FORECASTING CRYPTOCURRENCY RETURNS USING TIME SERIES NEURAL NETWORKS MODELS

R. COCCARDA[1], L. GIOLLI[2] AND E. SARTORELLI[3]

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Keywords: Cryptocurrency, Bitcoin, Forecasting, Neural Networks, NAR Model, NARX Model

JEL Codes: Mathematical and Quantitative Methods (C2, C3, C45, C53)

1 *e-Campus University and University of Foggia. Italian Society of Statistics (SIS) and Italian Econometric Association (SIdE). Scientific Director of DidatticaInterattiva (www.didatticainterattiva.it).*
2 *e-Campus University, Roma Tre University and European School of Economics. Italian Economic Association (SIE) and Italian Econometric Association (SIdE).*
3 *Italian Technical Analysis Association (SIAT).*

Abstract

Bitcoin is the most well known and prominent cryptocurrency dependent on the decentralization and cryptography. The decentralization implies that the Bitcoin system is entirely controlled and possessed by its clients, who must hold to a similar arrangement of principles. The cryptography controls money creation (fixed to a limit of 21 million coins) and exchanges, no national bank is required. This decentralized nature offers numerous focal points, for example, being free from government control and guideline, however critics say that there is no control on the entire system apart from its users. Predicting the future evolution of Bitcoin returns could be a central concern in finance. Forecasts are typically produced from simple linear time series models. The classical methods used for time series prediction, like ARIMA models or structural time series models, assume that there is a linear relationship between inputs and outputs. Artificial neural network modeling has recently attracted much attention as a new technique for estimation and forecasting in economics and finance. The chief advantages of this approach are that such models can usually find a solution for very complex problems, and that they are free from the assumption of linearity that is often adopted to make the traditional methods tractable. Neural networks have been successfully used for forecasting of financial and economics data series. The objective of this article is to forecast Bitcoin returns using neural networks, and to compare the forecasting performance of such non-linear models. We found that best time series neural network models roughly exhibit the same MSE.

1 INTRODUCTION

The theory of efficient markets has been largely denied by the statistical study of time series of the financial markets. The financial markets do not have completely unpredictable trends, but there are some forms of order that have certain recurrences, that allow to obtain results in forecasting techniques. Mandelbrot called these non-regular order forms with the name of Noah effect. Above all technical analysis has developed over time from a purely graphic discipline to a decidedly more quantitative one. The classical technical analysis tries to weigh and predict the sentiment of the markets, or the continuation or not of a trend. Quantitative analysis tries to identify the most probable future scenario in respect of a series of calculation parameters. There are many forecasting techniques related to the use of indicators and oscillators that combined within the trading system can give excellent results with a good regularity. In the present work we considered two categories of time series neural network models, the NAR model and the NARX model, and we compared the goodness of fit and forecasts of the competing models on the basis of MSE (mean squared error) and R (correlation coefficient between outputs and targets). There are five input series for NARX model, the lagged Bitcoin returns, and four of the most popular oscillators in the literature: Moving Average Convergence Divergence (MACD), Relative Strength Index (RSI), Rate of Change (ROC) and Acceleration (Momentum of ROC). Those four oscillators, together with lagged Bitcoin returns, are used, in the NARX model, to improve the forecasting ability of the methodology while NAR model relies only on lagged Bitcoin returns.

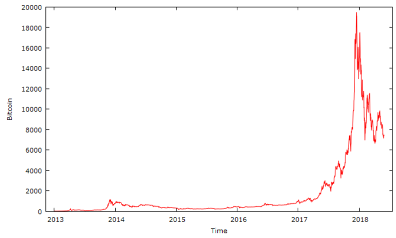
2 SURVEY OF THE EMPIRICAL LITERATURE

The use of neural networks was introduced in economics at the beginning of the nineties. There has been considerable interest in applications of neural networks in the economics literature, particularly in the areas of financial statistics and exchange rates. In contrast, relatively few studies have applied neural network methods to cryptocurrencies, such as Bitcoin. The article by [39] is likely the definitive introduction of neural networks to the econometrics literature, the authors draw many of the parallels between econometrics and neural networks. [39]'s theoretical contribution has been followed with some applied work by [44]. These authors demonstrate that the 14 macroeconomic series analyzed can be nicely modelled using neural networks. [55] represents another major attempt at using neural nets to forecast macroeconomic variables. [8] studies the behaviour of cryptocurrencies financial time series of which Bitcoin is the most prominent example. The dynamic of those series is quite complex displaying extreme observations, asymmetries, and several nonlinear characteristics which are difficult to model. [8] develops a new dynamic model able to account for long memory and asymmetries in the volatility process as well as for the presence of timevarying skewness and kurtosis. [10] focuses on predicting the conditional volatility of the four most traded cryptocurrencies: Bitcoin, Ethereum, Litecoin and Ripple. [10] investigates the effect of accounting for long memory in the volatility process as well as its asymmetric reaction to past values of the series to predict: one day, one and two weeks volatility levels. [59] adopts artificial neural network (ANN) and two varieties of time series neural network models to forecast the stock index of Chinese market. [13] provided a statistical analysis of the log-returns of the exchange rate of Bitcoin versus the United States Dollar. Fifteen of the most popular parametric distributions in finance were fitted to the log returns. They found that generalized hyperbolic distribution gave the best fit. Predictions were given for future values of the exchange rate. [37] forecasted daily returns of cryptocurrencies using a wide variety of different econometric models. To capture salient features commonly observed in financial time series like rapid changes in the conditional variance, non-normality of the measurement errors and sharply increasing trends, [37] developed a time-varying parameter VAR with t-distributed measurement errors and stochastic volatility. [37] used around one year of daily data, to perform real-time forecasting exercise and investigated whether any of the proposed models is able to outperform the naive random walk benchmark.

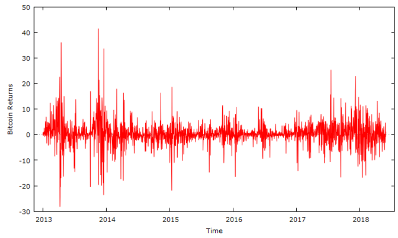
3 DATA

The working time series are daily Bitcoin closing prices, daily Bitocoin returns and four daily oscillators. There are 6 series in the data set and the sample is from January 1, 2013 until May 31, 2018. There are 1977 observations in each series, so the sample is quite large, neural networks need larger samples in order to be estimated properly, this is due to the large number of parameters introduced in such models. In Table 1 are reported the descriptive statistics for the six time series. All the series are right skewed and leptokurtic, so they have fat tails and high peak (which is a feature of most financial time series), except for RSI(14) that is platykurtic. In Table 2 are shown the test statistics (and p-values) for normality, autocorrelation and randomness regarding the mentioned six time series. All the series are non normal, autocorrelated and non random, except for Bitcoin returns that exhibits no autocorrelation and randomness. In Figure 1 are shown the time series plots for Bitcoin daily closing prices and Bitcoin daily returns. The plot for Bitcoin returns exhibits many spikes which means that the distribution has fat tails. In Figure 2 there are the time series plots for the four oscillators computed using the Bitcoin time series.

For our forecasting we used four daily oscillators among the best known in the literature: MACD, RSI, ROC and Momentum of ROC. Let's see some details of these oscillators. The Moving Average Convergence Divergence/Divergence (MACD) was designed by Gerald Appel and is the difference between 2 exponential moving averages: a short moving average at 12 periods and a longer at 26 periods. The Relative Strength Index (RSI) was designed by Wells Wilder and substantially compares the pressure of buyers compared to that of sellers and is calculated over 14 periods. The Rate of Change (ROC) is calculated over 10 periods and is the percentage difference between the last data and 10 previous periods. Practically is a velocity of price variation. The Acceleration is the Momentum of the ROC, or the speed of variation of the ROC and is calculated over 10 periods. We know that the objective of the oscillators is to show the most cyclical part of the markets, reducing the effects of the trend. Each of the selected oscillators performs this work differently and on slightly different time horizons.

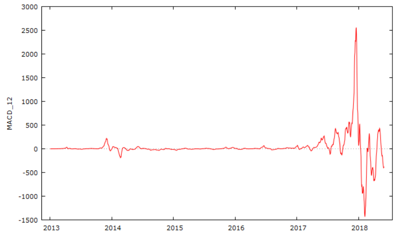


(a) Bitcoin closing prices.

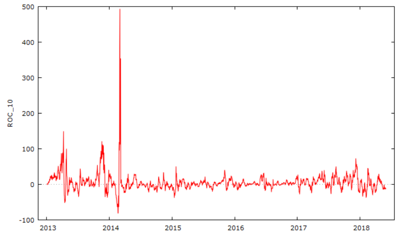


(b) Bitcoin returns.

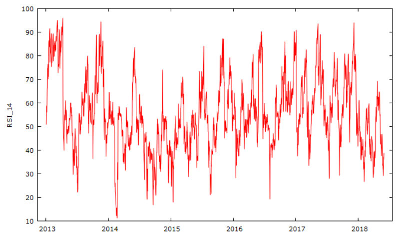
Figure 1: Time series plot of Bitcoin closing prices and Bitcoin returns from January 1, 2013 until May 31, 2018.



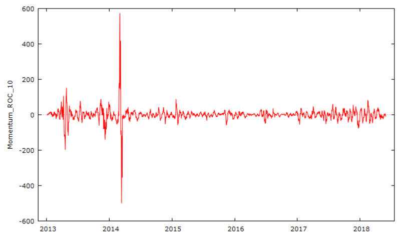
(a) MACD (12).



(b) ROC (10).



(c) RSI (14).



(d) Momentum (10) of ROC.

Figure 2: Time series plot of MACD (12), ROC (10), RSI (12) and Momentum (10) of ROC for Bitcoin from January 1, 2013 until May 31, 2018.

4 NEURAL NETWORKS FOR TIME SERIES FORECASTING

We considered two types of neural network models for time series forecasting, the NARX model and the NAR model (see [6] and [5]). In the first type of neural network model, you can predict future values of a time series $y(t)$ from past values of that time series and past values of a second time series $x(t)$. You can add also same more series $x(t)$. This form of prediction is called nonlinear autoregressive with exogenous (external) input, or NARX (see Figure 3), and can be written as follows:

$$y(t) = f(y(t-1), \ldots, y(t-d), x(t-1), \ldots, (t-d)) \tag{1}$$

Where d is the number of delays. This model could be used to predict future values of a stock or bond, based on such economic variables as unemployment rates, GDP, etc. In the second type of neural network model, there is only one series involved. The future values of a time series $y(t)$ are predicted only from past values of that series. This form of prediction is

Table 1: Descriptive statistics for Bitcoin closing prices, Bitcoin returns and oscillators from January 1, 2013 until May 31, 2018.

Variable	Mean	Median	Standard Deviation	Skewness	Kurtosis	IQR
Bitcoin	1761.400	475.320	3233.600	2.715	7.233	733.130
Bitcoin returns	0.435	0.247	4.786	0.468	7.767	3.406
MACD(12)	29.411	3.423	278.620	3.304	33.600	27.911
ROC(10)	5.495	2.070	26.488	7.114	101.620	16.718
RSI(14)	55.507	53.730	15.288	0.280	-0.205	19.994
Momentum(10) of ROC	-0.054	0.585	37.429	1.523	78.700	20.417

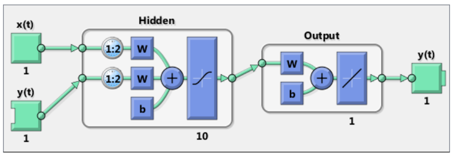


Figure 3: The NARX network with 1 input series, 2 number of delays and 10 hidden neurons (source http://www.mathworks.com).

called nonlinear autoregressive, or NAR (see Figure 4), and can be written as follows:

$$y(t) = f(y(t-1), ..., y(t-d)) \tag{2}$$

Where d is the number of delays. This model could also be used to predict financial instruments, but without the use of one or more companion series.

To train the NARX and NAR neural network models we divided each time series into three sets as follows:

- The first 60% of the data has been used for training, these are presented to the network during training, and the network is adjusted according to its error.
- From 60% to 80% of the data has been used to validate that the network is generalizing and to stop training before overfitting, these are used to measure network generalization, and to halt training when generalization stops improving.
- The last 20% of the data has been used as a completely independent test of network generalization, these have no effect on training and so provide an independent measure of network performance during and after training.

We used a log-sigmoid transfer function for the Hidden part of the network and a linear transfer function for the Output part of the network (see Figure 3 and Figure 4). Moreover to train the neural network we used different number of delays and ten hidden neurons. There are plenty of the training algorithms available in neural network. In this article three algorithms have been used to train the NARX and NAR network: Levenberg-Marquardt (see [41] and [46]), Scaled Conjugate Gradient (see [50]), and

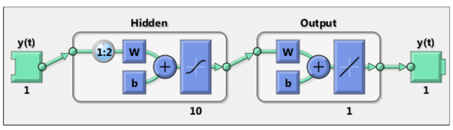


Figure 4: The NAR network with 2 number of delays and 10 hidden neurons (source http://www.mathworks.com).

Table 2: Tests statistics for Bitcoin closing prices, Bitcoin returns and oscillators from January 1, 2013 until May 31, 2018 (p-values in parentheses).

Variable	Jarque-Bera Normality Test	Shapiro-Wilk Normality Test	Ljung-Box (2) Autocorrelation Test	Ljung-Box (4) Autocorrelation Test	White Noise Test
Bitcoin	6726.600	0.547	3911.600	7767.300	2844.306
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
Bitcoin returns	7926.000	0.870	3.126	3.687	-0.209
	(0.000)	(0.000)	(0.210)	(0.450)	(0.917)
MACD(12)	96546.000	0.466	3840.400	7340.400	739.814
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
ROC(10)	866880.000	0.613	2142.800	3708.500	212.182
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
RSI(14)	29.364	0.986	3396.500	6176.900	1157.402
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
Momentum(10) of ROC	510710.000	0.583	1770.500	2777.900	192.509
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)

Bayesian Regularization (see [45] and [22]). Levenberg-Marquardt (L-M) is a network training function that updates weight and bias values according to Levenberg-Marquardt optimization, this is a simple method for approximating a function. One of the main drawbacks of the Levenberg-Marquardt algorithm is that, for certain problems it needs the large storage of some matrices. Scaled Conjugate Gradient algorithm (S-C-G) can train any network as long as its weight, net input, and transfer functions that have derivative functions. Bayesian regularization algorithm (B-R), a suitable method for estimation when a large number of inputs is used for best output, minimizes a grouping of squared errors and weights, and then determines the correct combination so as to produce a network that generalizes well.

5 EMPIRICAL RESULTS

In Table 3 are reported the MSE, calculated on the testing sample, and R, for different training algorithms and different delays, for estimated NAR models, in Table 4, Table 5 and Table 6 those for NARX models. The testing sample is roughly 395 days, so one year and a month. As we can see for those tables, on the basis of MSE, the best models for Bitcoin are NAR with 3 delays estimated with Levenberg-Marquardt algorithm and NARX with 2 delays and input series with 3 days of lag estimated with Bayesian regularization. Moreover the best models for Bitcoin for imput series with 6 and 9 days of lag are NARX with 14 delays estimated with Levenberg-Marquardt method and NARX with 3 delays estimated with Bayesian regularization, respectively. When selecting the models we considered also the correlation coefficient R between outputs and targets computed on all sample period (training, validating and testing samples). In Table 7 are presented the test statistics for forecasting errors of the mentioned models. All the series exhibit non normality, uncorrelation and randomness. In Figure 5 we reported the plots of MSE for estimated NAR and NARX models for different estimation algorithms and different lags for input series. We can see from Figure 5 that, as an overall performance, the best models are NARX with 3 days of lag for input series, NARX with 6 days of lag for input series, both estimated with Bayesian regularization, and NAR models trained with Levenberg-Marquardt optimization and Scaled Conjugate Gradient algorithm.

Table 3: MSE and R for estimated NAR models.

	Levenberg-Marquardt		Bayesian Regularization		Scaled Conjugate Gradient	
	MSE	R	MSE	R	MSE	R
NAR delay 1	30.959	0.197	104.385	0.143	42.029	0.063
NAR delay 2	29.840	0.199	34.628	0.338	29.499	0.178
NAR delay 3	29.033	0.223	34.673	0.398	29.461	0.015
NAR delay 4	30.150	0.139	29.614	0.039	30.045	0.093
NAR delay 5	29.615	0.206	29.555	0.074	30.299	0.029
NAR delay 6	30.333	0.203	29.836	0.097	30.274	0.103
NAR delay 7	30.890	0.324	29.811	0.107	35.752	0.059
NAR delay 8	29.969	0.271	42.106	0.483	31.641	0.064
NAR delay 9	30.871	0.366	30.072	0.109	29.489	0.106
NAR delay 10	30.908	0.154	29.823	0.121	30.844	0.111
NAR delay 11	31.019	0.135	29.920	0.137	31.068	0.120
NAR delay 12	34.069	0.449	47.057	0.517	31.344	0.088
NAR delay 13	36.878	0.517	30.054	0.136	31.268	0.066
NAR delay 14	38.423	0.513	30.190	0.141	30.693	0.152

Table 4: MSE and R for estimated NARX models using the input series with 3 days of delay.

	Levenberg-Marquardt		Bayesian Regularization		Scaled Conjugate Gradient	
	MSE	R	MSE	R	MSE	R
NARX delay 1	171.358	0.064	30.412	0.093	40.141	0.015
NARX delay 2	80.200	0.159	30.220	0.096	35.439	0.106
NARX delay 3	184.820	0.116	30.229	0.097	37.293	0.056
NARX delay 4	69.071	0.090	30.412	0.092	138.135	0.035
NARX delay 5	59.071	0.102	36.193	0.088	49.798	0.059
NARX delay 6	38.891	0.115	34.020	0.106	113.572	0.042
NARX delay 7	142.320	0.167	33.560	0.115	42.193	0.071
NARX delay 8	304.065	0.104	32.971	0.122	89.368	0.035
NARX delay 9	33.184	0.132	32.971	0.123	44.967	0.073
NARX delay 10	30.439	0.088	34.752	0.119	126.490	0.034
NARX delay 11	48.562	0.076	35.891	0.126	200.187	0.006
NARX delay 12	100.053	0.067	34.325	0.126	91.908	0.083
NARX delay 13	159.400	0.077	33.410	0.134	380.511	0.003
NARX delay 14	228.350	0.145	32.741	0.142	249.955	0.032

Table 5: MSE and R for estimated NARX models using the input series with 6 days of delay.

	Levenberg-Marquardt		Bayesian Regularization		Scaled Conjugate Gradient	
	MSE	R	MSE	R	MSE	R
NARX delay 1	227.108	0.047	40.208	0.044	95.824	0.028
NARX delay 2	274.629	0.032	41.538	0.056	75.477	0.010
NARX delay 3	246.005	0.036	35.420	0.074	43.915	0.066
NARX delay 4	210.093	0.072	34.342	0.083	93.261	0.038
NARX delay 5	55.939	0.165	32.916	0.099	60.316	0.052
NARX delay 6	30.298	0.141	34.034	0.106	56.413	0.029
NARX delay 7	87.074	0.148	37.195	0.103	32.971	0.132
NARX delay 8	40.092	0.057	37.304	0.107	33.498	0.132
NARX delay 9	57.367	0.192	36.116	0.114	36.696	0.131
NARX delay 10	34.486	0.159	34.471	0.123	51.118	0.061
NARX delay 11	31.428	0.097	34.810	0.132	48.989	0.031
NARX delay 12	137.843	0.128	33.233	0.141	62.282	0.003
NARX delay 13	123.011	0.206	33.310	0.143	52.785	0.036
NARX delay 14	30.446	0.066	34.281	0.140	99.426	0.002

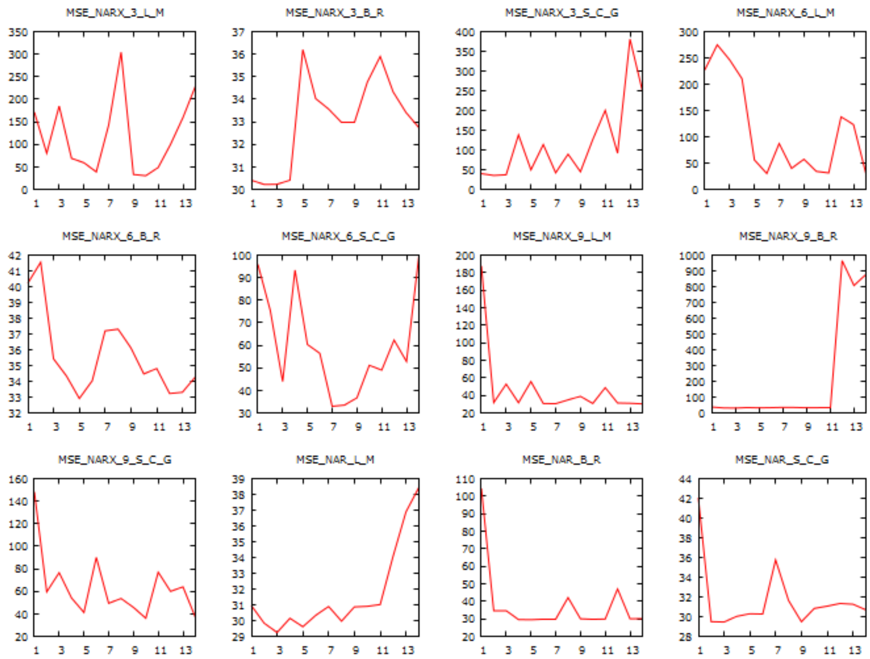


Figure 5: Plot of MSE against delay for estimated NAR and NARX models, for Bitcoin returns, using different lags for input series and different estimation algorithms.

6 CONCLUDING REMARKS

Cryptocurrencies have recently gained a lot of interest from investors, central banks and governments worldwide. The lack of any form of political regulation and their market far from being efficient, require new forms of regulation in the near future. From an econometric viewpoint to understand the process underlying the evolution of cryptocurrencies returns is essential. In this paper we perform a systematic comparison of NAR and NARX artificial neural network models in terms of predicting returns for Bitcoin. We found that the MSE for best NAR and NARX models is roughly the same and that the forecasting errors are non normal, uncorrelated and random.

Table 6: MSE and R for estimated NARX models using the input series with 9 days of delay.

	Levenberg-Marquardt		Bayesian Regularization		Scaled Conjugate Gradient	
	MSE	R	MSE	R	MSE	R
NARX delay 1	187.498	0.123	37.817	0.113	148.030	0.082
NARX delay 2	31.867	0.114	31.833	0.124	59.470	0.077
NARX delay 3	52.863	0.105	30.075	0.097	76.457	0.071
NARX delay 4	31.715	0.055	34.460	0.109	54.122	0.063
NARX delay 5	55.755	0.141	32.997	0.117	41.233	0.051
NARX delay 6	30.756	0.008	34.095	0.098	90.036	0.091
NARX delay 7	30.639	0.021	35.245	0.088	49.390	0.003
NARX delay 8	34.871	0.025	34.755	0.081	53.660	0.015
NARX delay 9	38.931	0.102	33.333	0.081	45.734	0.016
NARX delay 10	30.877	0.076	32.926	0.082	36.141	0.011
NARX delay 11	48.841	0.004	32.745	0.066	76.917	0.096
NARX delay 12	31.348	0.049	963.986	0.067	59.936	0.065
NARX delay 13	31.092	0.066	806.648	0.038	63.912	0.047
NARX delay 14	30.391	0.007	875.996	0.057	36.996	0.065

Table 7: Test statistics for forecasting errors of best NAR models and NARX models (p-values in parentheses).

Variable	Jarque-Bera Normality Test	Shapiro-Wilk Normality Test	Ljung-Box (2) Autocorrelation Test	Ljung-Box (4) Autocorrelation Test	White Noise Test	MSE	Estimation Method
NAR delay 3	41.618 (0.000)	0.984 (0.000)	1.895 (0.388)	6.102 (0.192)	1.040 (0.603)	29.033	Levenberg-Marquardt
NARX(delay 2) 3 days	84.637 (0.000)	0.972 (0.000)	0.607 (0.738)	5.888 (0.208)	0.761 (0.703)	30.220	Bayesian Regularization
NARX(delay 14) 6 days	92.953 (0.000)	0.972 (0.000)	0.228 (0.892)	3.591 (0.464)	0.498 (0.803)	30.446	Levenberg-Marquardt
NARX(delay 3) 9 days	86.458 (0.000)	0.972 (0.000)	0.672 (0.715)	4.838 (0.304)	0.577 (0.773)	30.075	Bayesian Regularization

REFERENCES

[1] M. Adya and F. Collopy. How effective are neural networks at forecasting and prediction ? A review and evaluation. *Journal of Forecasting*, **17**:481–495, 1998.

[2] R. L. Andrews. Forecasting performance of structural time series models. *Journal of Business and Economic Statistics*, **12**:129–133, 1994.

[3] J. Angstenberger. Prediction of the s&p 500 index with neural networks. In *Neural Networks and Their Applications*, pages 143–152. John Wiley and Sons, Chichester, 1996.

[4] M. Baxter and R. G. King. Measuring business cycles: Approximate band-pass filters for economic time series. *Review of Economics and Statistics*, **81**:575–593, 1999.

[5] M.H. Beale, M. T. Hagan, and H.B. Demuth. *Neural Network Toolbox User Guide*. MathWorks, 2015.

[6] S. A. Billings. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. John Wiley and Sons, Chichester, 2013.

[7] G. E. P. Box and G. M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, San Francisco, CA, 1970.

[8] L. Catania and S. Grassi. Modelling crypto-currencies financial time-series. *Technical Report, SSRN Working paper*, 2018.

[9] L. Catania, S. Grassi, and F. Ravazzolo. Forecasting cryptocurrency financial time-series. *CAMP Working Paper Series*, 2018.

[10] L. Catania, S. Grassi, and F. Ravazzolo. Predicting the volatility of cryptocurrency time-series. *Technical Report, MIMEO*, 2018.

[11] M. Caudill. *Neural Networks Primer*. Miller Freeman Publications, San Francisco, 1989.

[12] M. Caudill and C. Butler. *Understanding Neural Networks: Computer Explorations, volume1 and 2*. MIT Press, Cambridge, 1992.

[13] S. Chan, J. Chu, and S. Nadarajah. Statistical analysis of the exchange rate of bitcoin. *PLoS ONE*, **10**, 2015.

[14] S. Chan, J. Chu, S. Nadarajah, and J. Osterrieder. A statistical analysis of cryptocurrencies. *Risk and Financial Management*, **10**, 2017.

[15] C. Chatfield. Neural networks: Forecasting breakthrough or passing fad ? *International Journal of Forecasting*, **9**:1–3, 1993.

[16] P. K. Clark. Trend reversion in real output and unemployment. *Journal of Econometrics*, **40**:15–32, 1989.

[17] F. X. Diebold and R. S. Mariano. Comparing predictive accuracy. *Journal of Business and Economic Statistics*, **13**:253–263, 1995.

[18] G. Dreyfus. *Neural Networks, Methodology and Applications*. Springer and Verlag, Berlin Heidelberg, 2005.

[19] J. Durbin and S. J. Koopman. *Time Series Analysis by State Space Methods*. Oxford University Press, Oxford, 2nd edition, 2012.

[20] B. Efron and R. J. Tibshirani. *An Introduction to the Boostrap*. Chapman and Hall, Boca Raton, 1993.

[21] R. F. Engle and C. W. J. Granger. Co-integration and error correction: representation, estimation, and testing. *Econometrica*, **55**:251–276, 1987.

[22] F.D. Foresee and M.T. Hagan. Gauss-newton approximation to bayesian regularization. In *Proceedings of the 1997 International Joint Conference on Neural Networks*, pages 1930–1935, 1997.

[23] E. Gately. *Neural networks for financial forecast*. John Wiley and Sons, New York, 1996.

[24] C. W. J. Granger and P. Newbold. *Forecasting Economic Time Series*. Academic Press, 2nd edition, 1986.

[25] M.T. Hagan, H.B. Demuth, and M.H. Beale. *Neural Network Design*. PWS Publishing, Boston, 1996.

[26] J. D. Hamilton. *Time Series Analysis*. Princeston University Press, 1994.

[27] A. C. Harvey. *Forecasting Structural Time Series Models and the Kalman Filter*. Cambridge University Press, Cambridge, 1989.

[28] A. C. Harvey. *The Econometric Analysis of Time Series*. MIT Press, Boston, MA, 2nd edition, 1990.

[29] A. C. Harvey. *Time Series Models*. Harvester Wheatsheaf, Hemel Hempstead, 2nd edition, 1993.

[30] A. C. Harvey and A. Jaeger. Detrending, stylized facts and the business cycle. *Journal of Applied Econometrics*, **8**:231–247, 1993.

[31] A. C. Harvey and S. J. Koopman. Signal extraction and the formulation of unobserved components models. *Econometrics Journal*, **3**:84–107, 2000.

[32] A. C. Harvey and P. H. J. Todd. Forecasting economic time series with structural and box-jenkins models: A case study. *Journal of Business and Economic Statistics*, **1**:299–307, 1983.

[33] A. C. Harvey and T. Trimbur. Generalised model-based filters for extracting trends and cycles in economic time series. *Review of Economics and Statistics*, **85**:244–255, 2003.

[34] H. C. Harvey and D. Delle Monache. Specification and misspecification of unobserved components models. In *Economic Time Series: Modelling and Seasonality*, pages 83–108. Chapman and Hall/CRC Press, London, 2012.

[35] S. Haykin. *Neural networks: A comprehensive foundation*. Prentice Hall, New Jersey, 1999.

[36] R. J. Hodrick and E. C. Prescott. Post war us business cycles: An empirical investigation. *Journal of Money, Credit and Banking*, **24**:1–16, 1997.

[37] C. Hotz-Behofsits, F. Huber, and T. O. Zorner. Predicting cryptocurrencies using sparse non-gaussian state space models. *Journal of Forecasting*, **37**:627–640, 2018.

[38] N. Kohzadi, M. S. Boyd, I. Kaastra, B. S. Kermanshahi, and D. Scuse. Neural networks for forecasting: An introduction. *Canadian Journal of Agricultural Economics*, **43**:463–474, 1995.

[39] C. M. Kuan and H. White. Artificial neural networks: An econometrics perspective. *Econometric Review*, **13**:1–91, 1994.

[40] H. R. Kunsch. The jackknife and the bootstrap for general stationary observations. *Annals of Statistics*, **17**:1217–1241, 1989.

[41] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, **2**:164–168, 1944.

[42] H. Lutkepohl. *Applied Time Series Econometrics*. Cambridge University Press, Cambridge, 2004.

[43] H. Lutkepohl. *New Introduction to Multiple Time Series Analysis*. Springer-Verlag, Berlin, 2005.

[44] E. Maaoumi, A. Khotanzad, and A. Abaye. Artificial neural networks for some macroeconomic series: A first report. *Econometric Reviews*, **13**:105–122, 1994.

[45] D.J.C. MacKay. Bayesian interpolation. *Neural Computation*, **4**:3:415–447, 1992.

[46] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, **11**:2:431–441, 1963.

[47] K. Mehrotra, C. K. Mohan, and S. Ranka. *Elements of Artificial Neural Networks*. MIT Press, Cambridge, 1997.

[48] T. Mills. *Modelling Trends and Cycles in economic Time Series*. Palgrave MacMillan, New York, 2003.

[49] T. Mills and R. N. Markellos. *The Econometric Modelling of Financial Time Series*. Cambridge University Press, Cambridge, 3rd edition, 2008.

[50] M. F. Moller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, **6**:525–533, 1993.

[51] S. Moshiri and N. Cameron. Neural network versus econometrics models in forecasting inflation. *Journal of Forecasting*, **19**:201–217, 2000.

[52] E. Nakamura. Inflation forecasting using a neural network. *Economics Letters*, **86**:373–378, 2005.

[53] B. M. Onimode, J. K. Alhassan, and S. A. Adepoju. Comparative study of inflation rates forecasting using feed-forward artificial neural networks and auto regressive (AR) models. *IJCSI International Journal of Computer Science Issues*, **12**:2, 2015.

[54] DARPA Neural Network Study. *DARPA Neural Network Study*. MIT Lincoln Laboratory, Lexington, MA, 1998.

[55] N. R. Swanson and H. White. A model selection approach to real-time macroeconomic forecasting using linear models and artificial neural networks. *The Review of Economics and Statistics*, **79**:4:540–550, 1997.

[56] K. Taylor. *NEURAL NETWORKS TIME SERIES using MATLAB. PREDICTION and MODELING*. Amazon Media EU, 2017.

[57] G. Tkacz. Neural network forecasting of canadian gdp growth. *International Journal of Forecasting*, **17**:57–69, 2001.

[58] G. Tkacz and S. Hu. Forecasting gdp growth using artificial neural networks. Bank of Canada Working Paper, 1999.

[59] C. Wang. Time series neural network systems in stock index forecasting. *Computer Modelling & New Technologies*, **19**(1B):57–61, 2015.

[60] H. White. *Artificial Neural Networks: Approximation and Learning*. Blackwell, Cambridge, 1992.

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

[3, 2, 1, 5, 7, 4, 15, 16, 19, 17, 20, 23, 28, 27, 29, 30, 31, 36, 32, 33, 40, 38, 39, 42, 48, 51, 44, 49, 52, 53, 55, 57, 58, 60, 61, 43, 26, 24, 21, 34, 6, 11, 25, 12, 35, 54, 18, 47, 56, 8, 14, 10, 13, 9, 59, 37]