$   
 158 is the output of the mixture of experts. A common  
 159 implementation of a mixture of expert uses a weighted  
 160 combination of  $y_i$  :

$$161 \quad \hat{y} = \sum_{i=1}^m g_i \hat{y}_i \quad (2)$$

162  
 163 where  $m$  is the number of experts in the model,  $\hat{y}_i$  is the  
 164 prediction of the expert  $i$  and  $g_i$  is the weighting on expert  $i$ .  
 165 This weighting can itself be the prediction of another model,  
 166 known as the gate, or can simply be set to 1 for each

167 classifier. The results of the application of mixture of  
 168 experts are encouraging, with these systems often out-  
 169 performing standard statistical techniques and other classi-  
 170 fication methods, although it is difficult to say in general in  
 171 what kind of tasks mixtures-of-experts or any other kind of  
 172 model will perform well on any given dataset (Kang & Oh,  
 173 1997, 2000; Jordan & Jacobs, 1994; Jordan & Xu, 1995;  
 174 Waterhouse, 2002).

175  
 176 In the ANNs literature, inappropriate topology selection  
 177 and weight training are frequently blamed for poor  
 178 performance. Increasing the number of hidden layer neurons  
 179 helps improving network performance, yet many problems  
 180 could be solved with very few neurons if only the network  
 181 took its optimal configuration. Due to the large size of the  
 182 hypothesis space that these ANNs are capable of generating,  
 183 it is difficult to guarantee that an ANN will converge to a  
 184 globally optimal hypothesis for the distribution of the  
 185 underlying the given dataset.

186  
 187 The inherent nonlinearity of ANN results in the existence  
 188 of many sub-optimal networks, and the great majority of  
 189 training algorithms converge to these sub-optimal configura-  
 190 tions (local minima). Solutions to local minima problems  
 191 often imply methods that are probabilistic in their nature.  
 192 The methods can in fact find the globally optimal solution  
 193 with a certain probability, which usually depends on the  
 194 number of iterations of the algorithm. At the same time,  
 195 the danger of overfitting is always present, rendering the  
 196 problem of the solution of an optimal network configuration  
 197 very difficult.

198  
 199 To summarize, there are multiple factors that influence  
 200 the choice of a given system of networks, and within a given  
 201 network type there are multiple combinations of parameters,  
 202 network architecture, activation and learning functions,  
 203 input selection and preprocessing that produce a combina-

204 torial explosion of possible systems.  
 205 A natural solution for searching this very large space of  
 206 system configuration is provided by GAs. GAs are a class of  
 207 probabilistic search techniques that has been developed in  
 208 the past decades as a general-purpose optimization tool  
 209 (Holland, 1975; Goldberg, 1989). GAs mimic biological  
 210 evolution by using a massively parallel search mechanism  
 211 that involves (a) the initialization of a random population of  
 212 systems (or candidate solutions) (b) ordering the systems  
 213 based on a measure of success in approximating the desired  
 214 solution (fitness) (c) a reproduction stage, in which the best  
 215 exemplars of the last generation have the chance to produce  
 216 an offspring through the application of some GA operators,  
 217 namely crossover and mutation. GAs are suited for  
 218 particularly hard problems when little or no knowledge of  
 219 the optimal function is given and the search space is very  
 220 large. GAs are thus an ideal tool for solving the problem if  
 221 discovering appropriate parameterizations for ANNs, and  
 222 good results have been obtained in combining GAs and  
 223 ANNs in hybrid systems (Belew, McInernay, & Schraudolph,  
 224 1990; Montana & Davis, 1989; Shaffer, Whitely, &  
 225 Eshelman, 1990; Chang & Lippmann, 1991; Harp & Samad,

1991; Miller, Todd, & Hedge, 1989; Whitley, 1989; Kingdon, 1997; Venkatesan & Kumar, 2002).

This paper presents an application of such a hybrid system that uses GAs for selecting an appropriate combination of networks, parameters and training regimen for predicting the direction of variation of the closing price of an exchange traded fund, DIA. The ‘Diamonds’ (AMEX ticker: DIA) is an exchange traded fund tracking the 30 corporations of the Dow Jones Industrial Average. The price of DIA, which is worth about 1/100th of the Dow Jones’ value, faithfully tracks the fluctuations in the Industrial Average. We have decided to concentrate on this stock because DIA is a readily available instrument for trading on the Dow Jones Industrial Average, a canonical financial index.

Section 2 of this paper presents a description of the data set employed to predict the daily direction of variation of the DIA closing price. In Section 3 we describe the prediction model, which embodies a combination of GA and ANNs. In Section 4 we analyze the performance of the model on 63 test trading days on which the ANNs of the model have not been trained. Finally, in Section 5 we make some concluding remarks.

## 2. The input data

### 2.1. Data collection

Data selection and preprocessing constitute a crucial step in any modeling effort. The phases of data preparation can be

Table 1  
The raw data loaded from Yahoo! database

Type	Name	Ticker (yahoo!)	Data structure
Stock	DIA (to be predicted)	dia	Open-High-Low-Close-Volume
Index	Nasdaq	^IXIC	Open-High-Low-Close-Volume
	S.Poor 500	^GSPC	Open-High-Low-Close-Volume
	Dow Jones Ind	^DJI	Open-High-Low-Close-Volume
	Dow Jones Trasp	^DJT	Open-High-Low-Close-Volume
	Dow Jones Util	^DJU	Open-High-Low-Close-Volume
	Dow Jones Compos	^DJA	Open-High-Low-Close-Volume
	Nikkel	^N225	Open-High-Low-Close
	Bovespa	^BVSP	Open-High-Low-Close
Currency	Dax	^GDAX	Open-High-Low-Close
	Ftse 100	^FTSE	Open-High-Low-Close
	Dollar/yen	^CJJ	Open-High-Low-Close
	Dollar/Swiss Frank	USDCHF = X	Open-High-Low-Close
Bonds	T-Bond 30 years	^TYX	Open-High-Low-Close
	T-Bond 10 years	^TNX	Open-High-Low-Close
	T-Bond 5 years	^FVX	Open-High-Low-Close
	T-Bond 13 years	^IRX	Open-High-Low-Close
	Eurobond	EUROD	Open-High-Low-Close-Volume
Commodities	Gold	^XAU	Open-High-Low-Close-Volume

The to-be-predicted stock, DIA, is loaded along with other historical values of indices, currencies, bonds and commodities. The sampling interval is from the 11th of November 2001 through the 12th of February 2003 (320 total trading days). The raw input matrix consists of 85 indicators and 320 data points.

broadly classified into three distinct areas: variable selection and collection, data inspection, and data pre-processing. Since our goal was to trade DIA frequently, we were able to narrow the list of choices to those variables relevant to the time frame. The data were obtained using a java-based query interface into a database of historical stock data available freely from Yahoo! (<http://finance.yahoo.com>).

A list of the variables employed in this study is shown in Table 1. The data were collected over the period from 11th November 2001 through 12th February 2003 (320 total trading days). All data were crosschecked to ensure accuracy. We have selected this data in order to provide a predictor model with a diversified database on which the ANNs can learn to extract input-output dependencies. We have decided to incorporate some of the major stock indices, as well as currency, bonds and gold prices, leaving the ‘decision’ of which indicators (or which non-linear combination of them) is predictive of the direction of variation of DIA to the model.

### 2.2. Data inspection

Even though the data were obtained from a reliable source, significant errors and missing data were present in the database. The validity of the input data was checked both visually and statistically. Missing data (holidays) were filled-in with the last trading day available. This filling-in has negligible consequences on the final performance of the system, since the data are pre-processed in the form of a normalized difference between the actual value and its moving average. This pre-processing allows normalizing

337 the indicator, therefore eliminating the bias given by its  
338 absolute magnitude in favor of a measure of its relative  
339 variation. This procedure allows using relatively long time  
340 series without the risk of overweighting the data with the  
341 higher absolute value.

### 342 2.3. Data pre-processing

343 In order to transform the data into a format acceptable by  
344 the prediction algorithms, derive a measure of the statistical  
345 fluctuations in the feature vectors used therein, and render  
346 the feature vectors independently of the absolute value  
347 assumed by the underlying indices from which the data are  
348 derived, two preprocessing operators were applied to the  
349 data. The renormalization operator (RNO) is used in some  
350 cases to extract daily changes in indicators that are  
351 independent of the absolute magnitude of the indicator  
352 over time and to transform the fluctuations into percentages  
353 at others. The RNO operator is specified by

$$354 \frac{(x - y)}{y} 100 \quad (3)$$

355 where  $x, y$  and  $z$  take values specific to the indicator operated  
356 on. The Fluctuation Sensitive Function (FSF) is used when  
357 the absolute magnitude of an indicator should be retained  
358 for the prediction algorithms. This operator is simply  
359 specified by  $x - y$ .

360 After the raw data listed in Table 1 were loaded and  
361 crosschecked for accuracy, each field of this dataset, called  
362 dataset  $A$ , was independently normalized using the RNO  
363 operator. The data in the dataset  $A$  were further pre-  
364 processed and saved in a second dataset called dataset  $B$ , in  
365 order to reduce the learning load of the system and allowing  
366 the learning algorithms to concentrate the more predictive  
367 portions of the input (Table 2). This procedure spares the  
368 networks from extracting knowledge that can be provided  
369 directly in the input pattern, thereby allowing the system to  
370 extract higher-order relationships between these complex  
371 variables. The values indicated in Table 1 were prepro-  
372 cessed and re-arranged in the final input matrix, consisting  
373 of 64 data types and 320 data points. The pre-processing  
374 performed on the raw data is described in Tables 2 and 3. In  
375 general, the to-be-predicted stock data was extensively  
376 preprocessed, adopting a percentage measure of the  
377 difference between the indicator (Open, High, Low, Close,  
378 Volume) and its moving average (10 days), among other  
379 measurements of variation like Rate of Change (ROC),  
380 percentage difference between Open and Close, etc. Some  
381 widely used Technical Analysis indicators were also  
382 employed, namely Moving Average Convergence/Diver-  
383 gence (MACD), Relative Strength Index (RSI), Chaikin  
384 Volatility and MOMENTUM (Table 2). For the other  
385 indicators a measure of the variation given by Open, High,  
386 Low, and Close indicators and a Moving Average was used.  
387 More details on the nature of the pre-processing are reported  
388 in Tables 2 and 3. This new input matrix was further

393 independently normalized for each field, resulting in each  
394 column assuming a value between 0 and 1. At the end of  
395 pre-processing, the absolute magnitude of all data was  
396 discarded, allowing usage of an arbitrarily long  
397 dataset although the absolute values of the indicators part  
398 of the dataset might substantially vary over time. The two  
399 datasets  $A$  and  $B$  differ by the extensive preprocessing that  
400 characterizes the database  $B$ . Before being fed to the  
401 component networks of the prediction model, an additional  
402 preprocessing step called complement coding was applied.  
403 This is a normalization procedure in which a given input  
404 vector  $x$  is coded along with its complement  $x^c = (1 - x)$ .  
405 As a result, the input vector and the number of input nodes  
406 double in size, allowing automatic normalization of the  
407 input vector and, more importantly, an improvement of the  
408 pattern classification capabilities of the network (Carpenter,  
409 Grossberg, & Rosen, 1991). Complement coding allows a  
410 higher-order hidden node to be associated to both the  
411 presence and the absence of a given feature.

412 Finally, an additional preprocessing stage consists in the  
413 application of Principal Component Analysis (PCA). The  
414 input vector can therefore be substituted by a vector with the  
415  $n$ th Principal Component of the original one.

416 Which dataset the network will be trained on, as well as  
417 the application of complement coding and PCA, are  
418 determined by the chromosome (further details in the  
419 Section 3).

### 420 3. The model

421 The GA employed in this study is used to estimate  
422 appropriate parameters for  $m$  mixtures of networks (MON)  
423 used to predict the closing price of a security. The  
424 chromosome used by this GA has 11 ‘genes’, which directly  
425 define the important parameters of the network, as well as  
426 the type of network and the input data type (database  $A$  or  $B$ ,  
427 Table 3 for details). Every  $i$ th MON is composed of  $j$   
428 networks to be selected from one of two types: Recurrent  
429 Backpropagation (Elman, 1991) or RBF networks (Duda  
430 et al., 2000). Elman networks are particularly suited for data  
431 when the temporal order of the data plays a crucial role  
432 (Elman, 1991), as it does in financial time series. RBF  
433 networks provide a highly compact representation for the  
434 underlying data distribution in cases where the bases  
435 accurately reflect the distribution. Each  $j$ th network is  
436 initialized by the chromosome and it is trained on the  
437 section of data defined, again, by the chromosome (Fig. 1).  
438 The chromosome defines network type, the architecture, the  
439 training set, and the most important parameters of the  
440 network as shown in Table 3.

441 Every network belonging to the  $i$ th MON is then tested  
442 on a blind data set, and single network responses are then  
443 conveyed into a voting procedure. Here a majority voting  
444 scheme chooses the prediction of the  $i$ th MON over the  
445 blind testing data, and assesses the fitness of the  $i$ th MON.  
446

Security	Operators	Equation
DIA	% ROC Close	$[(Close_t - Close_{t-1})/Close_{t-1}]100$
DIA	% Diff. Open-Close	$[(Open - Close)/Open]100$
DIA	% Diff. High-Low	$[(High - Low)/Low]100$
DIA	% Diff. Open and Mob. Avg. 10 dd	$[(Open - Mavg10 dd)/Mavg10 dd]100$
DIA	% Diff. High and Mob. Avg. 10 dd	$[(High - Mavg10 dd)/Mavg10 dd]100$
DIA	% Diff. Low and Mob. Avg. 10 dd	$[(Low - Mavg10 dd)/Mavg10 dd]100$
DIA	% Diff. Close and Mob. Avg. 10 dd	$[(Close - Mavg10 dd)/Mavg10 dd]100$
DIA	% Diff. Volume and Mob. Avg. 10 dd	$[(Volume - Mavg10 dd)/Mavg10 dd]100$
DIA	Chaikin Volatility	$H - L \text{ Average} = \text{Exponential moving average of} (High - Low)$
		$\left( \frac{H - L \text{ Average} - (H - L \text{ Average } n - \text{ periods ago})}{H - L \text{ Average } n - \text{ periods ago}} \right) 100$
DIA	MACD	$MACD_t = EMavg_1 - EMavg_2$
DIA	MOMENTUM	$Momentum = Close_t - Close_{t-n}$
DIA	RSI	$RS = (\text{Avg. price change on up days} - \text{Avg. Price change on down days})$ $RSI = 100 - (100/1 + RS)$
Nasdaq	% Diff. Open and Mob. Avg. 10 dd	$[(Open - Mavg10 dd)/Mavg10 dd]100$
S.Poor 500	% Diff. High and Mob. Avg. 10 dd	$[(High - Mavg10 dd)/Mavg10 dd]100$
Dow Jones Ind	% Diff. Low and Mob. Avg. 10 dd	$[(Low - Mavg10 dd)/Mavg10 dd]100$
Dow Jones Trasp	% Diff. Close and Mob. Avg. 10 dd	$[(Close - Mavg10 dd)/Mavg10 dd]100$
Dow Jones Util		
Dow Jones Comp		
Nikkei Bovespa		
Dax		
Ftse 100		
Dollar/yen		
Dollar/Swiss Frank		
T-Bond 30 years		
T-Bond 10 years		
T-Bond 5 years		
T-Bond 13 weeks		
Eurobond	% Diff. Open and Mob. Avg. 10 dd %	$[(Open - Mavg10 dd)/Mavg10 dd]100$
Gold	Diff. High and Mob. Avg. 10 dd	$[(High - Mavg10 dd)/Mavg10 dd] \times 100$
	% Diff. Low and Mob. Avg. 10 dd	$[(Low - Mavg10 dd)/Mavg10 dd] \times 100$
	% Diff. Close and Mob. Avg. 10 dd	$[(Close - Mavg10 dd)/Mavg10 dd] \times 100$
	% Diff. Volume and Mob. Avg. 10 dd	$[(Volume - Mavg10 dd)/Mavg10 dd] \times 100$
DJIA and T	% Diff. between Normalized DJIA and Normalized	Norm. DJIA-Norm. Tbond
Bond 30 years	T Bond	

On the left column, the securities for which each operator is calculated are listed. The second column contains the name of the operator and the third column contains the equation by which the operator is calculated. ROC, rate of change; Diff., difference; MAvg, mobile average; EMAvg, exponential mobile average. Normalization: if  $\min(x) > 0$ ,  $x_{norm} = [(x_{max} - x_{min}) \cdot (x - \min(x))] / [(x_{max} - \min(x))]$ , where  $x_{max}$  and  $x_{min}$  are the maximum and minimum value of the required mapping, respectively. If  $\min(x) \leq 0$ ,  $\min(x) = -|\max(x)|$ ,  $x_{norm} = [(x_{max} - x_{min}) \cdot (x - \min(x))] / [(x_{max} - \min(x))]$ .

The majority vote is simply given by:

$$\text{sign} \left[ \sum_{j=1}^n y_j / n \right] \quad (4)$$

where  $y_j$  is the output of a component network, and  $n$  is the number of networks in a given MON.

In this study, the fitness of a MON is given by the percentage of correct responses over the blind data set. Note that a different measure of performance, namely the net return of the system over time, could have been employed. A correct response is defined as a match between

the predicted and the actual direction of variation of the closing price of DIA in the following trading day.

It follows from this choice that the best MON in the population is not necessarily the one that has the lowest MSE on the training set, or the best  $j$ th network on the blind data set, but the one that expresses the highest number of correct voting predictions. This fitness function allows the dissociation of the performance on the training set for any single network to the actual generalization capabilities due to selection based on the collective voting measure, and on a blind data set that the component networks have not been trained on.

This choice follows the assumption that a single network cannot perform well on an extensive blind testing data set,

Table 3

The chromosome has 12 'genes' coding different parameters of the networks

Gene #	Expressing these parameters	For this network type	Param. range (min–max)
0	Network type	RBF or ELMAN	0 or 1
1	Training epochs	ELMAN	20–1000
2	Learning rate	ELMAN	0.1–0.3
3	Type of input data (Database A or B)	RBF and ELMAN	0 or 1
4	Number of training data-upper bound	RBF and ELMAN	2
5	Number of training data-lower bound	RBF and ELMAN	$n$ (where $n = \max \#$ of data)
6	Complement coding	RBF and ELMAN	0 or 1
7	# of PCA component	RBF and ELMAN	0–100
8	SSE Goal	RBF	0–10
9	Gaussian spread	RBF	0.1–10
10	Number of hidden layers-first layer	ELMAN	3–200
11	Number of hidden layers-second layer	ELMAN	3–200

The genes are differentially expressed depending on the type of network phenotype created by the chromosome [RBF or Recurrent Backprop (Elman)]. The parameter range for this simulation is shown on the right. For the genes at position 0, 3 and 6, a binary code (1 = on, 0 = off) was adopted.

but a family of networks trained on different time intervals could achieve better results. In this study a simple voting procedure was used. More sophisticated techniques could include super-ordinate networks that learn a nonlinear combination of the output of the component networks.

After fitness is calculated, a new set of MON is created using an even number of high-fitness populations of the past generation as a basis. In this simulation, the best four populations were selected for reproduction. The chromosomes specifying the networks of two of the winning populations are coupled with the corresponding chromosomes of another winning population, and crossover and mutation take place.

After crossover is performed between the  $n$  chromosomes of the component networks, the GA operator of mutation was applied (probability of mutation = 0.05).

The system was run for 100 generations, with 10 MON composed of 10 networks each (Elman and RBF). The four best MON were selected on the basis of their fitness, and were allowed to generate offspring (Figs. 2 and 3).

4. Results

In this simulation, 257 trading days of DIA are used for training and 63 trading days for testing. The final score is 73.4% correct up/down predictions over the blind data set,

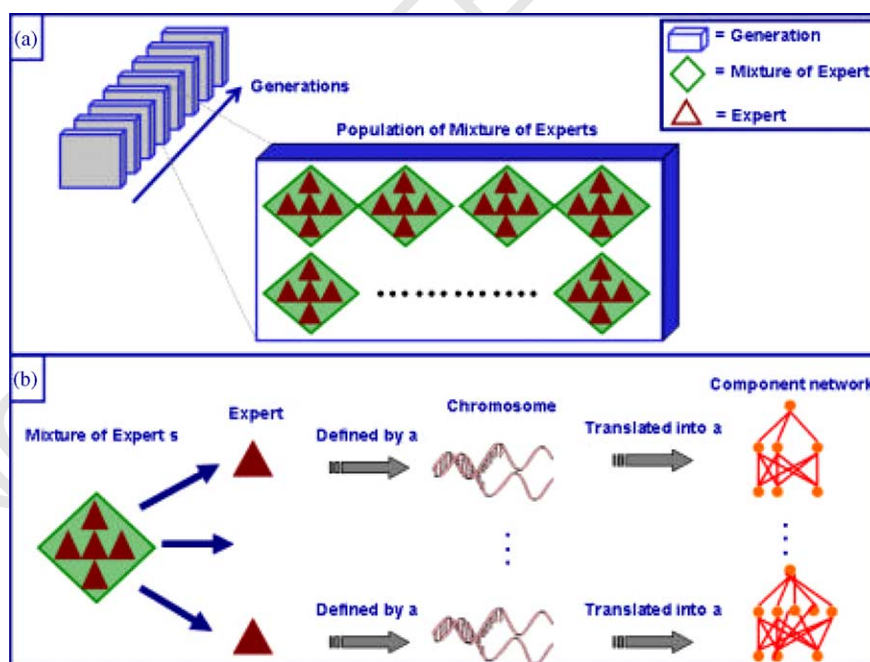


Fig. 1. (a) Every generation contains a population of MON which in turn contains  $n$  networks (Experts). (b) Each MON is composed of  $n$  networks (or experts), which are defined by a chromosome that specifies the architecture and parameters of the network.

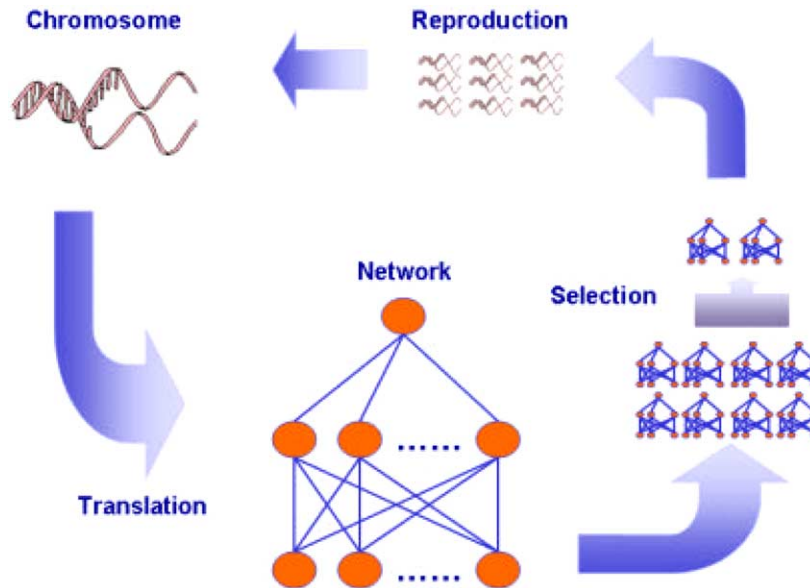


Fig. 2. The typical GA cycle. A chromosome defines the architecture of a network (in our study, a mixture of networks), which are then selected based on a measure of their fitness. The winning exemplars are then allowed to generate offspring, which is defined again through a new chromosome.

whereas the best population scored 75.2% on the 63rd generation. A plot of the fitness of the best population across epochs is shown in Fig. 4.

In order to verify that the classifiers trained with the above data and tested using the blind datasets were not deviating significantly from the overall distributions of up and down days in the test dataset, we performed two tests. First, we performed a  $\chi^2$  test on the number of actual up/down days versus the number of up/down days predicted by the classifier. The hypothesis that the distribution of up/down predictions was the same as the observed distribution of

up/down days in the test dataset could not be rejected ( $\chi^2 = 0.0353$ , critical value for confidence level of 0.05 for 1 DOF = 3.84). In order to analyze the bias of the classifier, we also performed a two-tailed pair wise  $t$ -test on the vectors signifying the up/down days (1 for up, -1 for down) for the test dataset and the classifier predictions. The hypothesis that the two vectors had different means could not be rejected either ( $p = 0.5992$ , two-tailed pair wise Student's  $t$ -test).

The above two tests strongly suggest that the classifiers trained using the methods described previously in this paper have a very low bias of prediction, if any.

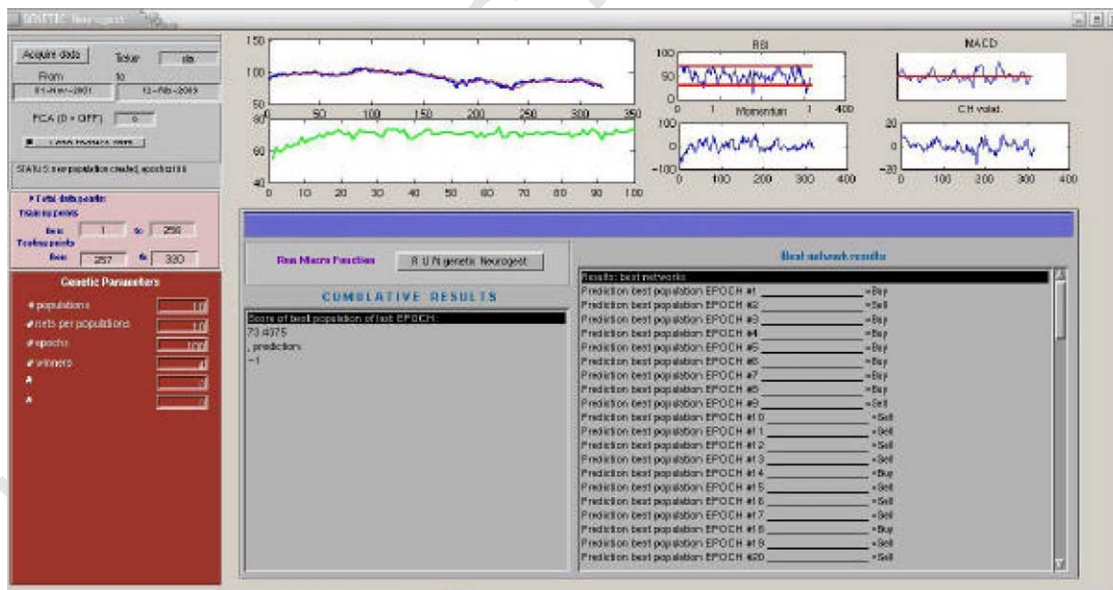


Fig. 3. The GUI realized in MATLAB controls the main parameters of the GA, namely: training and testing intervals, number of populations, number of networks per population and number of winners per epoch.

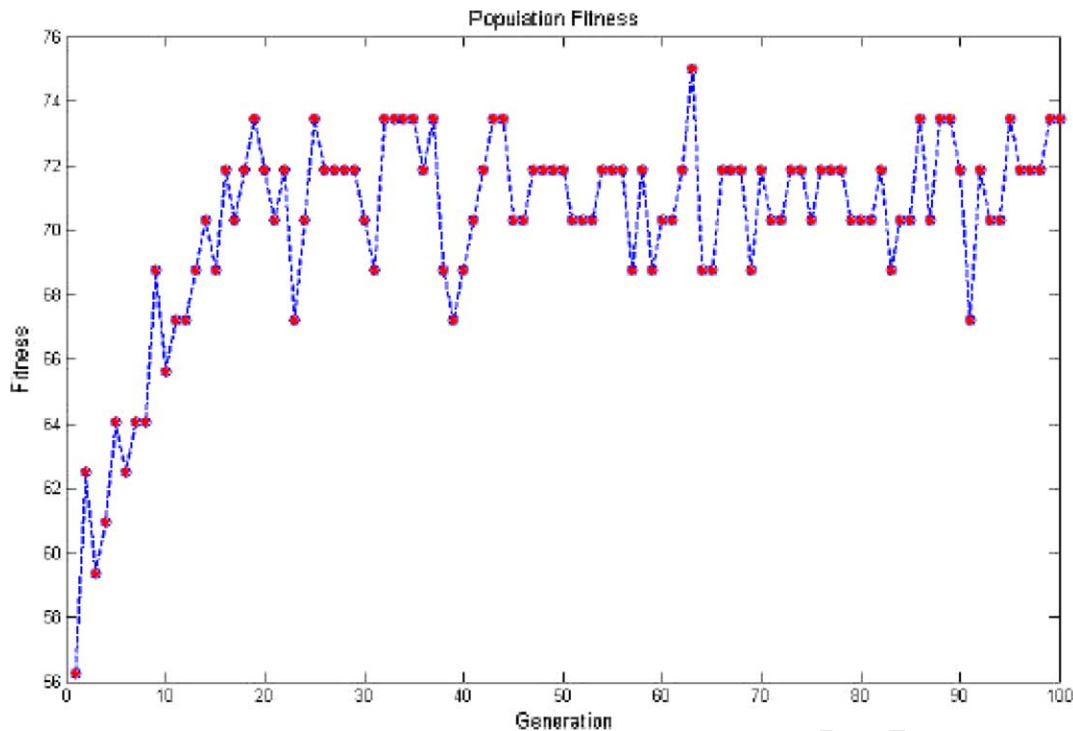


Fig. 4. The fitness of the best population across epochs shows a steep increase in the first 20 generations.

As can be seen in Fig. 4, the fitness of the best population of each epoch increases steeply in the first 20 epochs, reaches a peak on the 63rd generation, and oscillates between 66–74% in the following generations. These results are particularly good, and well above chance as shown by the  $\chi^2$  and  $t$ -test results.

## 5. Conclusions

The use of ANNs in financial applications has gained increasing popularity in the past decades. Nevertheless, a rigorous methodology on how to properly design a network or a system of networks that is able to successfully generalize from training to testing performance on a given time series is still lacking. This task is particularly difficult for large systems, especially when a combinatorial explosion of parameters is unavoidable due to the complexity and the magnitude of the system.

In this paper, we discuss a solution to the above problem employing a combination of ANNs and GAs that perform relatively well at predicting the closing price of a security. Whereas ANNs can be considered homologous to ontogenetic learning, namely the adaptation of an organism to its environment throughout its lifespan, GAs can be considered homologous to phylogenetic learning, which corresponds to the adaptive process running across generations through the selection of the best exemplars within a population. A system designed to model a complex problem such as financial forecasting should exploit the advantages and the differences of this two learning schemes. To conclude, this

contribution shows that a combination of ANNs with GAs offer promise as a good technique for forecasting stochastic time series like those seen in stock market data.

## 6. Uncited references

Gately, 1996. Kang et al., 1997.

## References

- Azoff, E. M. (1994). *Neural Network Time Series Forecasting of Financial Markets*. Chichester: John Wiley and Sons.
- Below, R. K., McInernay, J., & Schraudolph, N. (1990). *Evolving networks: using GAs with connectionist learning*. Technical Report CS90-174, Computer science and Engineering Department, University of California, San Diego.
- Carpenter, G. A., Grossberg, S., Rosen, D. B., & Fuzzy, A. R. T.: (1991). *An adaptive resonance algorithm for rapid, stable classification of analog patterns*. Proceedings of the International Joint Conference on Neural Networks (IJCNN-91), Piscataway, NJ: IEEE Service Center, II-411-416. Technical Report CAS/CNS-TR-91-006, Boston, MA: Boston University.
- Chang, E. J., & Lippmann, R. P. (1991). Using genetic algorithms to improve pattern classification performance. *Advances in Neural Information Processing*, 3, 797–903.
- Chu, C. H., & Widjaja, D. (1994). Neural network system for forecasting method selection. *Decision Support Systems*, 12, 13–24.
- Coleman, K. G., Graettinger, T. J., & Lawrence, W. F. (1991). Neural networks for bankruptcy prediction: The power to solve financial problems. *AI Review*, 5, 48–50.
- Duda, R. O., Hart, P. E., & Stork, D. E. (2000). *Pattern Classification* (2nd ed.). New York: Wiley-Interscience.

- 897 Elman, J. L. (1991). Distributed representations, simple recurrent networks,  
898 and grammatical structure. *Machine Learning*, 3, 195–225.
- 899 Engle, R. F. (1982). Autoregressive conditional to select inputs with  
900 estimates of the variance of UK inflation. *Econometrica*, 50, 987–1008.
- 901 Gately, E. (1996). *Neural Networks for Financial Forecasting*. New York:  
John Wiley.
- 902 Ginzburg, I., & Horn, D. (1994). Combined neural networks for time series  
903 analysis. *Advances in Neural Information Processing Systems Sci-*  
904 *Systems*, 6, 224–231.
- 905 Goldberg, D. E. (1989). *Genetic algorithms in Search, Optimization and*  
906 *Machine learning*. Reading, MA: Addison Wesley.
- 907 Granger, C. W. J., & Anderson, A. P. (1978). *An Introduction to Bilinear*  
908 *Time Series Models*. Gottingen: Vandenhoeck and Ruprecht.
- 909 Harp, S. A., & Samad, T. (1991). Genetic synthesis of neural network  
910 architecture. In L. David (Ed.), *In Handbook of Genetic Algorithms*.  
911 New York: Van Nostrand Reinold.
- 912 Haykin, S. (1999). *Neural networks: a comprehensive foundation. III*.  
913 Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann  
914 Arbor: University of Michigan Press.
- 915 Jordan, M., & Jacobs, R. (1994). Hsierarchical mixtures of experts and the  
916 EM algorithm. *Neural Computation*, 6(2), 181–214.
- 917 Jordan, M., & Xu, L. (1995). Convergence results for the EM approach  
918 to mixtures of experts architectures. *Neural Networks*, 8(9),  
919 1409–1431.
- 920 Kang, K., & Oh, J. (1997). Statistical mechanics of the mixture of experts.  
921 In M. Mozer (Ed.), (vol. 9) (pp. 183–189). *Advances in Neural*  
922 *Information Processing Systems*.
- 923 Kang, K., Oh, J. H., & Kwon, C. (1997). Learning by a population of  
924 perceptrons. *Physical Review E*, 55(3PtB), 3257–3261.
- 925 Kingdon, J. (1997). *Intelligent Systems and Financial Forecasting*.  
926 London: Springer Verlag.
- 927 Lapedes, A., Farber, R., (1987). Nonlinear signal processing using neural  
928 networks: prediction and system modeling. Technical Report LA-UR-  
929 87-2662, Los Alamos National Laboratory, Los Alamos, NM.
- 930 Lapedes, A., & Farber, R. (1988). How neural nets work. In D. Z. Anderson  
931 (Ed.), *Neural Information Processing Systems* (pp. 442–456). New  
932 York: American Institute of Physics.
- 933 Miller, G. F., Todd, P. M., & Hedge, S. U. (1989). Designing neural  
934 networks using genetic algorithms. In *Proceedings of the Third*  
935 *International Conference on Genetic Algorithms*, 379–384.
- 936 Montana, D. J., & Davis, L. (1989). Training feed-forward neural Networks  
937 using genetic algorithms. In *Proceedings of the International Joint*  
938 *Conference on Artificial Intelligence*, 746–767.
- 939
- 940
- 941
- 942
- 943
- 944
- 945
- 946
- 947
- 948
- 949
- 950
- 951
- 952
- Odom, M. D., & Sharda, R. (1990). *A neural network model for bankruptcy*  
953 *prediction (vol. 2)*. In: *Proceedings of the IEEE International Joint*  
954 *Conference on Neural Networks*. San Diego, CA, pp. 163–168.
- 955 Pelikan, E., de Groot, C., & Wurtz, D. (1992). Power consumption in West-  
956 Bohemia: Improved forecasts with decorrelating connectionist net-  
957 works. *Neural Network World*, 2(6), 701–712.
- 958 Refenes, A. N. (1995). *Neural Networks in the Capital Markets*. Chicester:  
959 Wiley.
- 960 Salchenkerger, L. M., Cinar, E. M., & Lash, N. A. (1992). Neural networks:  
961 A new tool for predicting thrift failures. *Decision Science*, 23(4),  
962 899–916.
- 963 Shaffer, J. D., Whitley, D., & Eshelman, L. J. (1990). On crossover as an  
964 evolutionary viable strategy. In R. K. Belew, & L. B. Booker (Eds.), *In*  
965 *Proceedings of the four International Conference on Genetic*  
966 *Algorithms* (pp. 61–68). San Mateo, CA: Morgan Kaufmann.
- 967 Tam, K. Y., & Kiang, M. Y. (1992). Managerial applications of neural  
968 networks: The case of bank failure predictions. *Management Science*,  
969 38(7), 926–947.
- 970 Trippi, R. R., & Turban, E. (1993). *Neural Networks in Finance and*  
971 *Investment: Using Artificial Intelligence to Improve Real-world*  
972 *Performance*. Chicago: Probus.
- 973 Tong, H., & Lim, K. S. (1980). Threshold autoregressive, limit cycles and  
974 cyclical data. *Journal of the Royal Statistical Society Series B*, 42(3),  
975 245–292.
- 976 Venkatesan, R., & Kumar, V. (2002). A genetic algorithms approach to  
977 growth phase forecasting of wireless subscribers. *International Journal*  
978 *of Forecasting*, 18(2002), 625–646.
- 979 Waterhouse, S.R (2002). *Classification and Regression Using Mixtures of*  
980 *Experts*. Jesus College, Cambridge, and Department of Engineering,  
981 University of Cambridge, Dissertation submitted to the University of  
982 Cambridge for the degree of Doctor of Philosophy.
- 983 Weigend, A. S., Rumelhart, D. E., & Huberman, B. A. (1991).  
984 Generalization by weight-elimination with application to forecasting.  
985 *Advances in Neural Information Processing Systems*, 3, 875–882.
- 986 Whiteley, D. (1989). The GENITOR algorithms and selection pressure:  
987 why rank-based allocation of reproductive trials is the best. In J. D.  
988 Shaffer (Ed.), *In Proceedings of the International Joint Conference on*  
989 *Genetic Algorithms*. San Mateo, CA: Morgan Kaufmann.
- 990 Wilson, R., & Sharda, R. (1994). Bankruptcy prediction using neural  
991 networks. *Decision Support Systems*, 11, 545–557.
- 992 Zhang, G., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial  
993 neural networks: The state of the art. *International Journal of*  
994 *Forecasting*, 14, 35–62.
- 995
- 996
- 997
- 998
- 999
- 1000
- 1001
- 1002
- 1003
- 1004
- 1005
- 1006
- 1007
- 1008