Training Neural Networks Beyond the Euclidean Distance, Multi-Objective Neural Networks using Evolutionary Training.

Abstract

When committed to forecasting or classification tasks Neural Networks (NNs) are typically trained with respect to Euclidean distance minimisation / likelihood maximisation. This is commonly irrespective of any other end user preferences. In a number of situations, most notably time series forecasting, users may have other objectives than solely Euclidean minimisation. Users may, for example, desire model predictions be consistent in their residual error, be accurate in their directional change predictions, or may prefer other error measures than just Euclidean distance. In this paper a methodology is devised for training Multi Layer Perceptrons (MLPs) with competing error objectives using an Evolutionary Strategy (ES). Experimental evidence is presented, using the Santa Fe competition data, which supports the effectiveness of this training methodology. Identical networks are trained with different and competing error measures. Different solutions are achieved, and these solutions are shown in general to reflect the error measure preferences as defined during training. Areas for future research and model extensions are also discussed.

Index Terms

Neural Networks, Evolutionary Strategies, Multiple Objectives, Time Series Forecasting.

I. INTRODUCTION

The use of NNs in the time series forecasting domain is now well established, there are already review papers on this matter [2], as well as methodology studies [8, 10]. The main attribute which separates NN time series modelling from traditional Econometric methods, and the reason practitioners most often site for their use, is their ability to generate non-linear relationships between time series input variables with little or no *a priori* knowledge of the form that this non-linearity might take. This is opposed to the rigid structural form of most other time series forecasting methods (e.g. linear regression models, Exponential Smoothing models, (G)ARCH, and ARIMA). Apart from this important difference, the underlying approach to time series forecasting itself has remained relatively unchanged during its progression from explicit regression modelling to the non-linear generalisation approach of NNs. Both of these approaches are fundamentally based on the concept that the most accurate forecast, if not the actual realised (target) value, is that with the smallest Euclidean distance from the actual. When measuring predictor performance however, practitioners use a whole range of different error

measures [1]. These error measures on the whole tend to reflect the preferences of potential end users of the forecast model, as opposed to just the Euclidean measure.

Recent approaches to time series forecasting using NNs have introduced limited augmentations to the traditional gradient descent algorithm in order to penalise particular misclassifications more heavily, e.g. [11]. They do not however present a methodology that allows the user to state their own forecast preferences in terms of the many error measures commonly in use, and train the NN model accordingly. This paper aims to solve this problem by developing methods for training neural networks with multiple and competing objectives, through the framework of ES training.

In this paper Neural Networks will be devised that can incorporate such objectives as error consistency and Direction Success, which will be described in section II. The model itself, and the necessary accompanying algorithms for objective weighting and training completion, will be presented in section III. Experimental results on publicly available time series data will be presented in sections IV - VI using objectives of Mean Absolute Percentage Error (MAPE) minimisation, Root Mean Squared Error (RMSE) minimisation and the maximisation of the Percentage Direction Success error measure. The paper concludes with a discussion of the salient results (section VII) and theoretical extensions to the model and future work (section VIII).

II. JUSTIFICATION

Backpropagation (BP) and related algorithms are commonly used in studies, as it is generally assumed that error minimisation / likelihood maximisation is the only goal for an NN when modelling a relationship between input and output vectors. The underlying assumption of this paper, however, is that this is not always necessarily the only objective. This is because end users of time series forecasts commonly have other aims than just Euclidean distance minimised forecasts. Examples of some of these different aims will be discussed in this section.

A popular error measure for time series forecasting, due to its unit-free nature, is MAPE [1]. If a practitioner has a preference for this measure of accuracy it may also make sense that their model is trained with it in mind. (Due to their differing calculation methods, minimising the Euclidean error of a model does not mean that the MAPE error of a model is minimised). The calculation of MAPE is show in Eqs. 1 and 2, where APE_t is the Absolute Percentage Error at time *t*.

$$APE_t = \left| \frac{\hat{y}_t - y_t}{y_t} \right| \tag{1}$$

$$MAPE = \frac{\sum_{t=1}^{p} APE_t}{p} \tag{2}$$

In Eqs. 1 and 2 $\hat{y}_t$ is the predicted dependent variable at time step t, y_t is the actual (target) value at time t and p is the number of training patterns in the data set.

The advantage of MAPE over the Euclidean error measurement of RMSE is that MAPE calculates the error as a percentage of the actual value. Therefore the error measure does not contain the bias that the Euclidean does towards errors that occur when the time series value is near an absolute extreme of its range.

In a MAPE optimised model an error of '0.5' on a target of 5 is ranked the same as an error of '5' on a target of 50 or an error of '50' on a target of 500. In a Euclidean trained model an error of '50' on a target of 500 is given an error value ten times higher than an error of '5' on a target of 50 and one-hundred times higher than an error of '0.5' on a target of 5.

Therefore a trade-off between the two measures is forced when training, with current training methods by their very nature favouring RMSE.

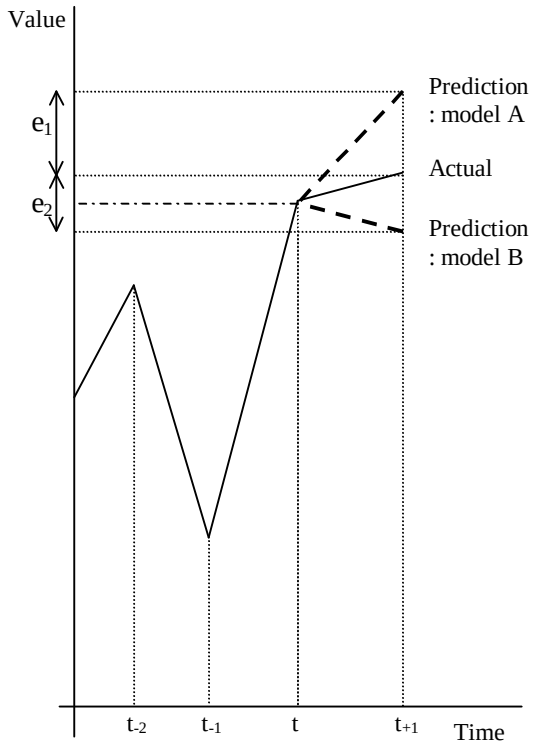

Fig. 1 Correct direction change prediction versus Euclidean minimised model.

The end user may wish to optimise learning with respect to error measures such as the correct directional change prediction [1, 3]. Consider predicting the consumer demand for a particular good. Just important as the actual level of demand predicted in the next time step, is whether the demand at the next time step is higher or lower

than the present level. The same can also be true of many physical and physiological systems. For example, in the medical field changes in direction of time series (heart rate, blood pressure) can be as important as the actual level (see Fig. 1). If it is accepted that any model forecast will contain some degree of error, the practitioner may be willing to sacrifice some of the Euclidean accuracy for a greater degree of confidence associated with the predicted directional change of the model forecast. This is achievable by calculating a composite error value, as described in Section III, and using a training process based on this composite error term.

In Fig. 1, even though the Euclidean error of model 'B' (e_2) is smaller than that of model 'A' (e_1), model 'A' correctly predicts the directional movement of the series (with respect to the realised previous time step), and as such may be the user-preferred model.

In addition, the end user might not be solely concerned with the accuracy of the forecast, but also with the properties of the residual error. They may desire the model's error be consistent (having low volatility), as opposed to one that has both periods of very low error and also occasional periods of very high forecast error. In other words, end users may often prefer a model that has overall higher average Euclidean error than another, provided it is less likely to produce instances of very large prediction error. This preference may be derived from the costs associated with large forecast errors. Take for example the situation of forecasting a manufacturing firm's inventory level. A certain degree of forecast error may be absorbed by the company, but sporadic large differences between the predicted and actual could leave the firm with a large surplus of material (and associated storage costs) - or worse, lacking the materials it needs to satisfy demand.

Given these different and competing objectives, a NN model framework shall be discussed where learning can be optimised with respect to different error preferences.

III. THE MULTI-OBJECTIVE NEURAL NETWORK MODEL

In this section the ES training model for NNs will be introduced and an extension proposed for encompassing multiple error measures. A method of error smoothing is also introduced as an attempt to solve the current stopping problem associated with ES (and GA) trained NNs.

The use of Evolutionary and Genetic approaches to Neural Network training has received increasing attention in recent years [4, 5, 6, 7, 9, 12, 14]. Indeed the ability of these approaches to facilitate NN training beyond the Euclidean objective was highlighted by Porto et al [9], but apparently taken no further. Other limited approaches to multi-objective training do appear in the literature, but in the form of simultaneously choosing the network topology as well as Euclidean training e.g. [7]

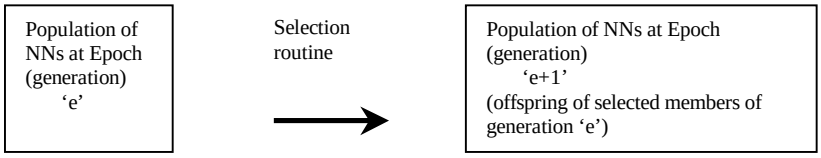

Diagram 1, generic population selection.

The ES proposed for use in this multi-objective learning model is that used in [4, 9, 12] (in the latter two it is referred to as Evolutionary Programming) and is shown in Eq. 3. Here the weight space of a network is perturbed by some values drawn from a known distribution each epoch (generation).

$$w_{n,i,j,k+1} = w_{n,i,j,k} + \gamma \cdot \Theta \tag{3}$$

where $w_{n,I,j,k}$ is the weight between the i^{th} and j^{th} nodes of the of the n^{th} network in the population at the k^{th} epoch of training, 'Θ' is a number drawn at random from a Normal (0,1) Guassian distribution and 'γ' is some fixed (or weight dependant) multiplier [4, 9, 12, 14]. Typically a given network's ability to reproduce into the next epoch's (generation's) population is determined by its relative average Euclidean error, either in relation to the training set, or the validation set. Usually the population of networks is ranked at each epoch (generation), with the fittest Networks having a higher probability of offspring in the next generation (see Diagram 1).

In this multi-objective case, a Network's fitness f_n is defined by the error metric related to the user defined objectives,

A weighted summation is used in this paper as shown in Eq 4. Here the fitness of a particular network is a linear composite of k different error terms, where the number k and type of error measures incorporated are decided by the user.

$$f_n = \alpha_1 \cdot \varepsilon_{1,n} + \alpha_2 \cdot \varepsilon_{2,n} + \ldots + \alpha_k \cdot \varepsilon_{k,n} \tag{4}$$

Before f_n is calculated, the various different errors of the n^{th} network ($\varepsilon_{1,n}$ to $\varepsilon_{k,n}$ in Eq. 4) are transformed to lie in the range 0 and 1. The coefficient weightings themselves (α_1 to α_k) are user defined, being dependent on the user's preferences for particular error measures (for instance, if they have a larger preference for error measure ε_1 in comparison to ε_2, they will set the α_1 coefficient larger than the α_2 coefficient). More complex dynamic weighting algorithms for this purpose are also discussed in section VI.

Of concern when training Neural Networks with any training algorithm are the stopping criteria. In relation to BP there are a number of methods currently in use. The more robust method incorporate three data sets, one to train on, one on whose error the network makes its decision to stop training and the final being the test set (upon

which the network's relative success is judged). A common problem with Evolutionary and Genetic approaches to training NNs is that their error curves are not as smooth as those of gradient descent algorithms (see Fig. 2) and as such the decision as to when to halt network training is more complicated. Indeed, in some published papers training is stopped at an *a posteri* selected error level, which is set after continuous runs by the user to find the optimum (e.g. [4]), or after an arbitrary set error level is reached on the training set [9]. This makes meaningful comparisons between the performance of differently trained networks difficult (given that not only the training method, but also the stopping routine may be different). A solution to this problem is proposed in this paper. By using a Finite Impulse Response Filter (also known as Single Exponential Smoothing in the economic disciplines) to transform the realised network error per epoch, a smoothed term $\tilde{f}_{n,e}$ is generated at epoch *e* (see Eq. 5).

$$\tilde{f}_{n,e} = \eta \cdot \tilde{f}_{n,e-1} + (1-\eta) \cdot f_{n,e} \tag{5}$$

The degree of smoothing is dependent on the value of η; the higher the value of η the greater the degree of smoothing (an η value of 0 would result in $\tilde{f}_{n,e}$ equalling $f_{n,e}$). Fig 3. shows an example of smoothed ES Network training and validation error which is resultant from applying Eq. 5 on the errors in Fig. 2. Figures 2 and 3 were generated by training a standard Euclidean ES NN on the Santa Fe time series A (see section IV for data definition).

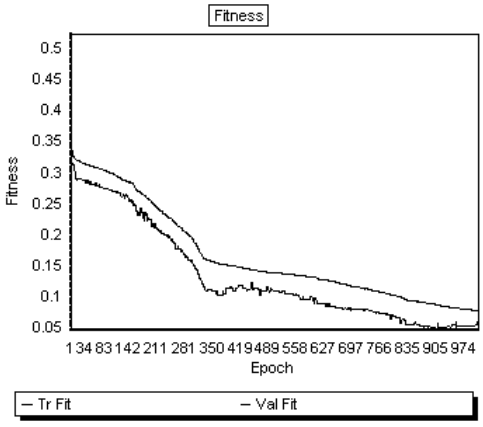

Figure 2. Standard *f*, Training and Validation.

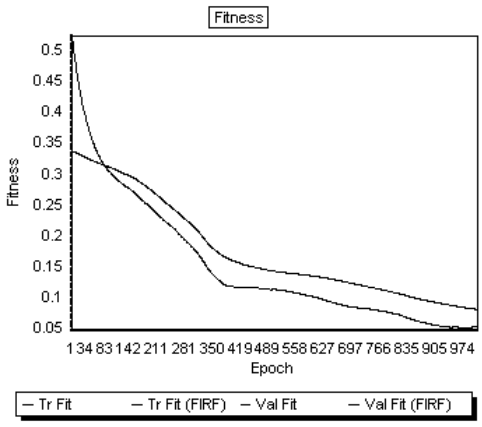

Figure 3. *f-tilda*, Training and Validation

As can be seen in Figs. 2 and 3, it is much easier to ascertain when a plateau is reached, or the when the validation error starts to rise in the transformation $\tilde{f}_n$ as opposed to f_n. This permits identical stopping regimes to be implemented on gradient descent and Evolutionary trained networks based upon the validation global minimum, as opposed to the current method of stopping ES trained networks at a fixed error level.

IV. NETWORK TOPOLOGY AND LEARNING RULES

A. Network topography

In order to test the practicality and the effect of the multi-objective NN training a simple set of experiments were devised. As, to the best of the authors' knowledge, this is the first paper to deal with this specific problem, it is intended to keep network complexity to a minimum. This is in order to reduce the possibility of exogenous factors affecting the results.

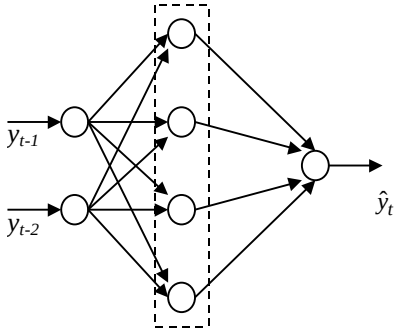

Figure 4: Network topography

The chosen network topology is identical to that used in [4] (see Fig. 4) where ES was also used in the time series forecasting domain. This consists of a single layered MLP with four hidden nodes, two input nodes and one output node. The input vector being the past two realisations of the time series, y_{t-1} and y_{t-2} where the next realisation is y_t . The functional relationship therefore being as described in Eq. 6.

$$\hat{y}_t = f(y_{t-1}, y_{t-2}) \tag{6}$$

B. Data used

Data set	Description	Observations
A	Laser generated data	1000
B1, B2, B3	Physiological data, spaced by 0.5 second intervals. The first set is the heart rate, the second is the chest volume (respiration force), and the third is the blood oxygen concentration (measured by ear oximetry).	34000 each
C	Tickwise bids for the exchange rate from Swiss francs to US dollars; recorded by a currency trading group from August 7, 1990 to April 18, 1991	30000
D1, D2	Computer generated series	50000
E	Astrophysical data. A set of measurements of the light curve (time variation of the intensity) of the variable white dwarf star PG1159-035 during March 1989, 10 second intervals.	27204

Table 1. 1991 Santa Fe Competition data

The time series data sets used in this paper are drawn from the Santa Fe competition sets of 1991, and consist of 8 diverse time series sets (see Table 1).

These time series were chosen due to the breadth and diverse nature of the time series generation processes.

C. Learning rules and stopping criteria

The core ES learning routine is that presented in Eq. 3, the population selection routine will follow the methodology of [6].

1) ES population Selection Algorithm

(a) Initiate 100 Networks with a randomised weight vectors (between 0.1 and –0.1).

(b) Perturb weights by ES.

(c) Rank by fitness.

(d) Select base networks for next generation.

(e) Go to (b) or stop if stopping criteria met.

The perturbation in (b) is as previously described in Eq. (3). In (c) the population of networks are ranked in descending order by relative fitness according to Eq. (4). In (d) the fittest 20 networks are then replicated twice and inserted into the next population, the next 60% are replicated once and inserted. The bottom 20% is discarded from the population. This is then repeated until the criteria for stopping is met. Elitism is also implemented, such that one of the two replicas of the fittest individual NN passed into the next generation does not have its weights perturbed. The perturbation is performed with a fixed γ value of 0.02, as used in [4].

In the multiple objective variant of ES Direction success (MODS) was calculated using the Eqs. 7 and 8, based on the first Sobolov Norm.

$$\text{MODS} = \sum_{t=1}^{p} \frac{1}{1+|\Delta_t|} \cdot S_t \tag{7}$$

$$\Delta_t = \Delta y_t - \hat{\Delta} y_t \tag{8}$$

Where S_t=1 if both actual and predicted changes (Δy_t and $\hat{\Delta} y_t$) are of the same sign, and S_t=-1 if actual and predicted changes are of different signs. p is as previously defined in Eq. (2). As can be seen the maximum value this can take is p and the minimum $-p$. This can be transformed to a percentage value using Eq. (9)

$$\text{Percentage MODS} = \frac{\text{MODS} + p}{2p} \cdot 100 \tag{9}$$

This compares to the standard calculation of the Percentage Direction Success error [1], as described in Eq. 10.

$$\text{DS} = \frac{\sum_{t=1}^{p} S_t}{p} \cdot 100 \tag{10}$$

Here S_t=1 if both actual and predicted changes (Δy_t and $\hat{\Delta} y_t$) are of the same sign, and S_t=0 otherwise. Eqs. 7 and 8 are used as experimentation showed that the standard formula in Eq. 9 caused stagnation in learning, if used training as the only error term. It was found that most networks trained using this equation failed to achieve significantly above 50% direction success on training, validation or test sets. The cause of this was the trained networks always predicted significantly higher (or lower) than the actual, completely outside the range of the training data. Equations 7 and 8 are used as they do not entirely divorce the concept of direction success from some distance measure. This forces the prediction to be in the general range of the time series, while the direction success drives the exact fitting.

The re-weighting of the Direction Success error of the models to lie between 0 and 1 was relatively easy, due to the known maximum and minimum levels it could reach. In the case of MAPE and RMSE there was no *a priori* knowledge with regards to the maximum errors (in theory the maximum of both is ∞ and minimum 0); therefore a different approach had to be taken to the re-weighting of the errors before insertion into Eq. 4. The average Euclidean error (RMSE) of the 100 initialised networks of the ES model was taken as a proxy for maximum RMSE error (and the average MAPE of the 100 initialised networks for the maximum MAPE calculation). These values where then used for re-weighting the errors of the networks between 0 and 1.

The η value in the Finite Impulse Response Filter was set to 0.975 and BP networks were trained with a learning rate of 0.05 and a momentum of 0.5.

V. EXPERIMENT OUTLINE

EXPERIMENT	DISCRIPTION	AIM
Experiment 1	Train five different network types (two traditional Euclidean based models, three multi-objective) on each of the Santa Fe original time series.	Compare the error properties of the fitted models with their learning specifications.
Experiment 2	Train five different network types (two traditional Euclidean based models, three multi-objective) on each of the Santa Fe first difference time series.	Compare the error properties of the fitted models with their learning specifications.
Experiment 3	Train sixty different multi-objective networks with different preference weightings on a single time series	To generate estimated perato frontiers, and highlight real trade-off choices that are made in the choice of leaning method.

Experimental outline.

Five networks were trained on each of the data sets (and their first difference transformations) according to the stopping rule as defined below, with a consecutive data partition of 80/10/10 (the first 80% used as the training set the next 10% as the validation set and the remaining 10% as the unseen test set). The first differences were included as well as the originals, as these time series should (theoretically) be more difficult to predict, as a high degree of temporal correlation is removed from the series by the transformation. It was therefore of interest to see whether multi-objective training approaches would improve performance by the same degree on both groups of time series, or whether it seemed better suited to a particular type. First difference data does however cause problems for certain error measures, specifically MAPE, due to the zero mean reverting nature of the series. As can be seen in Eqs. 1 and 2, any near zero values in y_t will lead to extremely large values of APE_t (and therefore MAPE) even with small differences in actual and predicted. The skewness RMSE experiences by design on large error values is magnified in MAPE on near zero values. Therefore the MAPE results, with respect to the difference series, should be treated with extreme caution – they are only included for completeness.

The first set of networks were trained using the standard BP algorithm, and the second using the standard Euclidean ES approach. The remaining three sets of networks were trained using the multi-objective ES methodology. The first with an equally weighted Euclidean / Direction Success (E/D) fitness function, the second with an equally weighted Euclidean / MAPE (E/M) fitness function and the third with an equally weighted MAPE / Direction Success (M/D) fitness function.

It was decided to train standard ES NNs as well as the multi-objective variants as a number of studies [4, 9, 14] reported better results when using the ES method as opposed to BP. By comparing the results with standard ES trained NNs, as well as BP trained NNs, it is hoped any improvements in performance (with respect to certain error measures) can be judged as due to the augmented learning regime, as opposed to using the ES methodology to train NNs *per say*.

After training, the fittest NN of the population (with respect to its performance on the validation set) was used on the test set.

Neural Network training was stopped when the combined fitness error on the validation and training set had fallen by less than 0.00005% of their value 5 epochs ago. The inclusion of the validation error prevents over-fitting, however by summing the two errors (as opposed to strictly using the validation error on its own), a slight trade-off is permitted between the errors while pushing through local minima [13]. Commonly validation error fluctuates while still exhibiting a downwards trend, which often causes sub-optimal early stopping when used as stopping criteria.

In addition, this Stopping was not active until 1000 Epochs had been reached. This was implemented because during the early stages of training the BP networks pushed through expansive local minima which caused both the not only the validation error to rise, but also the training error to rise temporarily, this also would have caused sub-optimal early stopping in the standard regime.

VI. RESULTS

A Experimental results

Error	BP	ES euc	ES E/D	ES E/M	ES D/M
R^2	0.914	0.916	0.873	**0.931**	0.869
rmse	**9.990**	14.32	20.36	10.15	19.49
Dir.%	68.53	70.86	**71.27**	71.13	71.12
gmrae	**1.589**	1.679	1.986	4.608	1.897
BRW	48.12	54.85	**56.24**	54.82	52.01

Table 2: Santa Fe series, average error values from Experiment 1.

Error	BP	ES euc	ES E/D	ES E/M	ES D/M
R^2	0.280	0.323	**0.331**	0.323	0.334
rmse	9.740	9.725	9.747	**9.625**	9.728
Dir.%	74.02	72.13	**76.81**	73.59	**76.81**
gmrae	1.099	**0.890**	1.072	0.895	1.113
BRW	52.87	56.13	51.41	**56.18**	51.72

Table 3: Santa Fe difference Series, average error values from Experiment 2.
Geometric Mean Relative Absolute Error values in tables 2 and 3 are $\times 10^{-3}$.

Tables 2 and 3 show the average performance of all five models over the two data sets after performing Experiments 1 and 2 as described in section V. The error measures chosen are some of those recommended by Armstrong and Collopy [1], the calculation of which can be found in section VII (appendix). Average MAPE values are not included in Tables 2 and 3 as the diverse range of these values over the data sets makes an average comparison meaningless. The similar ranges of the data however allow average values of the (non unit-free) RMSE and other errors to be meaningfully computed.

Error	Euclidean Model	ES E/D	ES E/M	ES D/M
RMSE	BP	25.0	37.5	25.0
	Euc ES	12.5	**57.1**	14.3
Dir.	BP	**100.0**	**57.1**	**57.1**
	Euc ES	**66.7**	40.0	**50.0**
MAPE	BP	37.5	**50.0**	37.5
	Euc ES	25.0	**71.4**	12.5

Table 4: Percentage of models with better error measures over Santa Fe series.

Error	Euclidean Model	ES E/ D	ES E/M	ES D/M
RMSE	BP	14.3	**50.0**	14.3
	Euc ES	14.3	33.3	14.3
Dir.	BP	**62.5**	14.3	**62.5**
	Euc ES	**57.1**	28.6	**50.0**
MAPE	BP	*37.5*	*25.0*	*25.0*
	Euc ES	***62.5***	***62.5***	***50.0***

Table 5: Percentage of models with better error measures over Santa Fe difference series.

Error	Euclidean Model	M	N
X	P	A	**e**
	Q	B	f
Y	P	**C**	**g**
	Q	**D**	**h**

Table 6: Explanatory table for the inference from tables 4 and 5, on time series set Z.

Tables 4 and 5 show the percentage of time series where the various multi-objective NNs perform better than the Euclidean models on the different error measures. (Instances of equal performance are removed before calculation), Table 6 aids their proper interpretation. Here model 'M' performed better than Euclidean model 'P' on 'a' percentage of time series from set 'Z' on error term 'X'. Likewise model 'N' performed better than Euclidean model 'Q' on '**h**' percentage of time series from set 'Z' on error term 'Y'.

Those values shaded in dark grey are those, *cetris paribus* (all things remaining equal), expected to be 50% or above, due to the model training approach, if the initial assumptions hold. (N.B. MAPE values in table 5 are in italics as no conclusions can be meaningfully made about them).

VII. DISCUSSION

A. *General results*

As initially hoped, the results from Experiment 1 and 2 show a clear trade-off between the Direction Success error measure and the Euclidean error measure (RMSE), as can be seen in Tables 2, 3, 4 and 5. The results of training with MAPE do not however seem to support the trade-off assumptions. The nature of the MAPE error term itself and results from Experiment 3 aid the formation of two possible causes for the results as found in Experiments 1 and 2. These are discussed at the end of this section (part F) and section VIII.

B. *Experiment 1, (refer to Tables 2 and 4).*

- 1) *Training using Direction success and RMSE:* When compared to the multi-objective ES E/D models, the BP trained models had a lower RMSE on over 75% of the original time series. However the multi-objective ES models in return had higher direction success than the BP models on all (100%) of the of the original time series. The comparison between the two ES models (E/D and pure Euclidean) also support these initial results, although the trade-off between Euclidean distance minimisation and direction success accuracy is less pronounced than in the BP case. Here the ES Euclidean models had the lower RMSE on over 85% of the time series, whereas the multi-objective Eur/Dir ES models had the higher direction success on two thirds (67%) of them. E/D ES trained NNs perform on average 2.75% better than BP but only 0.4% better on average than Euclidean ES on Direction Success error measure.

- *2) Training using Direction success and MAPE:* The results with respect to Direction Success of the D/M ES trained models also support the results found from the E/D models, although the results are slightly less pronounced. The D/M models perform absolutely better on 57% of the data sets when compared with BP, but only 50% when compared with Euclidean ES. This not withstanding, the average Direction Success of the D/M models (as shown in tables 2) is almost exactly the same as the average E/D models. The lower absolute performance on the Direction Success measure may be symptomatic of problems encountered when training with MAPE than Direction Success itself. This will be discussed in section VIII. As can be seen in Table 4, the D/M models performed markedly less well on this measure compared to BP and Euclidean trained ES models (performing better on only 37.5% and 12.5% of the time series respectively).

- *3) Training using RMSE and MAPE:* The ES models trained on MAPE and RMSE performed better on MAPE than the models trained on Direction Success and MAPE, the possible reasons for which will be discussed in section VIII. Average MAPE values are meaningless given the diverse range of the MAPE values over the data sets (as mentioned in section VI), however, as can be seen in Table 4, the E/M ES models had lower MAPE on the test sets of 71.4% of the time series compared to the Euclidean ES models, but only on half on the time series when compared to BP.

C. *Experiment 2, (refer to Tables 3 and 5).*

- 1) *Training using Direction success and RMSE:* When compared to the multi-objective ES E/D models, the BP and Euclidean ES trained models both had a lower RMSE on over 85% of the differenced time series. However the multi-objective ES models in return had higher direction success than the BP models on 62.5% of the of the original time series, and was better than the Euclidean ES models on 57.1% of the differenced time series. E/D ES trained NNs had, on average, 2.8% higher Direction Success compared to BP models and were 4.7% higher compared Euclidean ES models.

- *2) Training using Direction Success and MAPE:* Due to the problems in using MAPE in this time series range it is difficult to draw conclusions, however the Direction Success values are in line with those achieved using the E/D training approach.

- *3) Training using RMSE and MAPE:* These results are presented purely for completeness no inferences can be drawn due to the reasons stated in section VI.

It was hypothesised that the relatively simplistic weighting method used to calculate the weightings of the errors when training might not ensure that the resultant model was on or near its perato frontier when its training was stopped. Therefore, after compiling the results of training over the 16 data sets, the experimentation was taken one step further by the composition of example perato frontiers of the different error measures. This was achieved using the first difference of time series A (see Fig. 5) as the training set (with identical partition and training methods as described previously). The creation of these helps underline the real trade-off between different error properties that is possible using the methodology as described previously, and also highlights any sub-optimal training that the networks may experience.

D. *Experiment 3*

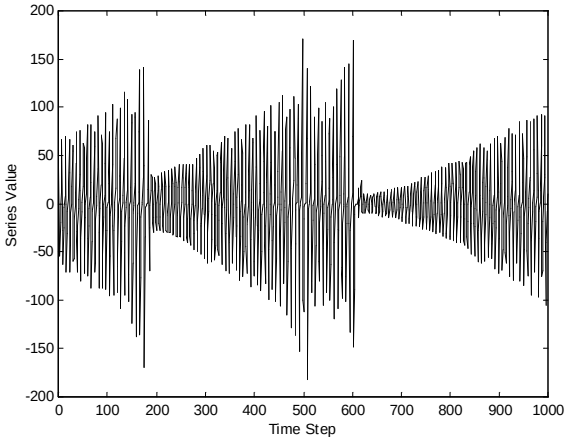

Fig. 5 Santa Fe Competition series 'A', first difference.

Fig. 6 shows the trade-off plot between Direction Success and Euclidean distance minimisation on the series 'A' difference data set in the form of a perato curve (or frontier). This was created by training 21 E/D ES models with different error weightings (and therefore user preferences). This was achieved by slowly increasing the α_1 coefficient of the direction error measure (in Eq. 4) from zero to one (by 0.1 intervals) with the α_2 coefficient of Euclidean distance fixed at one. This was followed by decreasing the α_2 coefficient of Euclidean distance to zero in the same manner with the direction error measure fixed. The plot therefore starts with a purely Euclidean error driven ES model (marked with a 'Δ') and ends with a purely Direction success trained ES model (marked with a 'o'). A '◊' highlights the equally weighted ES model.

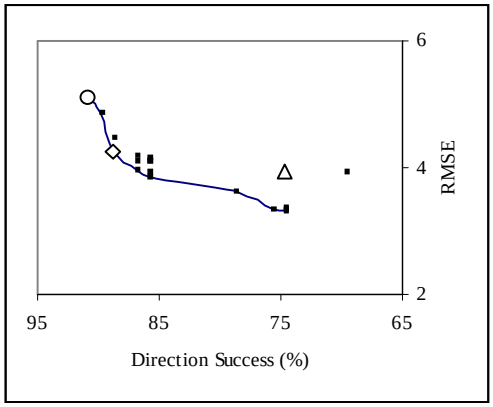

Fig 6. RMSE/Direction Success Pareto Frontier.

The exact cost of the trade-off at any point on the frontier is determined by the gradient of the frontier at that point. It is of interest to note that the extreme of Euclidean only ES training is not on the frontier, therefore highlighting the opportunity to better both error terms at once by simply including direction success metric by a small weighting in the learning regime.

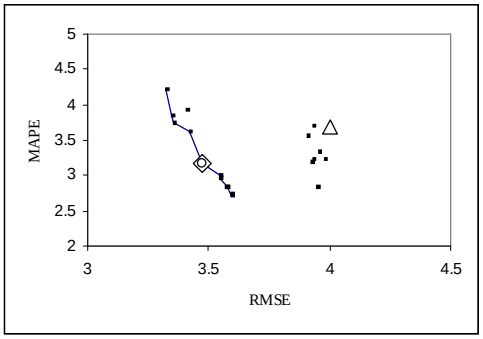

Fig 7. MAPE / RMSE Pareto Frontier.

Fig. 7 shows a trade-off plot between MAPE and Euclidean distance minimisation on the series 'A' difference data set. This was created in a similar fashion to Fig. 5. The plot again starts with a purely Euclidean error driven ES model (marked with a 'Δ') on the right of the plot, and ends with a purely MAPE trained ES model (marked with a 'o') on the left. A '◊' highlights the equally weighted ES model. Once again the purely Euclidean ES model is seen to lie off the perato frontier, with the same implication as in figure 5, and the pure MAPE model, although on the frontier, is seen not to minimise the MAPE value.

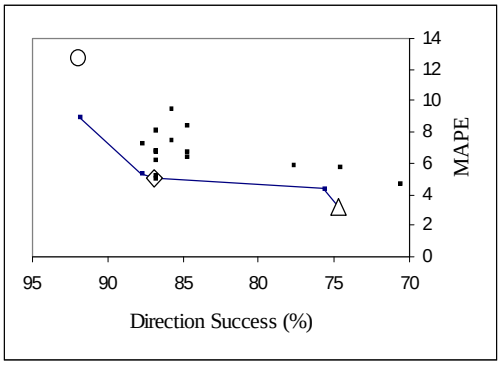

Fig 8. Direction Success / MAPE Pareto Frontier.

Fig. 8 shows a trade-off plot between Direction Success maximisation and MAPE minimisation. The plot again starts with a purely MAPE driven ES model (marked with a 'Δ') and ends with a purely Direction success trained ES model (marked with a 'o'). With '◊' highlighting the equally weighted ES model. In this case the Purely direction success model is seen to lie away from the perato frontier.

E. *Euclidean, Direction Success and MAPE*

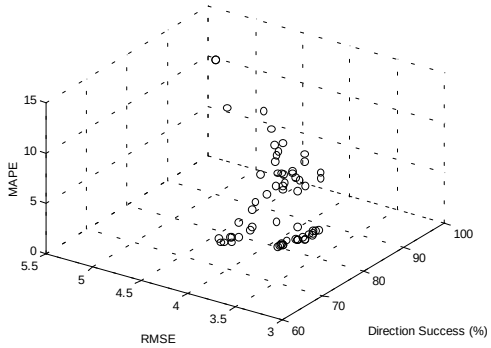

Fig. 9 3-D Error scatter plot.

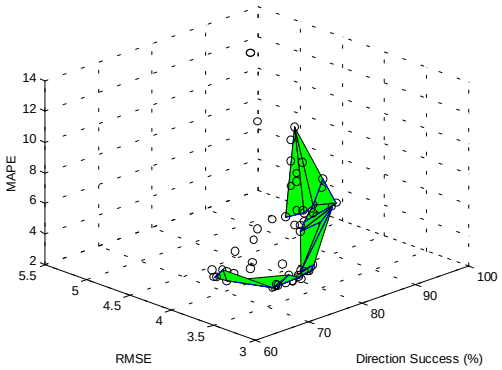

Figure 10 Implied 3-Dimensional Perato Frontier.

Figs 9 and 10. are generated using the three error measures for the NNs used in the generation of Figs. 5, 6 and 7. The three dimensional frontier in Fig. 10 is that implied by these errors. Figure 10 shows the wide range of error combinations possible whilst still on the series perato error surface (as defined by the network topology used),

F. *Variable error dominance problems with the composite error value.*

After the results of using the D/M model in Experiment 2 where received, where the results were very similar to that of the E/D model, it was realised that the error values may not be changing in a uniform manner over time, leading to different errors exhibiting dominance in the composite error value as training progressed, and skewing the final model away from the initial user weightings. Figures 11 – 16 support this conclusion, the unbroken lines on each plot showing the error value and the dashed lines showing that error as a proportion of the composite error. The plots where taken during the training of an equally weighted E/D ES network (on the 'A' difference series data set).

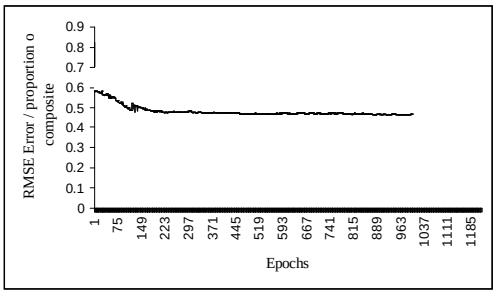

Figure 11 RMSE error each epoch in E/D ES model with relative composite weighting (scaled between 0 and 1).

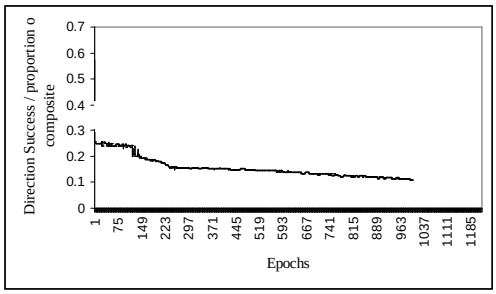

Figure 12 Direction Success each epoch in E/D ES model with relative composite weighting (scaled between 0 and 1).

As can be clearly distinguished in Figs 11 and 12, although both errors terms decrease with training, the relative contribution of these terms to the composite error value does not remain constant. Due to the fixed weighting in Eq. 4, if the network experiences a period of rapid progress in one error measure, the result is a composite error dominated by the other measure. The rapid progress itself may be due to a number of factors; initial calculation of the maximum error for adjusting the range to between 0 and 1 may be an incorrect proxy for the maximum, or the interaction between the errors may be unequal (it is plausible to have a model with 100% direction success but which does experience Euclidean error). An extension to the current model to compensate for this effect could be the application of a regular re-weighting of the errors between 0 and 1 (in effect re-calculating the maximum).

VIII. CONCLUSION

The results from experimentation support the initial premise of the paper, namely that it is possible for a user, if they so desire, to pre-determine the error attributes of their NN forecast model, which can be done through the training process described. Experiments 1 and 2 show that NNs trained with the Direction Success error measure alongside other error terms are much more likely to perform better on this error measure when compared with networks trained in the traditional fashion. Results on NNs trained with MAPE alongside other errors were inconclusive, however theoretical reasons for these results have been highlighted along with possible solutions
In addition, it has been demonstrated that within a fixed (and fairly simple) network topography, an entire range of possible error trade-off combinations is possible (as highlighted by the three-dimensional perato frontier of Fig. 10), justifying the relevance and initial assumptions of multi-objective NN training.

Furthermore the approach demonstrated is not restricted to the error measures employed in this paper; there are a large number of error terms regularly used in time series forecasting (15 mentioned in [1] alone), and any other forecast preferences a user may have is easily incorporated in the network composite error. Much further work needs to be accomplished in this area however. ES approaches to training still take significantly longer than BP training (approximate 50 times longer with a population of 100 networks). In addition, the multiple error weighting used in this paper is relatively simplistic (being a fixed summation), lending only a degree of probability to the final forecast properties, and does not ensure the resultant model will lie on or near its perato frontier (as shown in Figs 6, 7 and 8). One extension to the model has been discussed at the end of the previous section. A more radical extension to this approach would be to train a network such that it could move across the perato frontier (once it was reached). This would allow the user to actively infer the exact error properties of their end model (instead of simply stating certain preferences).

IX. APPENDIX

Root Mean Square Error

$$RMSE = \left(\frac{\sum_{t=1}^{p} (\hat{y}_t - y_t)^2}{p} \right)^{1/2}$$

Better than Random Walk (BRW)

$$BRW = \frac{\sum_{t=1}^{p} j_t}{p} \cdot 100$$

Where

$$j_t = \begin{cases} 1 & \text{if } |\hat{y}_t - y_t| < |y_{t-1} - y_t| \\ 0 & \text{otherwise.} \end{cases}$$

Geometric Mean Relative Absolute Error (GMRAE)

$$RAE_t = \frac{|\hat{y}_t - y_t|}{|y_{t-1} - y_t|}$$

$$GMRAE = \left(\prod_{t=1}^{p} RAE_t \right)^{1/p}$$

X. REFERENCES

[1] Adya, M. and Collopy, F., “How Effective are Neural Networks at Forecasting and Prediction? A Review and Evalution”, Journal of Forecasting, Vol. 17, pp481-495, 1998.

[2] Armstrong, J.S. and Collopy, F., “Error measures for generalizing about forecasting methods: Empirical comparisons”, International Journal of Forecasting, Vol. , pp69-80, 1992.

[3] Brownstone, D. “Using percentage accuracy to measure neural network predictions in Stock Market movements”, Neurocomputing, Vol. 10, No. 3, pp237-250, 1996.

[4] Greenwoood, G.W., “Training Multiple-Layer Perceptrons to Recognise Attractors”, IEEE Transactions on Evolutionary Computation, Vol. 1, No. 4, pp244-248, 1997.

[5] Janson, D.J. and Frenzel, J.F., “Application of Genetic Algorithms to the training of Higher Order Neural Networks”, Journal of Systems Engineering, Vol. 2, No. 4, 1992

[6] Janson, D.J. and Frenzel, J.F., “Training Product Unit Neural Networks with Genetic Algorithms”, IEEE Expert, October, pp26-33,1993.

[7] Merelo, J.J., Paton, M., Canas, A, Prieto, A and Moran, F., “Optimization of a competitive learning neural network by Genetic Algorithms”, Lecture Notes In Computer Scince, Vol. 686, pp185-192, 1993.

[8] Moody, J., “Forecasting the Economy with Neural Nets: A Survey of Challenges and Solutions”, Neural Networks: Tricks of the Trade,(Orr, G,B., and Mueller, K-R, eds.,) Berlin: Springer, pp347-371, 1998.

[9] Porto, V.W., Fogel, D.B. and Fogel, L.J., “Alternative Neural Network Training Methods”, IEEE Expert, June, pp16-21,1995.

[10] Refenes, A-P.N., Burgess, A.N. and Bentz, Y. “Neural Networks in Financial Engineering: A Study in Methodology”, IEEE Transactions on Neural Networks, Vol. 8, No. 6, pp1222-1267,1997.

[11] Saad, E.W., Prokhorov, D.V. and Wunsch,D.C., “Comparitive Study of Stock Trend Prediction Using Time Delay, Recurrent and Probabilistic Neural Networks”, IEEE Transactions on Neural Networks, Vol. 9, No.6, pp1456-1470, 1998.

[12] Saravanan, N. and Fogel, D.B., “Evolving Neural Control Systems”, IEEE Expert, June, pp23-27, 1995.

[13] Singh, S. and Fieldsend, J.E., "Pattern Matching and Neural Networks based Hybrid Forecasting System", Accepted Proceedings of ICAPR'2001 (Rio, Brazil), Forthcoming.

[14] Yao, X., “Evolving artificial neural networks”, Proceedings of the IEEE,Vol 89, No. 9, pp1423-1447, 1999.