Financial time series forecasts using fuzzy and long memory pattern recognition systems

Sameer Singh & Jonathan Fieldsend
PANN RESEARCH, DEPARTMENT OF COMPUTER SCIENCE, UNIVERSITY OF EXETER, EXETER, UK

ABSTRACT.
In this paper, the concept of long memory systems for forecasting is developed. The Pattern Modelling and Recognition System and Fuzzy Single Nearest Neighbour methods are introduced as local approximation tools for forecasting. Such systems are used for matching current state of the time-series with past states to make a forecast. In the past, the PMRS system has been successfully used for forecasting the Santa Fe competition data. In this paper, we forecast the FTSE 100 and 250 financial returns indices, as well as the stock returns of five FTSE 100 companies and compare the results of the two different systems, with that of Exponential Smoothing and Random Walk on seven different error measures. The results show that pattern recognition based approaches in time-series forecasting are highly accurate. Simple theoretical trading strategies are also mentioned, highlighting real applications of the system.

1. INTRODUCTION

Time-series forecasting is an important research area in several domains. Recently, neural networks and other advanced methods on prediction have been used in financial domains [1-3]. Peters [4] notes that most financial markets are not Gaussian in nature and tend to have sharper peaks and fat tails, a phenomenon well known in practice. In the face of such evidence, a number of traditional methods based on Gaussian normality assumption have limitations making accurate forecasts.

One of the key observations explained by Peters [4] is the fact that most financial markets have a very long memory; what happens today affects the future forever. In other words, current data is correlated with all past data to varying degrees. This long memory component of the market can not be adequately explained by systems that work with short-memory parameters. Short-memory systems are characterised by using the use of last *i* series values for making the forecast in univariate analysis. For example most statistical methods and neural networks are given last *i* observations for predicting the actual at time *i*+1. Long memory systems on the other hand are characterised by their ability to remember events in the long history of time-series data and their ability to make decisions on the basis of such memories

The paper is organised as follows. We first discuss the concept of local approximation in forecasting and detail its implementation in the PMRS and SNN algorithms. The paper then discusses the financial index data used. The results section compares the results obtained using the PMRS and SNN algorithms with that of the Exponential Smoothing and Random Walk methods which are widely used in the financial industry. The results are compared on a total of seven different error measures.

2. METHODOLOGY

PMRS

If we choose to represent a time-series as $\mathbf{y} = \{y_1, y_2, \ldots y_n\}$, then the current state of size one of the time-series is represented by its current value y_n. One simple method of prediction can be based on identifying the closest neighbour of y_n in the past data, say y_j, and predicting y_{n+1} on the basis of y_{j+1}. Calculating an average prediction based on more than one nearest neighbour can modify this approach. The definition of the current state of a time-series can be extended to include more than one value, e.g. the current state s_c of size two may be defined as $\{y_{n-1}, y_n\}$. For such a current state, the prediction will depend on the past state s_p $\{y_{j-1}, y_j\}$ and next series value y^+_p given by y_{j+1}, provided that we establish that the state $\{y_{j-1}, y_j\}$ is the nearest neighbour of the state $\{y_{n-1}, y_n\}$ using some similarity measurement. In this paper, we also refer to *states* as *patterns*. In theory, we can have a current state of any size but in practice only matching current states of optimal size to past states of the same size yields accurate forecasts since too small or too large neighbourhoods do not generalise well. The optimal state size must be determined experimentally on the basis of achieving minimal errors on standard measures through an iterative procedure.

We can formalise the prediction procedure as follows:

$$\ddot{y} = \phi(s_c, s_p, y^+_p, k, c)$$

where $\ddot{y}$ is the prediction for the next time step, s_c is the current state, s_p is the nearest past state, y^+_p is the series value following past state s_p, k is the state size and *c* is the matching constraint. Here $\ddot{y}$ is a real value, s_c or s_p can be represented as a set of real values, k is a constant representing the number of values in each state, i.e. size of the set, and *c* is a constraint which is user defined for the matching process. We define *c* as the condition of matching operation that series direction change for each member in s_c and s_p is the same.

In order to illustrate the matching process for series prediction further, consider the time series as a vector $\mathbf{y} = \{y_1, y_2, \ldots y_n\}$ where n is the total number of points in the series. Often, we also represent such a series as a function of time, e.g. $y_n = y_t$, $y_{n-1} = y_{t-1}$, and so on. A segment in the series is defined as a difference vector $\boldsymbol{\delta} = (\delta_1, \delta_2, \ldots \delta_{n-1})$ where $\delta_i = y_{i+1} - y_i$, $\forall i$, $1 \le i \le n-1$. A pattern contains one or more segments and it can be visualised as a string of segments $\rho = (\delta_i, \delta_{i+1}, \ldots \delta_h)$ for given values of i and h, $1 \le i,h \le n-1$, provided that $h > i$. In order to define any pattern mathematically, we choose to tag the time series y with a vector of change in direction. For this purpose, a value y_i is tagged with a 0 if $y_{i+1} < y_i$, and as a 1 if $y_{i+1} \ge y_i$. Formally, a pattern in the time-series is represented as $\rho = (b_i, b_{i+1}, \ldots b_h)$ where b is a binary value.

The complete time-series is tagged as $(b_1, \ldots b_{n-1})$. For a total of k segments in a pattern, it is tagged with a string of k *b* values. For a pattern of size k, the total number of binary patterns (shapes) possible is 2^k. The technique of matching structural primitives is based on the premise that the past repeats itself. Farmer and Sidorowich [5] state that the dynamic behaviour of time-series can be efficiently predicted by using local approximation. For this purpose, a map between current states and the nearest neighbour past states can be generated for forecasting.

Pattern matching in the context of time-series forecasting refers to the process of matching current state of the time series with its past states. Consider the tagged time series $(b_1, b_i, \ldots b_{n-1})$. Suppose that we are at time *n* (y_n) trying to predict y_{n+1}. A pattern of size k is first formulated from the last k tag values in the series, $\rho' = (b_{n-k}, \ldots b_{n-1})$. The size k of the structural primitive (pattern) used for matching has a direct effect on the prediction accuracy. Thus the pattern size k must be optimised for obtaining the best results. For this k is increased in every trial by one unit till it reaches a predefined maximum allowed for the experiment and the error measures are noted; the value of k that gives the least error is finally selected. The aim of a pattern matching algorithm is to find the closest match of ρ' in the historical data (estimation period) and use this for predicting y_{n+1}. The magnitude and direction of prediction depend on the match found. The success in correctly predicting series depends directly on the pattern matching algorithm.

The first step is to select a state/pattern of minimal size (k=2). A nearest neighbour of this pattern is determined from historical data on the basis of smallest offset ∇. There are two cases for prediction: either we predict high or we predict low. The prediction $\ddot{y}_{n+1}$ is scaled on the basis of the similarity of the match found. We use a number of widely applied error measures for estimating the accuracy of the forecast and selecting optimal k size for minimal error. The forecasting process is repeated with a given test data for states/patterns of size greater than two and a model with smallest k giving minimal error is selected. In our experiments k is iterated between $2 \le k \le 5$.

SNN

The SNN model is based on the premise that accurate forecasts can be made by finding the best match of a current actual with historical data. For finding the best match, a fuzzy proximity measures for membership computation can be used. The purpose of SNN is to predict data in the test period on the basis of known information about estimation period. The fuzzy proximity measure finds the best match of a given data point in historical data. For a given test value y_n, the aim is to generate the prediction y_{n+1}. The fuzzy measurement computes the proximity of y_n with all data in the estimation period $(y_1 \ldots y_{n-1})$ using the following measure:

$$\mu_j(y_n) = [1 + \{d(y_n, y_j)/F_d\}^{F_e}]^{-1.0} \quad \ldots(1)$$

$d(y_n, y_j)$ is the distance between y_n and y_j.

The neighbour J=j is the nearest if:

$$\mu_J(y_n) = \max(\mu_j(y_n)) \quad \ldots (2)$$

The prediction is:

$$\ddot{y}_{n+1} = y_{J+1} \quad \ldots (3)$$

In this manner, the nearest neighbour of y_n is y_j for which it has the highest membership value based on the fuzzy distance measure plus the same direction tag for previous values.

2. FINANCIAL DATA

The financial data used in this study are end of day data for the British FTSE 100 and FTSE 250 indices as well the stock prices of British Aerospace, Hong-Kong Shanghai Banking Corporation (HSBC), British Petroleum, Glaxo-Wellcome and Allied Domecq. The FTSE 100 data spans over 16 years from May 1982 to October 1999 and the FTSE 250 data over 5 years from February 1994 to October 1999. The stock data span on average 9.5 years from between May and July 1992 to October 1999. All data was obtained through LIFFE data services.

In all cases the models used were initially trained on 80% of the available data (converted to returns), the remaining 20% was used to test the forecasting ability of each of the models, with the models making one day predictions of the end of day returns.

4. RESULTS

The first error measure we shall focus on is the ability to correctly predict the direction of series change; positive or negative relative to the current position. The other measures include Root Mean Square Error, Geometric Root Mean Square Error, Geometric Mean Relative Absolute Error, Mean Square Error, Mean Absolute Percentage Error and Percentage Better than the Random Walk (Armstrong and Collopy [5]). The results with respect to the different financial time series are shown in Table (4.1).

For all time series, the fuzzy SNN model is able to predict the correct direction of movement seven out of ten times (the average accuracy being just under 71%). The PMRS algorithm also performed well, with the best model for each time series having an average accuracy of 69%. This supports the stated assumption of the models that the match between the current market situation and the historical market situation is a crucial link for making accurate forecasts.

The fuzzy SNN model performs better than the PMRS model on six of the seven time series, which in turn performs better than either the Exponential Smoothing or Geometric Walk methods.

The information as to whether the return series will be higher or lower the next day is important for the trading buy-sell-hold strategy.

If the return value is predicted as being higher tomorrow, and today's return is positive then it would be best to hold onto your stocks, or buy into that position if it is not presently held. If however the returns value is predicted to fall tomorrow, and today's return was also negative, then it may be useful to sell those stocks (if they have not already be sold).

A more ambiguous situation arises when the predicted directional movement of the return series is positive when the current returns are negative, or negative when the current returns are positive. Here accuracy with respect to the actual value, not just directional movement is needed, as a rise in returns from a negative return state may result in a positive return, or may just infer a lower negative return the next day. Likewise the actual size of a predicted returns fall from a positive returns state will determine whether a smaller positive return is realised, or a negative one.

The SNN model has the lowest Mean Absolute Percentage Error (MAPE) for all time series. It also has the lowest Mean Square Error (MSE), arithmetic Root Mean Square Error (RMSE) and Geometric Root Mean Square Error (GRMSE) of the four predictors for all the time series except the FTSE 250 (where Exponential Smoothing has the lowest). The SNN model has mixed results when compared to the other models and the random walk model. This is shown by the GMRAE measure (which should be minimised) where the SNN model consistently outperforms the other predictors, however when using the Percentage Better than Random Walk value it does not compare as favourably, with Exponential Smoothing performing the best.

The PMRS model performs slightly less well in comparison to the ES and RW models, having lower RMSE, MAPE and MSE values than the other two models for five of the seven time series, lower GRMSE for six and lower GMRAE for all seven. It's Percentage Better than Random Walk was worse than both Exponential Smoothing and SNN (except for the FTSE 250 data), However it's direction success value was uniformly higher than both Exponential Smoothing and Random Walk.

4. CONCLUSIONS

In this paper we have shown that a pattern recognition algorithm for forecasting is an alternative solution to using statistical methods, that have limitations in non-linear domains. The SNN model's performance was found to be better than that of the PMRS model and the statistical models; it is expected that further research on both these predictors will yield even better results in the future.

We also expect that SNN and PMRS technology will be further developed and fine-tuned for practical trials.

REFERENCES

[1] M. E. Azoff, *Neural Network Time Series Forecasting of Financial Markets*, John Wiley, 1994.

[2] E. Gately, *Neural Networks for Financial Forecasting*, John Wiley, 1996.

[3] A. N. Refenes, N. Burgess and Y. Bentz, "Neural networks in financial engineering: A study in methodology," *IEEE Transactions on Neural Networks*, vol. 8, no. 6, pp. 1222-1267, 1997.

[4] E. Peters, *Fractal Market Hypothesis: Applying Chaos Theory to Investment and Economics*, John Wiley, 1994.

[5] S. Armstrong and F. Collopy, "Error measures for generalizing about forecasting methods: Empirical comparisons", *International Journal of Forecasting*, vol. 8 pp 69-80, 1992

[6] S. Singh, "A long memory pattern modelling and recognition system for financial time-series forecasting", *Pattern Analysis and Applications*, vol. 2, no. 3 pp 264-273, 1999.

Table (4.1) Error measures of the four prediction models.

SERIES	Predictor	RMSE	GRMSE	GMRAE	MSE	MAPE	Better	Direction

							than RW	
Ftse100	SNN	**2.67**	**1.05**	**0.0010**	**5563**	**175**	59.90	**73.13**
	PMRS k=2	3.03	1.21	0.0011	7147	287	53.73	68.77
***	PMRS k=3	2.99	1.16	0.0011	6964	278	55.01	69.15
	PMRS k=4	3.03	1.22	0.0011	7143	285	**64.26**	68.38
	PMRS k=5	3.19	1.19	0.0011	7912	309	53.72	67.48
	ES	3.30	1.31	0.0012	8481	370	**64.26**	65.81
	RW	3.45	1.37	0.0013	9246	403	0	48.45
Ftse250	SNN	2.92	2.07	**0.0032**	2292	**183**	47.56	62.92
***	PMRS k=2	3.14	2.11	0.0036	2640	305	49.44	**67.79**
	PMRS k=3	2.96	**2.01**	0.0036	2335	280	46.06	60.30
	PMRS k=4	3.07	2.11	0.0036	2521	349	44.19	58.80
	PMRS k=5	3.33	2.23	0.0038	2962	335	43.07	61.04
	ES	**2.81**	**2.01**	0.0036	2115	303	**62.55**	64.04
	RW	2.92	2.08	0.0037	**2271**	331	0	49.44
AER	SNN	**4.12**	**3.55**	**0.0023**	**6337**	**142**	55.62	**71.92**
	PMRS k=2	4.13	3.57	0.0027	6366	217	47.86	64.97
	PMRS k=3	4.14	3.56	0.0026	6418	202	47.32	62.83
	PMRS k=4	5.80	4.21	0.0027	12620	247	48.93	65.78
***	PMRS k=5	5.80	4.19	0.0025	12586	235	49.73	68.72
	ES	5.58	4.78	0.0026	11642	218	**61.23**	62.83
	RW	5.85	5.01	0.0027	12817	231	0	51.60
HSBC	SNN	**5.02**	**1.75**	**0.0022**	**9249**	**177**	54.37	**68.85**
***	PMRS k=2	5.09	1.97	0.0024	9481	274	50.54	67.48
	PMRS k=3	5.07	2.08	0.0028	9417	337	48.63	64.75
	PMRS k=4	5.18	2.05	0.0026	9828	327	49.73	62.84
	PMRS k=5	5.22	2.11	0.0025	9966	306	52.18	66.12
	ES	6.77	2.18	0.0027	16783	662	**59.83**	60.92
	RW	7.10	2.30	0.0027	18457	726	0	52.73
BP	SNN	**1.05**	**0.56**	**0.0020**	**425**	**148**	58.53	**72.97**
***	PMRS k=2	1.16	0.68	0.0024	515	227	55.38	68.77
	PMRS k=3	1.20	0.65	0.0024	552	225	53.28	66.40
	PMRS k=4	1.25	0.66	0.0025	597	238	52.49	65.35
	PMRS k=5	1.23	0.63	0.0024	584	238	53.28	65.09
	ES	1.28	0.69	0.0025	630	259	**63.51**	66.67
	RW	1.34	0.72	0.0026	688	276	0	51.44
GXO	SNN	**2.16**	**1.22**	**0.0020**	**1784**	**179**	57.63	**71.84**
***	PMRS k=2	2.26	1.24	0.0022	1942	274	53.68	68.95
	PMRS k=3	2.70	1.45	0.0024	2768	261	52.37	68.68
	PMRS k=4	5.18	2.05	0.0026	9828	326	49.72	62.84
	PMRS k=5	2.76	1.50	0.0024	2902	269	51.05	66.05
	ES	6.77	2.18	0.0027	16783	662	**59.83**	60.93
	RW	7.10	2.30	0.0027	18457	726	0	52.73
ALD	SNN	**0.61**	**0.31**	**0.0021**	**139**	**121**	59.31	**74.80**
	PMRS k=2	0.71	0.39	0.0026	190	212	51.18	67.19
***	PMRS k=3	0.65	0.36	0.0024	158	196	52.49	70.87
	PMRS k=4	0.70	0.36	0.0026	188	198	49.34	66.67
	PMRS k=5	0.69	0.37	0.0025	181	200	47.50	63.78
	ES	0.72	0.38	0.0025	199	206	**62.73**	64.30
	RW	0.75	0.39	0.0026	215	220	0	50.39

Numbers in bold denote the best results for each error statistic, '***' denotes the chosen PMRS model for each time series