Journal of Autonomous Intelligence (2020) Volume 3 Issue 2.
doi:10.32629/jai.v3i2.207

Original Article

Deep Learning and Autoregressive Approach for Prediction of Time Series Data

Akhter Mohiuddin Rather*

Department of Business Analytics, Great Lakes Institute of Management, Gurgaon

ABSTRACT

A deep neural network based approach for prediction of non-stationary data has been proposed in this work. A new regression scheme has been used in the proposed model. Any non-stationary data can be used to test the efficiency of the proposed model, however in this work stock data has been used due to the fact that stock data has a property of being nonlinear or non-stationary in nature. Beside using proposed model, predictions were also obtained using some statistical models and artificial neural networks. Traditional statistical models did not yield any expected results; artificial neural networks resulted into high time complexity. Therefore, deep learning approach seemed to be the best method as of today in dealing with such problems wherein time complexity and excellent predictions are of concern.

Keywords: *Artificial Neural Networks; Autoregressive Model; Deep Learning; Time series; Keras*

ARTICLE INFO
Received: Nov 9, 2020
Accepted: Jan 5, 2021
Available online: Jan 22, 2021

***CORRESPONDING AUTHOR**
Akhter Mohiuddin Rather, Department of Business Analytics, Great Lakes Institute of Management, Gurgaon;
akhter.m@greatlakes.edu.in;

CITATION
Akhter Mohiuddin Rather. Deep learning and autoregressive approach for prediction of time series data. Journal of Autonomous Intelligence 2020; 3(2): 1-10. doi: 10.32629/jai.v3i2.207

1. Introduction

Prediction of non-stationary data by means of various statistical and artificial intelligence (AI) based models has always remained an area of much interest to various researchers and scholars. This area has received attention not just from academia but from industries too. Non-stationary data maybe be arranged in a time-series fashion and many statistical models are available in literature which work on time-series data. Much work has been already done in this field, however there are still many challenges left. For instance, the challenges are: calculating excellent predictions with least possible error, time complexity etc.

Broad classification of prediction-based models can lead us into two major subcategories: statistical models and AI based models. Some well known statistical models, also known as traditional models are: autoregressive integrated moving average (ARIMA) models[6], exponential smoothing models[7], autoregressive conditional heteroskedasticity (ARCH) models[17], generalized autoregressive conditional heteroskedasticity (GARCH)[5] etc. AI based models include soft computing based models such as artificial neural networks (ANN) inspired from human brain[20] which have given rise to deep neural networks[48], genetic algorithms (GA)[21], support vector machines (SVM)[9], fuzzy sets[57] etc.

The limitations of linear statistical models is that they are unable to capture the patterns of non-stationary behaviour present in the data, therefore the results are not satisfactory. Due to their stochastic nature and by using some nonlinear functions, AI based models are able to capture the non-stationary behaviour present in data therefore the results are pleasing most of the time. AI based models also have some limitations, one of the big-

gest limitations is high computation time required to arrive at the solution.

The remainder of this paper is organized as follows: In Sections 2, literature review is presented, which discusses about the contributions/proposed models and theories by various researchers over a period of time. Section 3 and Section 4 discuss about autoregressive integrated moving average model and artificial neural networks respectively. In Section 5, proposed model, its mathematical formulation as well as about its novelty is discussed in detail. In section 6 experiments and results are presented. Finally, conclusions are presented in section 7.

2. Literature Review

Apart from the broad categorization discussed in previous section; many researchers and scholars focussed on hybrid or integrated prediction-based models. A hybrid prediction-based model can be obtained by integrating the properties of various individual prediction-based models, thereby resulting into a powerful integrated or hybrid model. The goal of any hybrid prediction-based model is that its predictive efficiency should outperform the individual models used to form the hybrid model. Different hybrid prediction-based models are available in literature proposed by researchers over a period of time[1–4].

Deep Learning (DL) or Deep neural networks (DNN) have gained momentum over the recent years. DL is currently one of the favourite areas of research in many universities, research institutes and in industries around the globe due to its number of merits. Many industries have shifted their businesses in DL so as to obtain the solutions of hard and complex problems with greater efficiency and with least possible error. Recently Kim *et al*[29] demonstrated a new method of converting DNN into a spiking ANN without any loss of accuracy thus resulting into greater efficiency. Earlier Chang[10] studied about the deep and shallow architecture of ANN. For time series prediction a novel approach was proposed in which metaheuristic based DL approach was used[15]. Kraus & Feuerriegel[34] studied DL approach for financial domain in decision support field. DNN and DL have gained momentum very rapidly and are being applied almost in every field, for instance in the field of text mining and text analytics[49,53], self driving cars and object detection[25,43,55], healthcare [33,44,47], image and video processing[16,27,38]. Recently Lin *et al*[37] used DL in accelerating the topology optimization method of conductive heat transfer.

In the field of predictive analytics and time series forecasting such as stock market prediction, ANNs have dominated statistical models. Many researchers identified the shortcomings in statistical models, therefore they have obtained excellent results by using AI based models in time series based stock prices/returns prediction[30–35]. Because of its numerous advantages, DNNs and DL are replacing classical ANNs. Many researchers and scholars are now finding new methods or improving existing methods in the area of stock prediction by using DNN and DL[14,18,36]. Since DNN and DL are emerging areas, much more improvements are expected to take place in financial domain too, hopefully in forecasting stock prices, formation of an optimal portfolio etc.

3. Autoregressive Integrated Moving Average Models

ARIMA models are linear statistical models which work on the concept of regression on itself. ARIMA models was proposed by Box & Jenkins[6] and the work is regarded as one of the finest contribution in times series analysis. The work of Box & Jenkins[6] connects with the earlier work of Yule[56] and Wold[54]. ARIMA model can be constructed from two individual models, autoregressive model, *AR(p)* and moving average model, *MA(q)*. Integration of both models result into a combined model in the form of *ARIMA(p, d, q)*; where p is an integer referred as order of *AR* model, d is differencing order so as to convert non-stationary into stationary and q is an order of *MA* model.

The mathematical formulation of *AR(p)* model is given in Equation 1.

$$y_t = \alpha + \omega_1 y_{t-1} + \omega_2 y_{t-2} + \ldots + \omega_p y_{t-p} + \varepsilon_t \quad (1)$$

where α is a constant intercept; $\omega_i (i = i, \ldots, p)$ are constants which are coefficients of the lagged terms; ε_t is

an error term, also known as white noise having mean equal to 0 and variance equal to σ_ε^2.

In moving average, time series depends only on q past random terms and a present random term ε_t. Therefore, mathematical formulation of $MA(q)$ model is given in Equation 2.

$$y_t = \beta + \phi_1\varepsilon_{t-1} - \phi_2\varepsilon_{t-2} - \ldots - \phi_q\varepsilon_{t-q} + \varepsilon_t \quad (2)$$

where β is a constant term; $\phi i(i = i, \ldots, q)$ are constants which are coefficients of the lagged terms; ε_t is an error term, also known as white noise having mean equal to 0 and variance equal to σ_ε^2.

The final combined model i.e. $ARIMA(p, d, q)$ model is formulated in Equation 3.

$$y_t = \omega_1 y_{t-1} + \omega_2 y_{t-2} + \ldots + \omega_p y_{t-p} + \varepsilon_t - \phi_1\varepsilon_{t-1} - \phi_2\varepsilon_{t-2} - \ldots - \phi_q\varepsilon_{t-q} \quad (3)$$

Where ω and ϕ are regression coefficients and ε_t is an error term.

4. Artificial Neural Networks and Deep Learning

ANN is a learning based AI model inspired from human brain[20]. ANNs were initially used on logic gates[40] and afterwards lot of development happened in this field. **Figure 1** shows most commonly used ANN with one hidden layer.

The data is fed into ANN from input layer wherein there are n number of neurons $(x_1, \ldots, x_n)$ representing n number of inputs. There are m number of neurons in hidden layer, $(h_1, \ldots, h_m)$ wherein data is passed through nonlinear activation functions such as sigmoid function, hyperbolic tangent function etc. Finally output is obtained from output layer wherein there are z number of output neurons, $(o_1, \ldots, o_z)$. The connection links between input layer to hidden layer and from hidden layer to output layer are associated with random weights, w_{ij} and w_{jk} respectively.

At each m hidden neuron in hidden layer, weighted sum is calculated as shown in Equation 4.

$$L_m = \sum_{i=1}^{n} x_i w_{mi} \quad (4)$$

L_m is passed through a nonlinear activation function as shown in Equation 5.

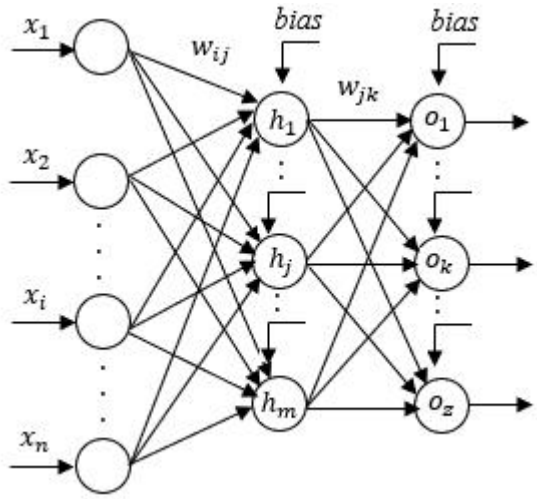

Figure 1. ANN with one hidden layer.

$$Y_m = \sigma(\beta + L_m) \quad (5)$$

where σ is a nonlinear activation function; β is a bias input to hidden neurons and output neuron(s).

Number of neurons in input and output layer is determined as per the data is arranged. Number of neurons in hidden layers(s) can be chosen based on trial and error method. There can be one hidden layer in ANN or more than one hidden layers present in ANN. Such ANN is also known as multilayer perceptron (MLP)[40] which works on backpropagation algorithm[8]. Since the network shown in Fig 1 takes data from input layer, processes it in hidden layer(s) and produces output from output layer, therefore this ANN is also called as feedforward neural network. For every data pattern (number of inputs and its corresponding output) ANN produces high error in the beginning. The learning takes place iteration wise, where from each iteration, error is

produced. A set of iterations is called one epoch. It is expected that with the passage of epochs, error reaches to minimum preset value. A network may take hundreds, thousands or millions of epochs to converge[20]. Therefore time complexity is a concern in ANN.

DNN is an advanced, more complicated and more sophisticated form of ANN. Unlike in traditional ANN such as MLP, DNN has multiple hidden layers present. There is no such definition which states how many layers should be there in any ANN to call it a DNN. DNNs are known with their depth i.e. number of hidden layers present through which data passes. In DNN, each layer trains on different set of features received from the previous layer. Both ANNs and DNNs use backpropagation algorithm wherein error minimization happens using gradient descent method[52] and weights gets updated, resulting into better weights after each epoch. As stated earlier, sigmoid activation function has been very popular in ANNs, however use of other activation functions such as rectified linear unit (ReLU), softmax, softplus, hardsigmoid etc have gained momentum in DNNs. Since time complexity is a concern in ANN, traditional ANNs can take long time to arrive at a solution. To avoid such problem, few application programming interfaces (API) were built. Keras is one such interface which was written in python and is able to run on top of Tenser flow.

5. Proposed Model

The proposed model assumes non-stationary data to be arranged in a time series fashion. The autoregression is calculated on time series non-stationary data and implemented on *DNN*. Suppose we are given with time series data Y having length of past historical series equal to T as shown in Equation 6.

$$Y = (y_{t-(T-1)}, \ldots y_{t-2}, y_{t-1}, y_t) \tag{6}$$

A new name is given to the proposed model as, autoregressive deep neural network, *ARDNN(a, b)*, where a is regression order, b is the reference point in past historical series. The location of b is determined as per the chosen order of regression. Moving reference variable $c_{tk}(k = 1, \ldots, T\text{-}a)$ is also to be calculated as shown in Equation 7.

$$c_{tk} = y_{t-(a-1)-b} \tag{7}$$

The future estimate or predictions are obtained using *ARDNN(a, b)* as per Equation 8.

$$(y_{t+1} - c_{tk})_{ARDNN} = \psi(y_{t-(p-1)} - c_{tk}, \ldots, y_{t-1} - c_{tk}, y_t - c_{tk}) \tag{8}$$

where $(y_{t+1} - c_{tk})_{ARDNN}$ is output of *ARDNN* and ψ is its predictor. The regression model when implemented on *DNN* is roughly shown in **Figure 2**.

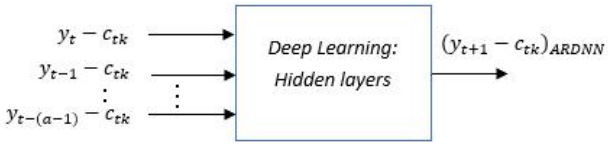

Figure 2. ARDNN.

Since actual output is being shown to *DNN* at output neuron, therefore this network learns in supervised manner. Predictions obtained from *ARDNN* are collected and corresponding reference values, i.e. c_{tk} values are added back which were once subtracted from original time series. This final process constitutes the final predictions of non-stationary data as shown in Equation 9.

$$\hat{y}_{t+1} = y_{t+1} - c_{tk} + c_{tk} \tag{9}$$

Where $\hat{y}_{t+1}$ is final predictions after adding c values back.

5.1 Sliding windows for ARDNN

Input to *ARDNN* can be given using sliding windows, where each window gives a prediction. **Figure 3** shows the arrangement of sliding windows using *ARDNN*. Suppose the time series data ranges between y_t and $Y_{t-(T-1)}$, where y_t is the latest observation, $Y_{t-(T-1)}$ is the last observation and T is the length of time series data. The first step of forming sliding windows for *ARDNN* is to divide the time-series data into two sets: training set and test set. Let the training set be in the range between last observation i.e. $y_{t-(T-1)}$ to $y_{t-(l-1)}$ and let test set be in the range between y_{t-1} to y_t as shown in **Figure 3**.

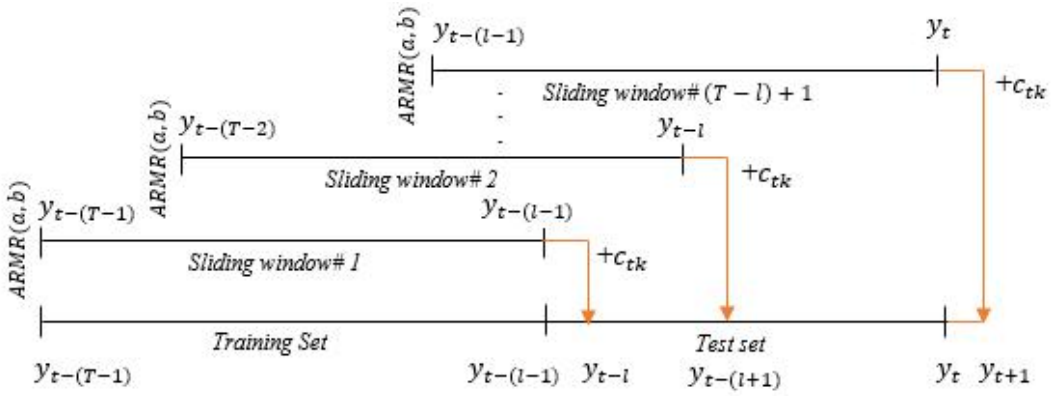

Figure 3. Sliding windows in ARDNN.

As shown in **Figure 3**, first sliding window is formed equal to length of training data which gives prediction y_{t-l}. Second sliding window is formed of same length but one observation slided ahead, giving prediction $y_{t-(l+1)}$. Final window, i.e. sliding window $(T-l) + 1$ gives prediction for future period i.e. y_{t+1}. In each window *ARDNN(a, b)* regression is performed as described earlier.

Therefore the input-output pairs formed and ready to fed to *DNN* are actually differenced values as described in Equation 8. c_{tk} values which were once subtracted from original time-series are added back to the output obtained from *DNN*, thus forming the final predictions.

6. Experiments & Results

Experiments were carried out on time series based stock prices using *ARDNN*. Beside using proposed model, experiments were also carried using traditional *ARIMA* model, so that comparison of models can be done.

6.1 Data

Stock prices of six stocks out of top 500 companies was obtained from National stock exchange (NSE) of India. These six stocks are listed in **Table 1**. Mean and standard deviation of each stock was calculated as shown in **Table 2**. The length of time series of each stock was considered for 164 days (daily stock price) between 20th Feb, 2018 to 19th Oct, 2018. However, there's no such restriction to consider only 164 observations. The longer the time series data, the better the predictions.

Table 1. List of six stocks obtained from NSE

S1	S2	S3	S4	S5	S6
ABB India	Adani Power	Bajaj Electricals	Castrol India Ltd	Coal India	GAIL(India)

Table 2. Mean and standard deviation of stocks

Stock	S1	S2	S3	S4	S5	S6
μ	1286.7617	25.5442	561.2858	170.2014	277.5499	347.6444
σ	95.2982	5.2810	47.5736	18.2724	9.5750	20.0394

6.2 Predictive performance of ARIMA

ARIMA model of regression order 4 was used to obtain predictions for all stocks. **Figure 4** shows prediction of stock 1 up to 32 future observations using ARIMA. The x-axis shows 32 future observations in 95% and 99% confidence intervals (shaded areas) and stock price is shown on y-axis.

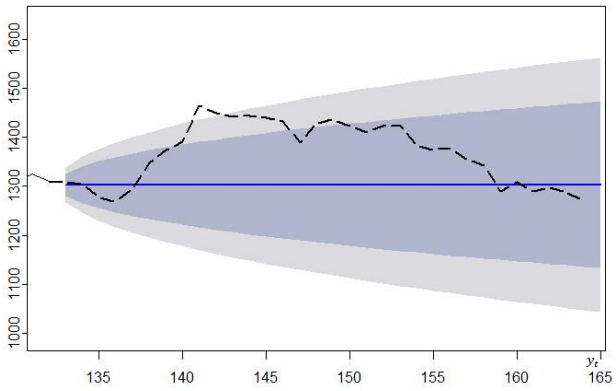

Figure 4. Time series plot of ARIMA for stock 1.

Similar nature was observed for rest of the stocks using ARIMA. It is observed that in all six stocks, ARIMA model has not been able to capture the patterns, thus the predictive performance is unsatisfactory.

6.3 Predictive performance of ANN and ARDNN

For each stock of 164 observations, training and test set were formed in the ratio of 80 : 20, i.e. 80% for training set and rest 20% for test set. Training set consisted of 132 observations between 20th Feb, 2018 to 30th Aug, 2018. Test set consisted of 32 observations between 31st Aug, 2018 to 19th Oct, 2018. 33 sliding windows were formed where in each window *ARDNN*(4, 1) regression was performed, i.e. regression order was chosen equal to 4. Therefore there were 4222 input-output pairs for each stock (33 windows * 128 input-output pairs in each window). Thus for entire test data, 32 predictions were obtained plus one future prediction, i.e. y_{t+1}.

6.3.1 Implementation of ANN and its results

Feedforward ANN using backpropagation algorithm of the configuration 4 : 12 : 1 (4 neurons in input layer; 12 neurons in hidden layer and an output neuron in output layer) was chosen for the experiments. This ANN can also be called as autoregressive neural network (ARNN) as it considers same input-outpur pairs as ARDNN. Sigmoid activation funcion was used in hidden neurons and learnining rate was kep as 0.20. On an average it took over 5000 epoch for ANN to converge (for each stock)

Figure 5 shows result of ARNN for stock1 in the form of time series plot. The figure shows target data, trained data as well as future predictions i.e. unseen or test data.

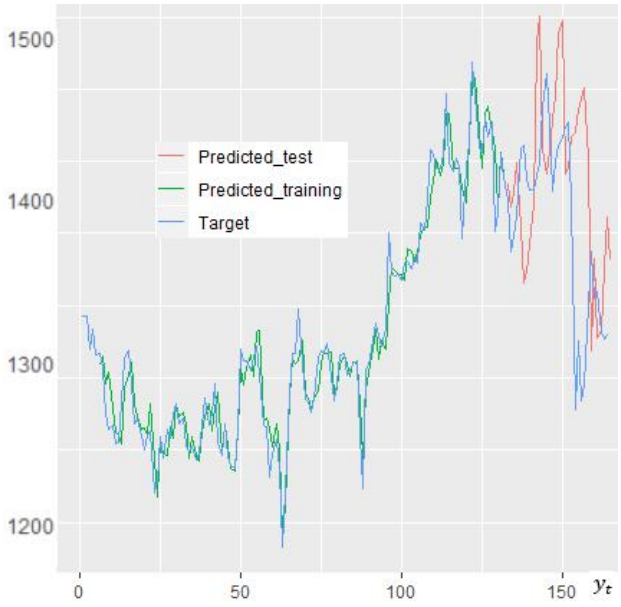

Figure 5. Time series plot of ARNN for stock 1.

ARNN is able to capture the patterns of the stock, but there is still scope of improvement.

6.3.2 Implementation of ARDNN and its results

The proposed model was implemented and executed in python programming by calling Keras library. Configuration of *DNN* was chosen as 4 : 100 : 50 : 20 : 1, 4 neurons in input layer, one neuron in output layer, three hidden layers having 100, 50 and 20 neurons respectively. Relu activation function was used in hidden layers. For each stock 20% of data was kept for cross validation which helps to reduce overfiting. Number of epochs for *DNN* was prespecified to 100 and batch size was kept as 32. The above configuration was chosen after trial and error.

The result was that the prediction error decreased rapidly during training period of *DNN*. As already mentioned, the predictions obtained from *ARDNN* are to

be added with c_{tk} values, thereafter MSE and MAE can be calculated for test set.

Figure 6 shows result of *ARDNN* for stock 1 in the form of time series plot. The figure shows target data, trained data as well as future predictions i.e. unseen or test data.

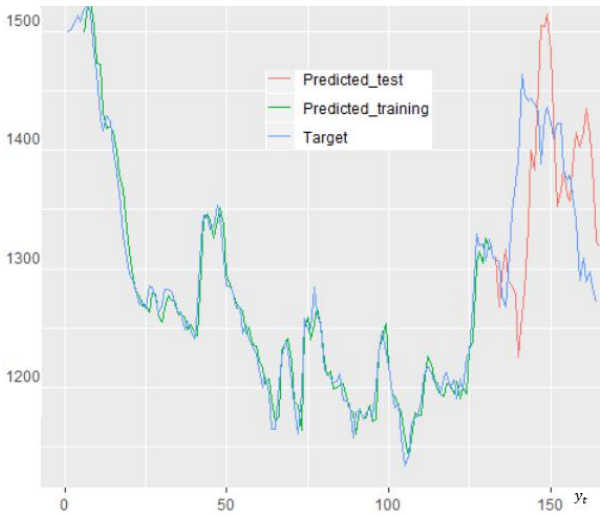

Figure 6. Time series plot of ARDNN for stock 1.

It is observed that the proposed model has been able to capture the patterns of stocks which states that the model is robust enough to work on prediction of non-stationary data.

6.4 Comparison

Based of the results obtained for six models; a comparison between ARIMA, *ARNN* and *ARDNN* is carried out here. **Table 3** shows MSE calculated for all six stocks for *ARDNN*, *ARNN* as well as for ARIMA model. As expected *ARDNN* outperforms both *ARNN* and ARIMA model in terms of least prediction error for all stocks.

Table 3. Mean-Squared Error

Stock	S1	S2	S3	S4	S5	S6
ARDNN						
	7710.73	19.02	1012.31	47.46	69.67	270.36
ARNN						
	7720.75	29.12	1202.55	74.45	88.56	295.78
ARIMA						
	7739.03	39.49	1257.99	164.37	118.76	302.02

7. Conclusions

An attempt was made to calculate the predictions of non-stationary data from the perspective of deep neural networks using keras library. The field of deep learning seems to be a very promising field in present world as it is being used in most of the areas. The future of deep learning and the areas where it is being used or explored also seems to be bright as it overcomes the limitations of existing statistical models including traditional AI models such as ANNs. One reason of the popularity of

deep learning is that it is very fast as well as accurate. The limitations of this work is that the proposed model may not always capture the patterns of nonlinear data. The future work includes to explore the model using some latest deep learning libraries and improve its predictive performance. This is certainly an important avenue for future research.

Acknowledgement

Research support by Great Lakes Institute of Management is highly acknowledged.

Conflict of Interest Statement

The authors certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

References

1. Adhikari, R., & Agrawal, R. (2014). A combination of artificial neural network and random walk models for financial time series forecasting. Neural Computing and Applications, 24 , 1441 – 1449.
2. Arajo, R., Oliveira, A., & Meira, S. (2015). A hybrid model for high-frequency stock market forecasting. Expert Systems with Applications, 42 , 4081 – 4096.
3. Araujo, R. (2010). A hybrid intelligent morphological approach for stock market forecasting. Neural Processing Letters, 31 , 195 – 217.
4. Asadi, S., Hadavandi, E., Mehmanpazir, F., & Nakhostin, M. (2012). Hybridization of evolutionary levenbergmarquardt neural networks and data pre-processing for stock market predictionl. Knowledge-Based Systems, 35 , 245 – 258.
5. Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. Journal of Econometrics, 31 , 307 – 327.
6. Box, G., & Jenkins, G. (1970). Time series analysis, forecasting and control. San Francisco: Holden-Day.
7. Brown, R. (2004). Smoothing, forecasting and prediction of discrete time series. Mineola, N.Y.: Courier Dover Publications.
8. Bryson, A., Ho, Y., & Siouris, G. (1979). Applied optimal control: Optimization, estimation, and control. IEEE Transactions on Systems, Man, and Cybernetics, 9 , 366 – 367.
9. Burges, C. (1998). A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2 , 121 – 167.
10. Chang, C. (2015). Deep and shallow architecture of multilayer neural networks. IEEE Transactions on Neural Networks and Learning Systems, 26 , 2477 – 2486.
11. Chen, M., & Chen, B. (2015). A hybrid fuzzy time series model based on granular computing for stock price forecasting. Information Sciences, 294 , 227 – 241.
12. Chen, M., Chen, D. R., Fan, M. H., & Huang, T. Y. (2013). International transmission of stock market movements: an adaptive neuro-fuzzy inference system for analysis of TAIEX forecasting. Neural Computing and Applications, 23 , 369 – 378.
13. Chen, Y., Cheng, C., & Tsai, W. (2014). Modeling fitting-function-based fuzzy time series patterns for evolving stock index forecasting. Applied Intelligence, 41 , 327 – 347.
14. Chong, E., Han, C., & Park, F. (2017). Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. Expert Systems with Applications, 83 , 187 – 205.
15. Coelho, L., Coelho, V., Luz, E., Ochi, L., & Frederico, G. (2017). A gpu deep learning metaheuristic based model for time series forecasting. Applied Energy, 201 , 412 – 418.
16. D.Ravi, Szczotka, A. B., Shakir, D., Pereira, S., & Vercauteren, T. (2018). Effective deep learning training for single-image super-resolution in endomicroscopy exploiting video-registration-based reconstruction. International Journal of Computer Assisted Radiology and Surgery, 13 , 917 – 924.
17. Engle, R. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. Econometrica, 50 , 987 – 1007.
18. Fang, Y. (2018). Feature selection, deep neural network and trend prediction. Journal of Shanghai Jiaotong University (Science), 23 , 297 – 307.
19. Freitas, F., de Souza, A., & de Almeida, A. (2009). Prediction-based portfolio optimization using neural networks. Neurocomputing, 72 , 2155 – 2170.
20. Haykin, S. (1994). Neural networks: A comprehensive foundation. Saddle River: Prentice-Hall.
21. Holland, J. H. (1992). Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and artificial intelligence. Cambridge, MA, USA: MIT Press.
22. Hsieh, T., Hsiao, H., & Yeh, W. (2011). Forecasting

stock markets using wavelet transforms and recurrent neural networks: An integrated system based on artificial bee colony algorithm. Applied Soft Computing, 11 , 2510 – 2525.
23. Hsu, V. M. (2013). A hybrid procedure with feature selection for resolving stock/futures price forecasting problems. Neural Computing and Applications, 22 , 651 – 671.
24. Huang, C., & Tsai, C. (2009). A hybrid sofm-svr with a filter-based feature selection for stock market forecasting. Expert Systems with Applications, 36 , 1529 – 1539.
25. Huang, X., Sun, J., & Sun, J. (2018). A car-following model considering asymmetric driving behavior based on long short-term memory neural networks. Transportation Research Part C: Emerging Technologies, 95 , 346 – 362.
26. Kao, L., Chiu, C., Lu, C., & Chang, C. (2013). A hybrid approach by integrating wavelet-based feature extraction with mars and svr for stock index forecasting. Decision Support Systems, 54 , 1228 – 1244.
27. Kappeler, A., Yoo, S., Dai, Q., & Katsaggelos, A. (2016). Video superresolution with convolutional neural networks. IEEE Transactions on Computational Imaging, 2 , 109 – 122.
28. Khashei, M., & Bijari, M. (2010). An artificial neural network (p, d, q) model for timeseries forecasting. Expert Systems with Applications, 37 , 479 – 489.
29. Kim, J., Kim, H., Huh, S., Lee, J., & Choi, K. (2018). Deep neural networks with weighted spikes. Neurocomputing, 311 , 273 – 386.
30. Kim, K., & Lee, W. (2004). Stock market prediction using artificial neural networks with optimal feature transformation. Neural Computing & Applications, 13 , 255 – 260.
31. Kim, K. J., & Ahn, H. (2012). Simultaneous optimization of artificial neural networks for financial forecasting. Applied Intelligence, 36 , 887 – 898.
32. Kim, M., Han, I., & Lee, K. (2004). Hybrid knowledge integration using the fuzzy genetic algorithm: prediction of the korea stock price index. Intelligent Systems in Accounting, Finance and Managemen, 12 , 43 – 60.
33. Kollias, D., Tagaris, A., Stafylopatis, A., Kollias, S., & Tagaris, G. (2018). Deep neural architectures for prediction in healthcare. Complex & Intelligent Systems, 4 , 119 – 131.
34. Kraus, M., & Feuerriegel, S. (2017). Decision support from financial disclosures with deep neural networks and transfer learning. Decision Support Systems, 104 , 38 – 48.
35. Kwon, Y., & Moon, B. (2007). A hybrid neurogenetic approach for stock forecasting. IEEE Transactions on Neural Networks, 18 , 851 – 864.
36. Lachiheb, O., & Gouider, M. (2018). A hierarchical deep neural network design for stock returns prediction. Procedia Computer Science, 126 , 264 – 272.
37. Lin, Q., Hong, J., Liu, Z., Li, B., & Wangl, J. (2018). Investigation into the topology optimization for conductive heat transfer based on deep learning approach. International Communications in Heat and Mass Transfer , 97 , 103 – 109.
38. Liu, D., Wang, Z., Fan, Y., Liu, X., Wang, Z., Chang, S., Wang, X., & Huang, T. (2018). Learning temporal dynamics for video super-resolution: A deep learning approach. IEEE Transactions on Image Processing, 27 , 3432 – 3445.
39. Lu, C. (2013). Hybridizing nonlinear independent component analysis and support vector regression with particle swarm optimization for stock index forecasting. Neural Computing and Applications, 23 , 2417 – 2427.
40. McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5 , 115 – 133.
41. Moghaddam, A., Moghaddam, M., & Esfandyari, M. (2016). Stock market index prediction using artificial neural network. Journal of Economics, Finance and Administrative Science, 21 , 89 – 93.
42. Pai, P., & Lin, C. (2005). A hybrid arima and support vector machines model in stock price forecasting. Omega, 33 , 497 – 505.
43. Pathak, A., Pandey, M., & Rautaray, S. (2018). Application of deep learning for object detection. Procedia Computer Science, 132 , 1706 – 1717.
44. Purushotham, S., Meng, C., Che, Z., & Liu, Y. (2018). Benchmarking deep learning models on large healthcare datasets. Journal of Biomedical Informatics, 83 , 112 – 134.
45. Rather, A. (2014). A hybrid intelligent method of predicting stock returns. Advances in Artificial Neural Systems, 2014.
46. Rather, A., Agarwal, A., & Sastry, V. (2015). Recurrent neural network and a hybrid model for prediction of stock returns. Expert Systems with Applications, 42 , 3234 – 3241.
47. Sannino, G., & Pietro, G. (2018). A deep learning approach for ecg-based heartbeat classification for arrhythmia detection. Future Generation Computer Systems, 86 , 446 – 455.
48. Shao, L., Wu, D., & Li, X. (2014). Learning deep and wide: A spectral method for learning deep networks. IEEE Transactions on Neural Networks and Learning Systems, 25 , 2303 – 2308.
49. Tsumoto, S., Kimura, T., Iwata, H., & Hirano, S. (2017). Mining text for disease diagnosis. Procedia Computer Science, 122 , 1133 – 1140.

50. Wang, J., & Wang, J. (2015). Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks. Neurocomputing, 156 , 68 – 78.
51. Wang, J. J., Wang, J. Z., Zhang, Z. G., & Guo, S. P. (2012). Stock index forecasting based on a hybrid model. Omega, 40 , 758 – 766.
52. Wang, L., Yang, Y., Min, R., & Chakradhar, S. (2017). Accelerating deep neural network training with inconsistent stochastic gradient descent. Neural Networks, 93 , 219 – 229.
53. Wang, Y., & Xu, W. (2018). Leveraging deep learning with lda-based text analytics to detect automobile insurance fraud. Decision Support Systems, 105 , 87 – 95.
54. Wold, H. (1938). A study in the analysis of stationary time series. Stockholm: Almqvist and Wiksells.
55. Yu, Y., Guan, H., & Ji, Z. (2015). Rotation-invariant object detection in high-resolution satellite imagery using superpixel-based deep hough forests. IEEE Geoscience and Remote Sensing Letters, 12 , 2183 – 2187.
56. Yule, G. (1926). Why do we sometimes get nonsense correlations between time series? a study in sampling and the nature of time series. Journal of the Royal Statistical Society, 89 , 30 – 41.
57. Zadeh, L. (1965). Fuzzy sets. Information and Control, 8 , 338 – 353.
58. Zhang, G. (2003a). Time series forecasting using a hybrid arima and neural network model. Neurocomputing, 50 , 159 – 175.
59. Zhang, G. P. (2003b). Time series forecasting using a hybrid arima and neural network model. Neurocomputing, 50 , 159 – 175.
60. Zheng, Y., Lin, Z., & Tay, D. (2001). State-dependent vector hybrid linear and nonlinear arma modeling: Applications. Circuits, Systems and Signal Processing, 20 , 575 – 597.