Journal of System Management (JSM)
Online ISSN: 2538-1571, Print ISSN: 2322-2301
Doi: 10.30495/JSM.2022.1954072.1623
8(4), 2022, pp. 1-17

Received: 02/03/2022
Accepted: 14/08/2022

RESEARCH ARTICLE **Open Access**

A Stock Market Prediction Model Based on Deep Learning Networks

Seyyedeh Mozhgan Beheshti Masalegou [1], Mohammad Ali Afshar Kazemie[2]*, Jalal Haghighat Monfared[3], Ali Rezaeian[4]

Abstract
Accurate stock market prediction can assist in an efficient portfolio and risk management. However, accurately predicting stock price trends still is an elusive goal, not only because the stock market is affected by policies, market environment, and market sentiment, but also because stock price data is inherently complex, noisy, and nonlinear. Recently, the rapid development of deep learning can make the classifiers more robust, which can be used to solve nonlinear problems. This study proposes a hybrid framework using Long Short-Term Memory, Autoencoder, and Deep Neural Networks (LSTM-AE-DNNs). Specifically, LSTM-AE is responsible for extracting relevant features, and in order to predict price movement, the features are fed into two deep learning models based on a recurrent neural network (RNN) and multilayer perceptron (MLP). The dataset used for this is Dow Jones daily stock for 2008-2018, which was used in this article. Besides, to further assess the prediction performance of the proposed model, original stock features are fed to the single RNN and MLP models. The results showed that the proposed model gives the more accurate and best results compared to another. In particular, LSTM-AE+RNN shows a better performance than the LSTM-AE+MLP. In addition, hybrid models show better performance compared to a single DNN fed with the all-stock features directly.

Keywords: Stock market prediction, Deep learning, Dimensionality reduction, Long-Short term memory Autoencoder (LSTM-AE)

Introduction

Financial markets are the lifeblood of the global economy, transferring trillions of dollars daily. A good forecast of market behavior in the future would be extremely useful in various situations. The stock market affects economic growth(Barboza et al., 2017; Hoseinzade & Haratizadeh, 2019). so, analyzing their behavior and predicting their future can be very helpful in achieving their financial goal (Hoseinzade & Haratizadeh, 2019). However, Stock price prediction is challenging because it is highly volatile, nonlinear, and dynamic. It is influenced by various elements such as political situations, the economy, trends, seasonality, investor psychology, and so on (Yadav et al., 2020).

1. Department of Information Technology Management, Tehran Central Branch, Islamic Azad University, Tehran, Iran
2*. Department of Industrial Management, Tehran Central Branch, Islamic Azad University, Tehran, Iran (Corresponding Author: Dr.mafshar@gmail.com)
3. Department of Industrial Management, Tehran Central Branch, Islamic Azad University, Tehran, Iran
4. Department of Governmental Management, Faculty of Management and Accounting, Shahid Beheshti university, Tehran, Iran

However, Stock price prediction is challenging because it is highly volatile, nonlinear, and dynamic. It is influenced by various elements such as political situations, the economy, trends, seasonality, investor psychology, and so on (Fama, 2021; Jin et al., 2019). So, making the right decision within a timely response has posed several challenges as such a large amount of information is required to predict the stock market price movement.

For stock price prediction, there are three standard methods: technical analysis, traditional time series forecasting, and machine learning. Traditional time series prediction algorithms use parametric statistical models such as autoregressive moving average (ARMA), autoregressive integrated moving average (ARIMA), and vector autoregression to find the best estimate. Although these econometric models are appropriate for describing and analyzing the relationships between variables using statistical inference, they have some drawbacks when analyzing financial time series. First, because they presume a linear model structure, they cannot represent the nonlinear nature of stock prices (Dietrich et al., 1999; Selvamuthu et al., 2019). Traditional time series prediction algorithms use parametric statistical models such as autoregressive moving average (ARMA), autoregressive integrated moving average (ARIMA), and vector autoregression to find the best estimate. Although these econometric models are appropriate for describing and analyzing the relationships between variables using statistical inference, they have some drawbacks when analyzing financial time series. First, because they presume a linear model structure, they cannot represent the non-linear nature of stock prices (Baek & Kim, 2018; Box et al., 2015).

Furthermore, these methods frequently necessitate assumptions and prior knowledge, such as the underlying data distribution, valid ranges for various parameters, and connections. However, because the stock market is complex with many influential factors and uncertainties, it exhibits non-linear solid characteristics, rendering conventional analytical methods ineffective. Furthermore, the amount of information processed by stock market modeling and forecasting is frequently considerable, posing significant challenges for algorithm design. Because of these qualities, traditional methods for stock market prediction are ineffective (Jin et al., 2019). Machine learning (ML) has evolved as an essential analytical tool in financial markets, used to assist and manage investment efficiently (Modjtahedi & Daneshvar, 2020). In the financial sector, ML has been widely used to provide a new mechanism that can help investors in making better decisions(Lin et al., 2018; Obthong et al., 2020). Artificial Neural networks (ANNs), an important branch of machine learning algorithms, have the following advantages over traditional statistical methods: numerical, data-driven, and adaptive. As a result, ANNs have a more remarkable ability to analyze inaccurate and noisy data and have been widely used to predict time series (Yu & Yan, 2020). Random Forest (RF), Linear Discriminant Analysis (LDA), Logistic Regression (LR), and Evolutionary Computation approaches are the other preferred methods in economic research (Barboza et al., 2017). The major problem with machine learning algorithms is that their performance depends heavily upon the representation of data they are given. Since the behavioral stock market is complex, non-linear, and noisy, selecting the right features gets more complicated. Additionally, as the feature space grows more extensive, the model's training time starts to increase, and the model's outputs become more challenging to comprehend (Hoseinzade & Haratizadeh, 2019). Due to the fact that a high-dimensional feature space results in poor generalization in machine learning models, dimensionality reduction is used to remove the detrimental impacts of high dimensionality and data sparsity.

It is challenging to discover an acceptable extraction strategy with non-linear and noisy data when employing feature extraction methods to lower an expanding feature space(Gunduz, 2021; Zhong & Enke, 2017). In recent publications, deep learning (DL) models have been suggested as a powerful alternative to feature extraction approaches; models can be considered feature extractors that create sophisticated feature representations from raw data or more minor features at varying degrees of abstraction in each layer (Gündüz et al., 2017). Autoencoder (AE) proposed by Hinton is favored by scholars for dimensionality reduction. It can extract spatial feature vector in a lower dimension from high dimensional training samples by multiple hidden layers without loss of information. However, AE can just model the spatial correlation of data but fails to capture the temporal correlation of time-series data. To address this problem, the recurrent neural network (RNN) is introduced to capture the implicit dependence of time-series data. Long short-term memory (LSTM), as a replacement of traditional RNN, is designed for time-series modelling, overcoming the problem called gradient vanish or explode. Many extensions of LSTM have been applied to feature extraction problems, for example, video representation and diagnosis of arrhythmia (Yang et al., 2020). Intuitively, LSTM combined with auto-encoder (LSTM-AE) can extract the features and reduce dimension. So, to avoid missing representative features, we should select many features as far as possible when using machine learning algorithms in stock trading. Meanwhile, these high-dimensional features can lead to the redundancy of information and reduce the efficiency and accuracy of learning algorithms. It is worth noting that dimensionality reduction operation (DRO) is one of the main means to deal with stock high-dimensional data. However, there are few studies on whether DRO can significantly improve the trading performance of deep neural network (DNN) algorithms. To address these challenges, this study proposes a deep learning-based stock market prediction model based on long short-term memory, Autoencoder, and deep neural networks. First, LSTM-AE is applied as a feature-extraction technique to extract important features from high-dimensional time-series data and remove redundant features. Then, the features are fed into two deep learning models based on a Recurrent Neural Network (RNN) and Multi-Layer Perceptron (MLP) to forecast stock prices. The hybrid framework named LSTM-AE-DNNs could capture more stochasticity within the stock price. Also, a single DNN (RNN&MLP) fed with the all-stock data is used as a based-line to evaluate the hybrid model. The quality of the proposed model is assessed through MSE, and the results are compared to show that the features with dimensional reduction are more discriminative than the original data.

The main contribution of this paper is:

A hybrid deep learning model comprises Long Short-term Memory Autoencoder as feature extraction and RNN, MLP as a forecaster. In the former studies, the role of the feature extractor is stack autoencoder Bao et al. (2017), and variational autoencoder Gunduz (2021) are taken place.

Literature Review

Prediction of the stock price has attracted more attention from investors to gain higher returns, and this has led researchers to propose various predicting models (Rather et al., 2017; Xu et al., 2020). The stock market prediction has been a source of contention for decades. Bachelier (1900)conducted the first study on stock behavior (Salmani Danglani et al., 2019). This was the first study to characterize stock price movement as a random walk. Furthermore, the Efficient Market Hypothesis by Malkiel and Fama states that stock prices are informationally efficient, meaning that they reflect all available information and that it is not possible to predict stock prices based on trading data or to obtain excess returns by exploiting any price predictability(Taheri et al., 2019). Several studies, on the other hand, have attempted to scientifically invalidate the efficient market theory, and empirical data has demonstrated that stock markets are, to a degree, predictable (Baek & Kim, 2018). Time series prediction methods that have been used in the past to find the best estimations employ parametric statistical models (Box et al., 2015). Linear models like AR, ARMA, ARIMA (Baek & Kim, 2018; Lin et al., 2018) have been used for stock market forecasting. For instance, Srivastava et al. (2022) used the ARIMA model to forecast stock prices, and the results showed that this model is suitable for short-term stock price prediction. These models are useful for describing and evaluating the relationships between variables by statistical inference, but they have limits for financial time-series analysis. First, they cannot capture the non-linear stock price behavior since they presume linear model structure. Financial time-series are very noisy and have time-varying volatility. In order to overcome the limitation of linear models, Machine learning models are often used to make accurate predictions in financial research. These models use various sources of information to obtain financially relevant characteristics (Cavalcante et al., 2016).Due to its capacity to deal with the non-linear and dynamic character of markets, SVM is a leading model in financial prediction.; for example, Henrique et al. (2018) used a machine learning technique called Support Vector Machine (SVM) to predict large and small capitalizations and in three different markets, employing price with both daily and up-to-the-minute frequencies. Prediction errors were measured, and the model was compered to the random walk model proposed by the EMH. Their results suggested that SVM had predicted power, especially when using a strategy of updating the model periodically. Doğan et al. (2022) aimed to develop an effective prediction model with Support Vector Machine and Logistic Regression Analysis. As the field of the study, 172 firms that were traded in Bosra İstanbul, have been chosen. Besides, two basic prediction methods, LRA was also used as a feature selection method and the results of this model were compared. Their empirical results showed, both methods achieved a good prediction model. However, the SVM model in which the feature selection phase was applied showed the best performance. Zhong and Enke (2019) offered those 60 macroeconomic and microeconomic variables over ten years be utilized to forecast the S&P index's daily return. Their prediction includes a phase to reduce the dimension and a step to classify the data. Principal component analysis (PCA) and rapid PCA were employed to reduce the dimension of the dataset, and ANN was chosen as the classifier model. PCA and ANN achieved the highest accuracy rates across all experiment configurations. Patel et al. (2015) predicted the direction of movement of Indian indices; this research used widely used tools such as ANN, SVM, random forest, and Naive Bayes Stocks. This study established that mapping data from a space conversion of ten technical variables to another feature space that corresponds to those variables' trends enhanced prediction performance. Due to the simplicity of shallow models, they may be incapable of mapping effectively from the input space to accurate predictions. Thus, due to the availability of enormous amounts of data and the development of effective methods for training deep models, academics have recently turned to such approaches for market forecasting. Deep models can extract vast sets of features from raw data. Several research have used RNN algorithms, particularly LSTM, to forecast time series. These approaches are used to examine time series data because they preserve historical data (Hoseinzade & Haratizadeh, 2019).Nelson et al. (2017)advocated using LSTMs to forecast the stock market. Technical indicators were fed into an LSTM for prediction, and the results indicate that LSTMs outperformed MLPs. Yadav et al. (2020) implemented LSTM model with various hidden layers to Indian stock market data removing the trend and seasonality components to predict the closing pric. Qiu et al. (2020) utilized LSTM to estimate stock prices, and the LSTM-based model demonstrated superior predictive accuracy. Bhandari et al. (2022) proposed a model based on LSTM network to predict the next-day closing price of the S&P 500 index. A well-balanced combination of nine predictors was carefully constructed under the umbrella of the fundamental market data, macroeconomic data, and technical indicators to capture the behavior of the stock market in a broader sense. They experimental results showed that the single layer LSTM model provides a superior fit and high prediction accuracy compared to multilayer LSTM models.

Additionally, many types of sequential data could be fed into networks to expand the information set available. An autoencoder can automatically extract features from the input. Thus, autoencoders are commonly employed to predict time series data. Albahli et al. (2022) presented the prediction of closing stock prices based on using ten years of Yahoo Finance data of ten renowned stocks and STIs by using an autoencoder. The experimental results showed that they proposed approach outperformed the state-of-the-art techniques by obtaining a minimum MAPE value of 0.41.Zhong and Enke (2019) used an autoencoder and a limit Boltzmann machine to forecast the stock market. They showed that without significant model previous knowledge, characteristics might be retrieved from vast raw. Numerous studies have demonstrated the ability of LSTM-autoencoders as a time series prediction tool. Stacked autoencoders with LSTM were employed to forecast the one-step-ahead closing price of six popular stock indices traded in multiple financial marketplaces, and they outperformed WLSTM (a mixture of WT and LSTM), LSTM, and the regular RNN in terms of prediction accuracy and profitability (Bao et al., 2017). Additionally, autoencoders combined with LSTM were found to be the ideal model in terms of root mean square error (RMSE) values for various training and test data sets. This demonstrates that these models have a greater capability for feature extraction than multilayer perceptron's, deep belief networks, and single LSTM(Gensler et al., 2016).Yan and Yang (2021) used LSTM neural networks in both the encoder and decoder for stock market prediction. They demonstrated that the model's efficacy in addressing stock forecasting issue. As a result, the combination of autoencoders with LSTM has demonstrated considerable potential in predicting time series.

As a result, this article proposes an LSTM-based autoencoder model for stock market dimension reduction. Autoencoder neural networks are chosen for learning a compressed representation of input data and changing the dimension of the input data, but the LSTM network is recommended for processing time-series data and identifying their temporal patterns.

Methodology

The main purpose of this research is to design a deep learning model in order to increase the accuracy of stock market prediction with a hybrid approach. Also, despite the inherent uncertainty and complexity of the stock market, Deep Neural Networks will be used instead of other Machin Learning methods. This model determines the relevant and important features to reach the minimum amount of error from the all-stock features with Dimensionality Reduction. This research method is descriptive-modeling, and the data collection method is library-field. The dataset used in this study includes the daily price of the Dow Jones Industrial Average. This data frame is from September 2008 to June 2016 (almost 5000 pieces of data). The historical data can be downloaded on Yahoo Finance. This experiment uses Intel i7-10700 8 core 16 thread processor, 16GB memory, win10 operating system, anaconda3 as the experimental platform, python language programming. The deep learning framework uses Keras in TensorFlow 2.0, which is powerful and concise. This section is divided into three subsections. One deals with the data that are primarily used for building the model. Another two sections deal with describing various theories and processes for building the models. Fig 1 shows the flow chart of this framework.

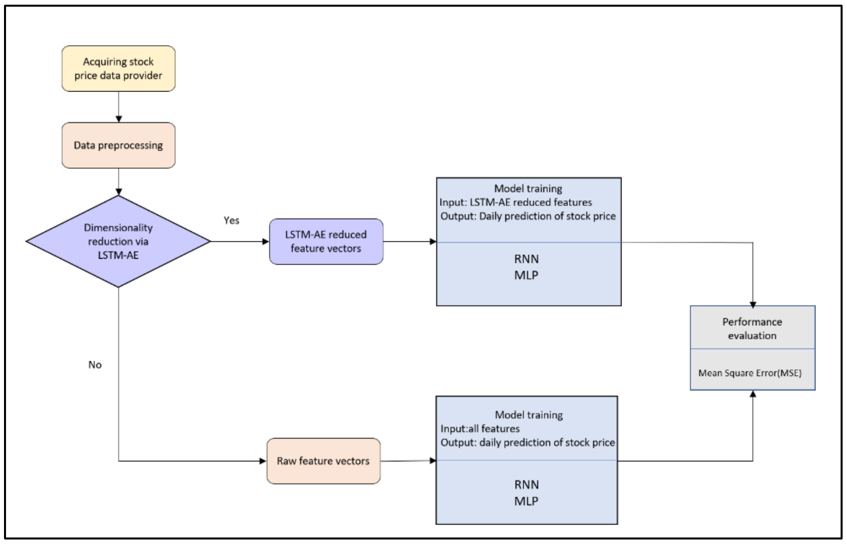

Acquiring stock price data provider
Data preprocessing
Dimensionality reduction via LSTM-AE
Yes
LSTM-AE reduced feature vectors
Model training
Input: LSTM-AE reduced features
Output: Daily prediction of stock price
RNN
MLP
Performance evaluation
Mean Square Error(MSE)
No
Raw feature vectors
Model training
Input:all features
Output: daily prediction of stock price
RNN
MLP

Figure 1. A graphical view of the proposed framework

Dataset preparation and preprocessing

Dataset: The dataset used in this study include the daily direction the Dow Jones Industrial Average. The daily trading data consists of open, high, low, closing, and adjusted closing prices, as well as the volume of trading and its assigned label, is determined according to the Eq. where Adj Close_t refers to the Adj Close at day t

$$\text{Target} = \begin{cases} 1 & \text{Adj Close}_{t+1} > \text{Adj Close}_t \\ 0 & \end{cases}$$

This data frame is from September 2008 to June 2018 (almost 5000 pieces of data). The historical data can be downloaded in Yahoo Finance.

Different variables may have a range of values. It is typically confusing for learning algorithms to deal with variables with a different range; data normalization attempts to map all variables' values to a single standard range. Additionally, it typically improves the prediction model's performance. This procedure was carried out using the following equation:

$$x' = \frac{(x - \min(x))}{\text{Max}(x) - \min(x)}$$

where x' is the normalized value, x min and x maxes are the minima and maximum values in the training data set.

Window size fixing: the window size is fixed by performing an error calculation on each window size which varies from 50 to 70. among these, the window size of 60 resulted in a minimum error than others. Table 1 shows that a time-series data set, the size of window 60, is used so that for the input of the price of sixty consecutive days, the output will be equal to the price of the sixty-first day, and by moving this window. The time-series data set will be formed.

Table 1.
The time-series data set

Stock Prices (Input) Index	Stock Price (Output) Index
1-60	**61**
2-61	**62**
3-63	**64**
...	**...**
(n-60) -(n-1)	**n**

Dimensionality reduction method

Dimensionality reduction (DR) is a preprocessing technique that reduces the complexity of machine learning models. DR increases such models' computational efficiency and prediction efficacy (Khalid et al., 2014; Kou et al., 2020). There are two types of DR: feature selection and feature extraction. Autoencoders, specifically the long short-term memory Autoencoder (LSTM-AE), can be applied to time-series data to directly learn robust deep feature representations (code) while decreasing the size of the feature space. LSTM-AE is based on the assumption that recurrent networks are better suited to modeling time series (Sagheer & Kotb, 2019), which we use in our study.

Long Short-Term Memory

The long short-term memory (LSTM) network is a recurrent neural network (RNN). RNNs are robust artificial neural networks that can remember their input. This form makes them ideal for challenges involving time series data. LSTMs can learn from inputs separated by lengthy time delays and have larger memories. Unimportant information is deleted by the forget gate, and the output gate picks what information to output. These three gates use the sigmoid function to work from 0 to 1. Fig. 2 shows three sigmoid gates. The cell state is represented by a horizontal line (Yadav et al., 2020).

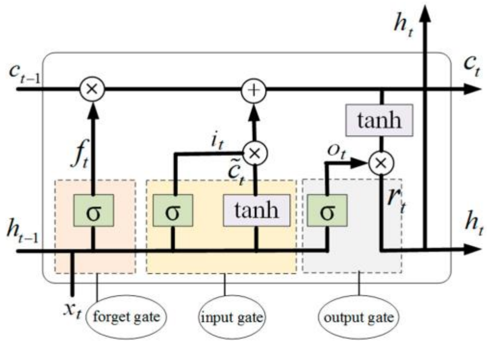


Figure 2. structural diagram of an LSTM neural node

$$f_t = \sigma\left(w_f[h_{t-1}, x_t] + b_f\right)$$
$$i_t = \sigma(w_i \cdot [h_{t-1}\,, x_t] + b_i)$$
$$\tilde{c} = tanh(W_c \cdot [h_{t-1}, x_t] + b_c$$
$$C_t = f_t * C_t - 1 + i_t * \tilde{C}_t$$
$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o)$$
$$h_t = o_{t^*}\, tanh(c_t)$$

Where f_t, i_t, and o_t indicate the forget, input, and output gates at time t, respectively. $C_t$is the cell state vector, and h_t represents the hidden state at the currenttime t. $tanh$ is the hyperbolic tangent function.

Autoencoder

Autoencoder is a subclass of unsupervised neural networks that utilize data to train the optimal encoding-decoding method. For self-supervised learning mainly contains an input layer, an output layer, and hidden layers. Although the output and input layers contain isomorphic vectors, the topology is comparable to a regular neural network. This model aims to generate a representation for an input data set (e.g., dimensionality reduction) and make the recognized data as similar to the input data possible (Jung & Choi, 2021). As shown in figure 3, the encoder shows a stage at which the model can learn significant characteristics of inputs and the decoder forms outputs similar to the inputs.

$$H = f(W_1 \cdot X + b)$$
$$\tilde{X} = \tilde{f}(W_2 \cdot H + b)$$

Where W_1 is the weight between input an X and hidden representation, $W_2$is the weight between a hidden representation H and output. $\tilde{X}$, and b is the bias, f and $\tilde{f}$represent the encoder and decoder, respectively, f accepts and compresses the input data (X)into a latent space (H) ,and$\tilde{f}$ is responsible for accepting latent space(H) representations and reconstructing original inputs $\tilde{X}$.

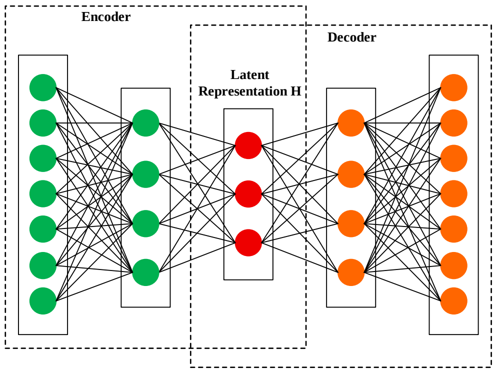


Figure 3. Structure of an autoencoder

An essential feature in the design of an autoencoder is that it reduces data dimensions while keeping the principal information of the data structure.

LSTM –AutoEncoder

LSTM autoencoder refers to the autoencoder that both the encoder and the decoder are the LSTM network. The ability of LSTM to learn patterns in data over long sequences makes them suitable for time series forecasting. That is, the use of the LSTM cell is to capture temporal dependencies in data. The input of each LSTM-AE is a time series sequnece , denoted in this work as v_t containes x_i where $i \in \{1,2,\ldots,I\}$ denote the index of the input data in each frame. The value of I is the same as the number of input data in a single frame. Each time series sequence is imported to a new encoder LSTM cell together with the hidden output from the previous.LSTM cell. The hidden output from the last LSTM cell of the encoder finally into a learned representation vector. This vector may be an entire sequence of hidden states from all the previous encoder LSTM cells. The equations for the encoder are:

$$h_t^{(E)} = RNN\left(v_t, h_{t-1}^{(E)}, C_t - 1\right)$$

$$f_t = w_{hf}^{(E)} + b_f^{(E)}$$

Where $h_t^{(E)}$and f_t denote the hidden states of the encoder and the learned represantation vector(the extracted features),respectively,at frame t. $f_t$is composed of f_{tn},where $n \in \{1,2,\dots,N\}$ and N denote the number of features.thus ,the extracted features of all frames are formated as a $T \times N$ matrix. f_t is calculated by using the weight w_{hf}^{E} And the biase $b_f^{(E)}$.

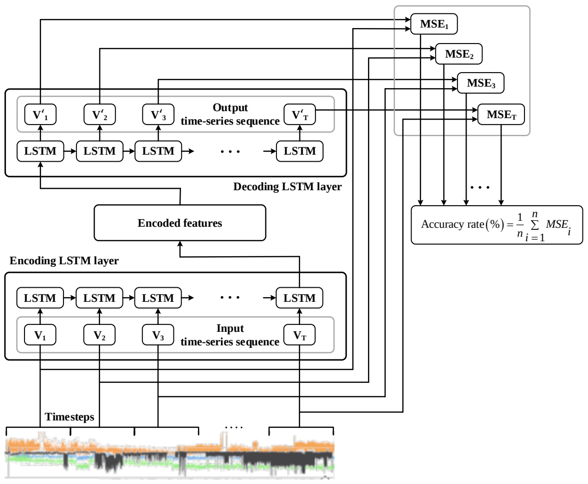


Figure 4. High-level Long Short- Term-Memory - autoencoder architecture

Then the decoder takes the encoded features as input to be processed through the various LSTM decoder cells and finally produces the output. The output of the decoder layer is a reconstruction of the initial input time series sequences. In other words, the dimensions are extracted in the decoding layer, and the features are used to reconstruct the data. The equations for the hidden state of the decoder$h_t^{(D)}$ and the reconstructed input data $\hat{v}_t$ are as follows:

$$h_t^{(D)} = RNN\left(f_t, h_{t-1}^{(D)}, C_t - 1\right)$$

$$\hat{v}_t = w_{h\hat{v}}^{D} h_t^{D} + b_{\hat{v}}^{D}$$

Where $W_{h\hat{v}}^{(D)}$ and $b_{\hat{v}}^{(D)}$ denote the weight and the biase, respectively,from the hidden states to the reconstructed data.

Based on their timestamp, data values are categorized and separated according to the number and kind of status chosen. The accuracy of the model is usually determined after the model training. Test samples are fed as input to the model, and the network compares the initial input values with the reconstructed ones. the mean square error of the difference between the reconstructed time series sequence, $\hat{v}_t$ and the initial input, v_t is the cost function of the LSTM-AE, as presented in the equation.

$$L = \sum_{I} (v_t - \hat{v}_t)^2$$

For each time sequence, a mean squared error is calculated. Then, the accuracy rate(%) of the LSTM-AE is obtained from the average calculation of these values. After the training of the set of LSTM-AE, obtained a good value, the decoder part of the model be removed, the encoder part is used as the feature, and this feature vector is fed into classifier models.A high-level architecture illustrating the proposed concept is provided in figure 4.

Classifcation models

Multilayer Perceptron (MLP):

MLP network, is a simple neural network. Each input neuron connects to the neurons of the following hidden layer through a weighted matrix w_{k_i}. A network is divided into three layers: input, hidden, and output (Moghaddam et al., 2016). In many situations, the MLP structure may have additional hidden layers that may cope with approximate solutions to various complex issues, such as reasonable approximation.

The equation for activation function of an i^{th} hidden neuron is given by

$$h_i = f(u_i) = f\left(\sum_{k=0}^{k} w_{k_i} x_k\right)$$

h_i : i^{th}hidden neuron, $f(u_i)$ link function which provides non-linearity between input and hidden layer,

w_{k_i}:weight in the $(k,i)^{th}$entry in a (KXN) weight matrix , x_k: K^{th} input value

$$y_j = f(u'_j) = f\left(\sum_{i=1}^{N} w'_{ij} h_i\right)$$

y_j: j^{th} output value.

Recurrent Neural Network(RNN)

The RNN is a deep neural network design (Dahl et al., 2011; Hinton et al., 2012) with a complex temporal structure. RNN receives input from two sources: the present and the past. They use information from these two sources to decide how to react to new data. Using a feedback loop, each moment's output becomes an input for the next. The recurrent neural network has memory. Each input sequence contains a lot of data, which is concealed in recurrent networks. The network uses this concealed information to deal with a new example (Hiransha et al., 2018). Input to hidden layer equation is given as:

$$h_t = f(W_{hh}h_{t-1} + w_{xh}x_t + b_h)$$
$$y_t = W_{hy}h_t + b_y$$

Whereas h_t:hidden layer at time : t^{th}instant, f :function, W_{xh}:input to hidden layer weight matrix, x_t:input at : t^{th} instant, b_h:biase or threshold value

Hidden to output layer equation is given as:

$$y_t = f(W_{hy}h_t + b_y)$$

$y_{t:}$ is the output vector at time t, W_{hy}:hidden to output layer weight matrix, b_y:biase or threshold

Evaluation metric

The following indicator is used to evaluate the test result: MSE(mean squared error). Its formula is shown as follows:

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(a_t - p_t)$$

The proposed hybrid method

efore setting up the LSTM-AE model, the data set was created that represents the stock market features. A sliding window moving from the beginning to the end of data was used for data preparation; timestep 60 was used for this study. A basic spilite was used to define, train, and test the data for LSTM-AE; 80% of the dataset was train data, and 20% was test data. Next, train and test data were standardized from 0 to 1, facilitating neural network training.

As illustrated in figure 5, the architecture of LSTM-AE initially includes an input layer where the size depends on the number of features selected; in this case, 60 features were selected. Then the first encoding LSTM layer reads the input data and outputs 60 features with timestep for each. The second encoding LSTM layer read the 60.60 input data from the first encoding LSTM layer and reduced the feature size to 50. Then the third encoding LSTM layer read the 60.50 input data from the third LSTM layer and reduced the feature size to 10. a repeat vector replicated the feature vector and prepared the 3D array input for the first LSTM layer in the decoder. Then the first decoding LSTM layer reads the 60.10 input data and outputs 60 .60.the second decoding LSTM layer reads the 60.60 input data and outputs 50 features with timestep for each. Then a time-distributed layer took the output and created a 60.50 (number of features output from the previous layer . number of features) vector. Finally, a matrix multiplication between the second decoding LSTM layer and the time distributed layer outputted the 60.6

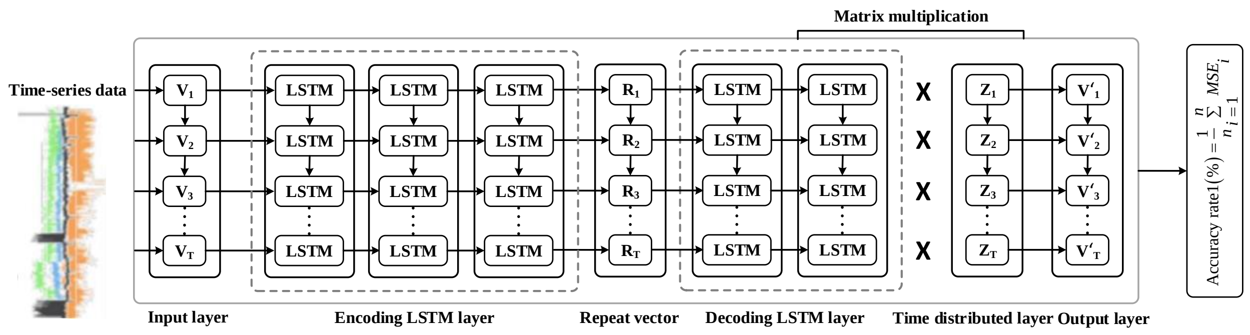


Figure 5. LSTM-AE architecture set

Once the model has achieved the appropriate level of performance, the decoder part is deleted, and the encoder part is used as a feature vector for classifier models. The last stage is to train LSTM and MLP prediction models. LSTM and MLP have numerous hidden levels and units in each hidden layer. Our hybrid model predicted Adj Closing price using four hidden layers with 50 neurons.

Result

The adj closing price of the Dow Jones Index was analyzed from September 2008 to June 2018 using different machine learning methods. Two sets of experiments were performed on the time series data for the stock, indicating intending to establish a reasonable model structure and the effectiveness of the proposed AE-LSTM approach in the analysis and predict financial time series. The mean square error was used in both experiments as the loss function and the Adam as the optimization algorithm. A batch size of 64 was used, and the number of epochs was fixed at 100. hold up method was used for train and test data, splitting 80% of stock selected for training and the remaining 20% testing.

Experimental results of feature selection

In the first experiment, reduced stock features were given as input to the models. LSTM-AE was used to reduce the size of the feature vectors while extracting deep and latent properties from the entire feature. The parameter of AE LSTM can be seen in table 2. the size of the feature vector was reduced from 60 to 10 with the help of the encoding component of the AE-LSTM model. Then reduced stock features were provided as inputs to the RNN and MLP models. The result of those with reduced features is shown in table 3.

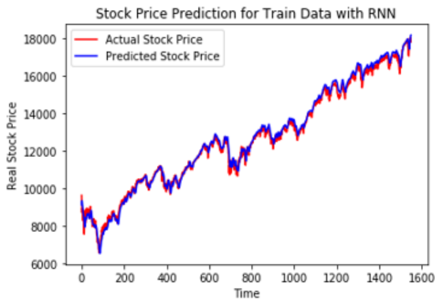
Table 2.
Parameter of LSTM-AE

parameters	value
Number of inputs	60
Structure of encoder layer	)10 (,output)50 (,hidden)60(input
Structure of decoder layer	)60 (,output)50 (,hidden)10(input
evaluation	) MSE(
epochs	15
Learning rate	Adam

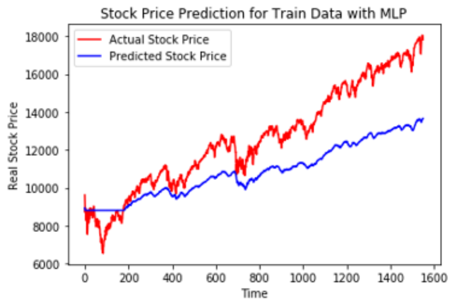
Table 3.
Classification results using LSTM-AE-reduced stock features

method	Train (MSE)	Test (MSE)
LSTM-AE+MLP	0.0270	**0.0771**
LSTM-AE+RNN	0.0014	**0.0020**

Two classifiers (Table 3) tested with LSTM-AE. LSTM-AE-RNN outperformed (0.0020) the LSTM-AE-MLP (0.0771).

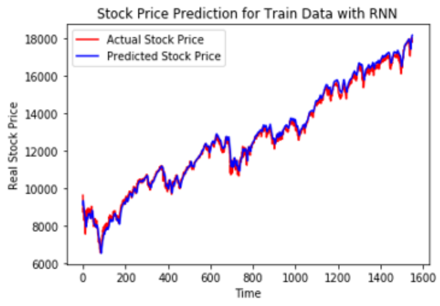


a

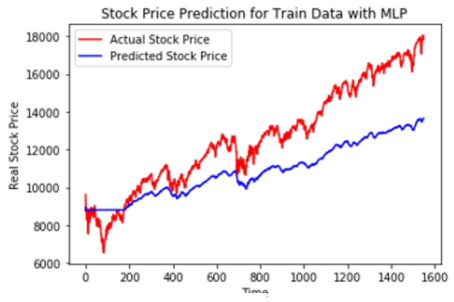


b

Figure 6: Visualization of the prediction results of the train part with dimensional reduction. (a)prediction results of the RNN model. (b)prediction results of MLP model



a



b

Figure 7: Visualization of the prediction results of the test part with dimensional reduction. (a)prediction results of the RNN model. (b)prediction results of MLP model

Figure 6 shows the trained data of RNN and MLP models tested with the trained data set of the stock market. Figure 6a RNN network is successful in capturing the seasonal pattern. From figure 6b MLP network failed to identify the pattern. Figure 7 shows the comparison between RNN and MLP prediction values and actual data. The prediction values and the real data are almost coincident, and the deviation between the prediction values and the actual data is so small, indicating the fact that the performance of RNN is better than MLP in the test data because there are significant errors between MLP prediction values and the actual data.

Experimental results of all own features

The second experiment was trained with the own stock feature, and the performance was assessed in terms of MSE. Two DNN models RNN and MLP used as classification. The parameter of the DNN models shown in table 4.

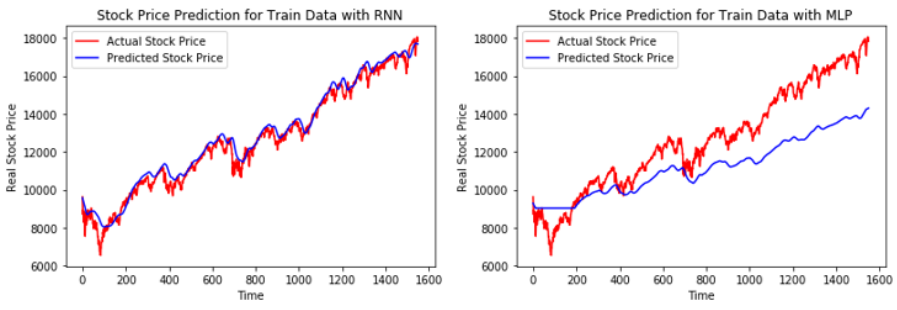
Table 4.
Parameter of DNNs models

parameters	**RNN**	**MLP**
Number of inputs	60	60
Number of hidden layers	4	4
Number of neurons in the hidden layer	50	50
evaluation	) MSE(	) MSE(
Number of epochs	100	100
Learning rate	Adam	Adam

Table 5.
Classification results using all stock features

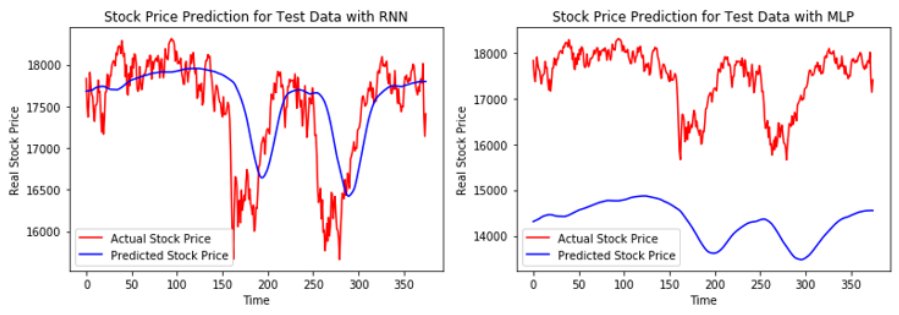
Method	MSE train	MSE TEST
MLP	0.0362	0.1186
RNN	0.0038	0.0654

Table 5 shows the results tasted with all-stock data. the RNN mode l was superior to the MLP models. RNN model has an MSE (0.065) and MLP has (0.1186).



a b

Figure 8: Visualization of the prediction results of the train part with the own stock features. (a)prediction results of the RNN model. (b)prediction results of MLP model.



a b

Figure 9: Visualization of the prediction results of the test part with the own stock features. (a)prediction results of the RNN model. (b)prediction results of MLP model

The trained data of RNN and MLP models were tested with the test data set of the stock. The test result is shown in figure 8. it can be seen from figure 8a that the waveform of the predictive and the actual curves for the RNN network are mostly consistent. From figure 8 b, it is depicted that the actual waveform of MLP and the prediction are also similar to a degree. Nevertheless, the price gap is significant. Fig 9 shows the comparison of RNN prediction values and actual data. From figure 9a, the deviation between the predicted values and the actual data is slight, but the price gap between 150 to 200 and 250 to 300 is significant. From figure 9, the b map cannot capture the pattern, and there is a large gap between prediction and actual stock price.

Conclusion

Predicting stock market prices is challenging and exciting in economics, science, and academic research. Recent developments in machine learning, especially DL, have made it possible to forecast future stock price movements based on past events. Advanced machine learning algorithms allow researchers to predict stocks using intelligent methods. In stock market prediction, feature extraction from stock data is necessary because the original data contain irrelevant information, which decreases the prediction of the results. This paper proposed a hybrid method for the trend prediction of stock based on dimensionality reduction and deep neural networks. In this paper, the feature extraction method using the LSTM-AE is proposed. The LSTM-AE aligns the original stock data more discriminatively and minimizes the unrelated information. Therefore, the features improve the prediction performance of DNNs. The results indicated that the LSTM-AE-DNN hybrid models perform better than the single DNN fed with the original stock data. In particular, LSTM-AE-RNN performed better than LSTM-AE-MLP. Furthermore, the two-step training models are more effective than the single-step training model. This paper focused on predicting stock prices in the US stock market. Furthermore, this study was limited to predicting closing prices at daily intervals, which remains to be investigated the possibility of predicting the stock price fluctuation on a time interval shorter than one day in future research work. This research requires more detailed historical stock market data not available in the current data set. This task requires another investigation and another series of experiments, which can be considered for future work.

References

Albahli, S., Nazir, T., Mehmood, A., Irtaza, A., Alkhalifah, A., & Albattah, W. (2022). AEI-DNET: a novel densenet model with an autoencoder for the stock market predictions using stock technical indicators. *Electronics*, *11*(4), 611. https://doi.org/10.3390/electronics11040611

Bachelier, L. (1900). Théorie de la spéculation. Annales scientifiques de l'École normale supérieure,

Baek, Y., & Kim, H. Y. (2018). ModAugNet: A new forecasting framework for stock market index value with an overfitting prevention LSTM module and a prediction LSTM module. *Expert Systems with Applications*, *113*, 457-480. https://doi.org/10.1016/j.eswa.2018.07.019

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one*, *12*(7), e0180944. https://doi.org/10.1371/journal.pone.0180944

Barboza, F., Kimura, H., & Altman, E. (2017). Machine learning models and bankruptcy prediction. *Expert Systems with Applications*, *83*, 405-417. https://doi.org/10.1016/j.eswa.2017.04.006

Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., Dahal, K. R., & Khatri, R. K. (2022). Predicting stock market index using LSTM. *Machine Learning with Applications*, 100320. https://doi.org/10.1016/j.mlwa.2022.100320

Box, G., Jenkins, G., Reinsel, G., & Ljung, G. (2015). Time series analysis, control, and forecasting . Hoboken. In: New Jersey: John Wiley & Sons. https://doi.org/10.1111/jtsa.12194

Cavalcante, R. C., Brasileiro, R. C., Souza, V. L., Nobrega, J. P., & Oliveira, A. L. (2016). Computational intelligence and financial markets: A survey and future directions. *Expert Systems with Applications*, *55*, 194-211. https://doi.org/10.1016/j.eswa.2016.02.006

Dahl, G. E., Yu, D., Deng, L., & Acero, A. (2011). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, *20*(1), 30-42. https://doi.org/10.1109/TASL.2011.2134090

Dietrich, R., Opper, M., & Sompolinsky, H. (1999). Statistical mechanics of support vector networks. *Physical review letters*, *82*(14), 2975. https://doi.org/10.1103/PhysRevLett.82.2975

Doğan, S., Koçak, D., & Atan, M. (2022). Financial Distress Prediction Using Support Vector Machines and Logistic Regression. In *Advances in Econometrics, Operational Research, Data Science and Actuarial Studies* (pp. 429-452). Springer. https://doi.org/10.1007/978-3-030-85254-2_26

Fama, E. F. (2021). *Market efficiency, long-term returns, and behavioral finance*. University of Chicago Press. 10.7208/9780226426983-019

Gensler, A., Henze, J., Sick, B., & Raabe, N. (2016). Deep Learning for solar power forecasting—An approach using AutoEncoder and LSTM Neural Networks. 2016 IEEE international conference on systems, man, and cybernetics (SMC), https://doi.org/10.1109/SMC.2016.7844673

Gunduz, H. (2021). An efficient stock market prediction model using hybrid feature reduction method based on variational autoencoders and recursive feature elimination. *Financial Innovation*, *7*(1), 1-24. https://doi.org/10.1186/s40854-021-00243-3

Gündüz, H., Çataltepe, Z., & Yaslan, Y. (2017). Stock daily return prediction using expanded features and feature selection. *Turkish Journal of Electrical Engineering & Computer Sciences*, *25*(6), 4829-4840. 10.3906/elk-1704-256

Henrique, B. M., Sobreiro, V. A., & Kimura, H. (2018). Stock price prediction using support vector regression on daily and up to the minute prices. *The Journal of finance and data science*, *4*(3), 183-201. https://doi.org/10.1016/j.jfds.2018.04.003

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., & Sainath, T. N. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, *29*(6), 82-97. https://doi.org/10.1109/MSP.2012.2205597

Hiransha, M., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. (2018). NSE stock market prediction using deep-learning models. *Procedia*

Computer Science, *132*, 1351-1362. https://doi.org/10.1016/j.procs.2018.05.050

Hoseinzade, E., & Haratizadeh, S. (2019). CNNpred: CNN-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*, *129*, 273-285. https://doi.org/10.1016/j.eswa.2019.03.029

Jin, Z., Yang, Y., & Liu, Y. (2019). Stock closing price prediction based on sentiment analysis and LSTM. *Neural Computing and Applications*, 1-17. https://doi.org/10.1007/s00521-019-04504-2

Jung, G., & Choi, S.-Y. (2021). Forecasting Foreign Exchange Volatility Using Deep Learning Autoencoder-LSTM Techniques. Complexity, 2021. https://doi.org/10.1155/2021/6647534

Khalid, S., Khalil, T., & Nasreen, S. (2014). A survey of feature selection and feature extraction techniques in machine learning. 2014 science and information conference, https://doi.org/10.1109/SAI.2014.6918213

Kou, G., Yang, P., Peng, Y., Xiao, F., Chen, Y., & Alsaadi, F. E. (2020). Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods. *Applied Soft Computing*, *86*, 105836. https://doi.org/10.1016/j.asoc.2019.105836

Lin, F.-L., Yang, S.-Y., Marsh, T., & Chen, Y.-F. (2018). Stock and bond return relations and stock market uncertainty: Evidence from wavelet analysis. *International Review of Economics & Finance*, *55*, 285-294. https://doi.org/10.1016/j.iref.2017.07.013

Modjtahedi, A., & Daneshvar, A. (2020). A New Credit Risk System Using Hybrid ELECTRE TRI and NSGA-II Methods. Journal of System Management, 6(4), 1-25. https://dx.doi.org/10.30495/jsm.2021.1924341.1445

Moghaddam, A. H., Moghaddam, M. H., & Esfandyari, M. (2016). Stock market index prediction using artificial neural network. *Journal of Economics, Finance and Administrative Science*, *21*(41), 89-93. https://doi.org/10.1016/j.jefas.2016.07.002

Nelson, D. M., Pereira, A. C., & de Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. 2017 International joint conference on neural networks (IJCNN), https://doi.org/10.1109/IJCNN.2017.7966019

Obthong, M., Tantisantiwong, N., Jeamwatthanachai, W., & Wills, G. (2020). A survey on machine learning for stock price prediction: algorithms and techniques. http://dx.doi.org/10.5220/0009340700630071

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. Expert Systems with Applications, 42(1), 259-268. https://doi.org/10.1016/j.eswa.2014.07.040

Qiu, J., Wang, B., & Zhou, C. (2020). Forecasting stock prices with long-short term memory neural network based on attention mechanism. *PloS one*, *15*(1), e0227222. https://doi.org/10.1371/journal.pone.0227222

Rather, A. M., Sastry, V., & Agarwal, A. (2017). Stock market prediction and Portfolio selection models: a survey. *Opsearch*, *54*(3), 558-579. https://doi.org/10.1007/s12597-016-0289-y

Sagheer, A., & Kotb, M. (2019). Unsupervised pre-training of a deep LSTM-based stacked autoencoder for multivariate time series forecasting problems. *Scientific reports*, *9*(1), 1-16. https://doi.org/10.1038/s41598-019-55320-6

Salmani Danglani, S., Saeedi, P., Bahramzadeh, H. A., & Pourshahabi, F. (2019). Representing the Pattern of Relationship between Personality Traits and Investment Patterns in the Stock Market. *Journal of System Management*, *5*(1), 79-114. https://dorl.net/dor/20.1001.1.23222301.2019.5.1.5.6

Selvamuthu, D., Kumar, V., & Mishra, A. (2019). Indian stock market prediction using artificial neural networks on tick data. *Financial Innovation*, *5*(1), 1-12. https://doi.org/10.1186/s40854-019-0131-7

Srivastava, A. K., Srivastava, A., Singh, S., Sugandha, S., & Gupta, S. (2022). Design of Machine-Learning Classifier for Stock Market Prediction. *SN Computer Science*, *3*(1), 1-11. https://doi.org/10.1007/s42979-021-00970-5

Taheri, A., Shafiee, M., & Evazzadeh Fath, F. (2019). Investigating the Role of Non-Financial Information Analysis and Risk-Return Analysis along with Financial Information in Increasing the Efficiency of the Stock Portfolio of Banks. *Journal of System Management*, *5*(3), 123-138. https://dorl.net/dor/20.1001.1.23222301.2019.5.3.8.3

Xu, Y., Chhim, L., Zheng, B., & Nojima, Y. (2020). Stacked deep learning structure with bidirectional long-short term memory for stock market prediction. International Conference on Neural Computing for Advanced Applications, https://doi.org/10.1007/978-981-15-7670-6_37

Yadav, A., Jha, C., & Sharan, A. (2020). Optimizing LSTM for time series prediction in Indian stock market. *Procedia Computer Science, 167*, 2091-2100. https://doi.org/10.1016/j.procs.2020.03.257

Yan, Y., & Yang, D. (2021). A stock trend forecast algorithm based on deep neural networks. Scientific Programming, 2021. https://doi.org/10.1155/2021/7510641

Yang, J., Zhang, S., Xiang, Y., Liu, J., Liu, J., Han, X., & Teng, F. (2020). LSTM auto-encoder based representative scenario generation method for hybrid hydro-PV power system. *IET Generation, Transmission & Distribution*, *14*(24), 5935-5943. https://doi.org/10.1049/iet-gtd.2020.0757

Yu, P., & Yan, X. (2020). Stock price prediction based on deep neural networks. *Neural Computing and Applications*, *32*(6), 1609-1628. https://doi.org/10.1007/s00521-019-04212-x

Zhong, X., & Enke, D. (2017). Forecasting daily stock market return using dimensionality reduction. Expert Systems with Applications, 67, 126-139. https://doi.org/10.1016/j.eswa.2016.09.027

Zhong, X., & Enke, D. (2019). Predicting the daily return direction of the stock market using hybrid machine learning algorithms. *Financial Innovation*, *5*(1), 1-20. https://doi.org/10.1186/s40854-019-0138-0