Multimedia Tools and Applications (2022) 81:14153–14171
https://doi.org/10.1007/s11042-022-12328-x

A stock market trading framework based on deep learning architectures

Atharva Shah[1] · Maharshi Gor[2] · Meet Sagar[3] · Manan Shah[4]

Received: 22 May 2021 / Revised: 14 January 2022 / Accepted: 18 January 2022 /
Published online: 25 February 2022

Abstract
Market prediction has been a key interest for professionals around the world. Numerous modern technologies have been applied in addition to statistical models over the years. Among the modern technologies, machine learning and in general artificial intelligence have been at the core of numerous market prediction models. Deep learning techniques in particular have been successful in modeling the market movements. It is seen that automatic feature extraction models and time series forecasting techniques have been investigated separately however a stacked framework with a variety of inputs is not explored in detail. In the present article, we suggest a framework based on a convolutional neural network (CNN) paired with long-short term memory (LSTM) to predict the closing price of the Nifty 50 stock market index. A CNN-LSTM framework extracts features from a rich feature set and applies time series modeling with a look-up period of 20 trading days to predict the movement of the next day. Feature sets include raw price data of target index as well as foreign indices, technical indicators, currency exchange rates, commodities price data which are all chosen by similarities and well-known trade setups across the industry. The model is able to capture the information based on these features to predict the target variable i.e. closing price with a mean absolute percentage error of 2.54% across 10 years of data. The suggested framework shows a huge improvement on return than the traditional buy and hold method.

Keywords Convolutional neural network (CNN) · Long short term memory · Deep learning architecture

✉ Manan Shah
manan.shah@spt.pdpu.ac.in

[1] Department of Mechanical Engineering, Nirma University, Ahmedabad, India

[2] Software Engineer at Quinbay Technology, Bangalore, India

[3] Data Science associate at ZS, Pune, India

[4] Department of Chemical Engineering, School of Technology, Pandit Deendayal Petroleum University, Gandhinagar, India

1 Introduction

Financial markets are the primary platform that allows investors and companies to deal with major economic transactions worldwide where billions of dollars' worth of financial instruments are traded daily [18]. Stock markets have a huge financial impact on the world economy [5] which was well evident by the market crashes in 2008 & 2020. Thus, it can be well established that predicting further movements becomes essential for optimum returns as well as financial security. Predicting the future of the market even seconds before can generate sizeable returns in comparison to the traditional buy and hold process. Previously, statistical models were used to predict the movements but research in Neural networks and in general AI has enabled researchers and industry to rely upon machines for trading securities fast and profitably [1]. With the advent of modern computers and high network transfer rates, algorithmic trading has become increasingly viable. Hedge funds like Renaissance trading incorporation have been using big data technology and hidden Markov models to generate returns much higher than the market benchmarks.

Stock markets come across as random when looked at from outside of the field however a large group of traders judge the movement through patterns and price levels on candlestick charts. The technique of analyzing the movements according to price actions and candlesticks is called technical analysis which indicates there might be an underlying pattern to the movements in markets. Statistical models and rule-based ML algorithms have been the basis of understanding macro-economic trends but have not been able to capture trends in a finer timeframe. There have been attempts by researchers to understand markets by using such ML models. Support vector machines (SVM) have been extensively used [19, 23, 41] by the researchers to take trading decisions by classification of the future price predictions. [23] have showcased that a simple artificial neural network (76%) had a better accuracy over an SVM kernel (72%). Logistic regression, K-nearest neighbors [37], random forests [28] have also been some of the traditional machine learning algorithms investigated by the researchers but the results have not been promising.

Researchers have also investigated the feature identification or extraction models (which is essentially what dimensionality reduction is) and their effects on standard machine learning algorithms' performance. The results showcased significant improvements. [14] have investigated the principal component analysis (PCA) dimensionality reduction algorithm to predict the movements of the stock market. PCA outperforms Gauss-Bayes and the moving average crossover method however, comparisons with deep learning models and other ML models are not present. In [16], it is seen that dimensionality reduction coupled with an LSTM network (68.5%) has performed slightly better than standalone time series forecasting with LSTM (67.5%) with a substantially smaller number of features. Evolutionary computing algorithms like genetic algorithm has been quite extensively used for feature extraction [3, 38]. [38] concludes that genetic algorithms perform better than the traditional buy and hold method but need a few signals to effectively model the dynamic and complex nature of the stock markets. Few researchers have coupled support vector machines with genetic algorithms to identify the best features among the inputs given to the SVM kernel [11] and it is noted that genetic algorithm coupled with SVM outperforms the standalone SVM model by 4%. From the results of the above articles, it seems that feature selection is crucial to the performance of the prediction system.

Deep learning is a particular branch of machine learning that mimics the human brain by creating individual neurons and can extract complex features from the data. Deep learning

offers a unique advantage in automatic feature extraction from raw data [26]. Artificial neural networks (ANN) have been used by companies like Tesla for automated driving assistance; Google, Apple, and others have been using forms of deep learning in the domain of natural language processing for their personal assistants. The deep learning models can identify faces and can sound signals to identify songs where it models complex features and transforms them into simpler variables. As the stock market data is also sequential like text generation used in chatbots and at the same time, noisy and complex like image data, deep learning proves to be quite promising. Simple multi-layer perceptron (MLP) models have been explored by researchers in the past. Several studies are present where various ANN architectures have outperformed conventional ML algorithms [4, 13, 25, 34, 42, 43]. Researchers have also used deep learning algorithms like restricted Boltzmann machines (RBM) [7], autoencoders (AE) [15]. All the above architectures have outperformed shallow ANN and other traditional ML algorithms. Deep learning architectures, especially CNNs have been quite successful in the prediction of stock markets due to their capacity to extract high-level features much more efficiently than shallow ANNs [21]. [12] have used CNN with using raw price data as a single feature while ignoring other technical indicators and have achieved a 53.6% accuracy on classification. The study lacks to incorporate a wider range of features in their ensemble models. [17] have built on the work of [12] and have incorporated technical indicators as well for the prediction of future prices. Their model is accurate up to 56.3% in classification but the returns on the trades are not documented. Also, they have not included various information-bearing sources and have their input space limited to only close prices and technical indicators. [21] have been able to incorporate a large variety of feature sets which include commodities, foreign markets, technical indicators and have built a classification model based on CNN. It is also noted that the technical indicators used are only limited to trend indicators while momentum, volatility, and strength indicators have been ignored. Also, they have not quantified the movement of prices and have limited the prediction to only the direction of the future movement. There is a lack of quantification of the future prices in all the 3 models discussed above. The quantification of a price is necessary for the traders and a trading system to judge whether the profit margin as per prediction will break even after the cost of brokerages. The model discussed in the present article aims to provide a quantification of the prices as an output.

Long short-term memory (LSTM) is another deep learning algorithm that captures temporal activity and therefore proves useful to model the time series behavior of stock markets [31]. However, [9] quotes *"Markets have very different statistical characteristics than natural phenomena such as weather patterns.*" which is why only time series modeling will not be fruitful as the past only looks for the pattern while not taking any other events into account. If a clear upwards trend exists for the training period, an RNN model will only be able to capture the variation and scaling in the price fluctuations whereas the co-dependencies amongst other sources of information will be completely ignored. A similar problem is encountered in [29] who have used a. RNN model and ARIMA model to predict limestone prices. There exists a clear upward sloping trend in the training data whereas the testing data has an erratic drop of prices (mostly due to coronavirus led market sell-off in 2020). The model fails to effectively capture the variation as a result. This tells us that there must be an additional source of information for the time series model to account for the external and noisy factors that govern the markets. The model discussed in the present article should be able to address the concerns mentioned above as the features can be extracted by convolutional layers while the seasonality and recurrences can be deciphered by the LSTM layers.

The present work is inspired by the combination of technical indicators as well as image processing. Technical candlestick patterns like bullish maribozu, three crows, evening star offers a unique insight into the market direction and suggest an underlying pattern. LSTM can capture the time series data effectively but the spatial sequence capturing through CNN that is pertinent to image processing, can be modified to capture temporal occurrences as well. Therefore, a CNN modified to fit a time series data can effectively process the recurrent occurrences that may not be visible conspicuously. CNN (for monthly predictions) alongside LSTM (for intraday predictions) [27] has been fruitful for predicting but still lacks to adjust to the scenarios mentioned above. That is why we propose a variety of inputs that have been gathered over from various sources. The proposed algorithm that is CNN stacked on top of LSTM is created to analyze daily trends of markets and take positional trades but can be modified to fit the high-frequency trading requirements as well.

2 Related works

Over the years, algorithmic trading has been the prime interest of many companies with their sole operation being trading securities. Along with the companies, market enthusiasts have also extensively researched the area of using machines to predict stock markets. As [21] suggest, researchers seem to adopt two classes of methodology when it comes to the use of deep learning algorithms in stock market predictions. The first class of papers focuses on feature extraction and using a variety of sophisticated algorithms to extract complex features from raw data. The other class focuses on improving prediction quality where the algorithm is finetuned to improve prediction accuracy.

For feature extraction, researchers have used a variety of algorithms such as CNN, Principal component analysis (PCA), ANN. Genetic algorithms have been used on top of ANN to find out the initial weights of the layers and errors are backpropagated to improve the quality of prediction. The other popular algorithms used are naïve Bayes, random forests, ARIMA (autoregressive integrated moving average) SVM but on multiple accounts [23, 36] it is found that ANN works better than classical ML models such as SVM to predict the non-linear and chaotic nature of stock prices.

[45] have used dimensionality reduction algorithms to extract better features from the Standard and Poor index. The features have then been provided as input to ANN for predicting intraday variation for stocks. PCA with ANN was found to be the best combination among other variations. The predictions were accurate to 58% but data loss was considerable. The paper however has not benchmarked their algorithm against other trading strategies purely based on technical analysis and algorithms other than dimensionality reduction algorithms. It can be seen that ML algorithms can't deconstruct the nonlinear and complex nature of stock markets.

A similar study was done by [36] where they have explored 4 different prediction algorithms namely SVM, naïve-Bayes classifier, random forests, ANN. The study is performed on the Indian stock market and its constituents. They have used 10 technical indicators as their inputs. Instead of modeling features as a continuous variable, they have rather mapped it into momentum space based upon its previous timestep values. The algorithms achieved considerable prediction accuracy but the study was meant to predict only the direction of the move while leaving out the quantification of prices. The study however showcases that mapping of technical indicators from continuous space to discrete space increased the prediction accuracy considerably.

[35] have explored an autoencoder-based dimensionality reduction and passed it on to LSTM neural network. Their area of focus was the Shanghai composite index where they found their version of the LSTM algorithm to be 58% accurate. The input vectors consisted of the raw historical price data, volumes, and a few other indicators which they have mentioned in detail in their study. The study incorporated web crawling to fetch the data and process it on the go which is the standard practice of modern sophisticated algorithmic trading software.

[39] have compared 3 architectures and benchmarked them to ARIMA, a linear time series forecasting model. Their area of focus is the Indian stock market and a few of its companies. The accuracy of CNN was found to be superior to those of other models. Their inputs considered a shorter period of 90 mins sliding overlap and the model would then predict 10 mins in the future. The sliding window approach seems to be in line with what LSTM does as it also processes last n time steps to predict the future. Similar results have been obtained by the authors in M et al., 2018 where several algorithms were applied to the NSE universe and it was found that CNN outperformed ARIMA.

Arevalo et al., 2016 [2] have used deep neural networks for predicting Apple stock using 5 layers. The method the authors in Arevalo et al., 2016 [2] employ to trade the equities is based on the difference in prediction to the actual value which can result in overtrading. The employed method is tested for a period of 3 months which might not be indicative of the performance for a longer time duration. The author is able to showcase the power of deep neural networks over shallow networks with deep networks reaching higher accuracies. The shallow networks cannot handle the data well which might lead them to give inaccurate predictions. With better gradient descent algorithms, researchers have been able to train deep neural networks quickly over large datasets.

There have been attempts to establish a correlation between sentiments of the retail investors and stock prices by the researchers. Nousi and Tjortjis, 2021 have used machine learning techniques to identify the sentiments in the social media posts and then use it to predict the market movements. A concrete relation is however not found which might be due to purchasing power of investing institutions over retailers. The works of [21] have used CNN in their deep network where they have used 82 inputs to predict 5 major U.S stock market indices. The inputs include returns for various stocks and commodities. 2D tensor input CNN performs well over the other methods it is benchmarked against. However, it must be noted that the stock market is an exchange platform that is governed ultimately by demand and supply. So, in addition to the trend indicators that authors have used, momentum, strength, and volatility indicators must be used to make decisions to ensure accuracy and safety against sudden volatility in opposite directions to our trade. A list of works with the feature extraction and prediction algorithms can be found in Table 1.

3 Model background

We have used CNN and LSTM layers on top of dense layers in the approach we showcase in this paper. So, we start by reviewing the concept of CNN and LSTM before we present our methodology.

Table 1 Comprehensive work on prediction algorithm based on feature extractions

References	Scope	Input features	Feature extraction	Prediction algorithm/s
[22]	Australian securities exchange	Price Data	Neural network	IOWA
[40]	NASDAQ	Price data, technical indicators	PCA	DNN
[44]	Nikkei 225 index	Price data	RBM	RNN-DBN
[30]	Tehran stock exchange	Technical indicators	Binary mapping	Machine learning and deep learning models
[33]	MSCI, United Kingdom	Price data	ANN	ANN, LSTM, RF, SVR
[10]	Korean stock index	Price data	AE, PCA, RBM	ANN, DNN, AR
Yong, Rahim and Abdullah	Singapore stock index	Price data	DNN	DNN
[24]	Indian stock market	Price data, Technical indicators	Scaled raw data	LSTM

3.1 Convolutional neural network (CNN)

Convolutional neural networks were introduced by LeCun and his colleagues through his paper in 1995 [26]. A convolutional neural network is mainly used to capture topology through feature extraction by applying filters to batches of data points. This can be used to capture the sequential as well as spatial data and therefore it is used extensively in image processing, speech recognition, time series analysis, etc. A convolutional neural network has an input layer followed by a convolutional layer, pooling layers, and output layers.

3.1.1 Convolutional layer

A convolutional layer is where the convolution operation is carried out. A convolution operation is when filters are applied to the data points in a neighborhood and the effect of that is passed on to the next layer. The filter is simply a matrix that is multiplied with the input matrix and mainly has 2 characteristics i.e., weights and shape. The weights are learned by the

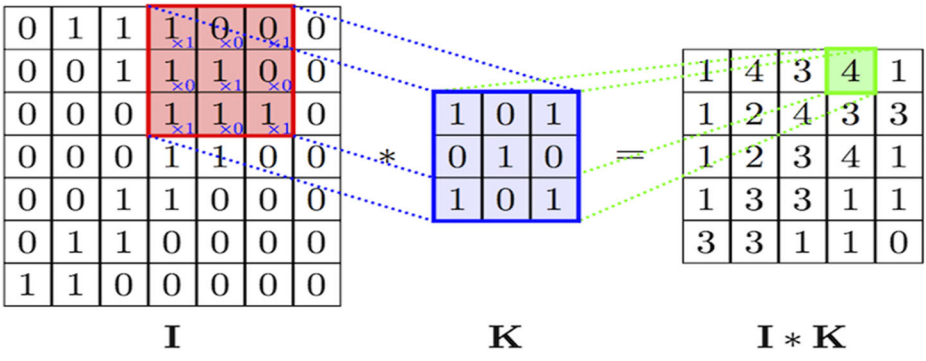

Fig. 1 Convolutional operation on input I with (3 × 3) filter K

model while it is being trained and the shape is the coverage of that filter. An example of convolution operation is seen in Fig. 1.

3.1.2 Pooling layer

The pooling layer is used to subsample the data. Pooling helps in reducing the dimensions and hence makes computation costs smaller. The pooling gathers the data after the convolutional layer outputs them and outputs the data according to the type of pooling chosen.

Pooling also addresses the overfitting problem which is quite prominent in CNN as a convolutional layer in the state-of-the-art VGG-16 model that has 512 filters. The number of parameters that are to be trained by the model is huge and hence is very much likely to overfit. Recent advances in CNN pruning have made the model space efficient but that is currently restricted to specific hardware only [8]. Pooling only outputs certain data and ignores the other as all the values inside the pooling window are reduced to a single value which makes the model less prone to overfitting. An example of a pooling operation is shown in Fig. 2.

3.1.3 Fully connected layer

The features extracted from CNN are given to dense layers or fully connected layers. The relation of input to the output of a perceptron in the Multi-layer perceptron (MLP) is

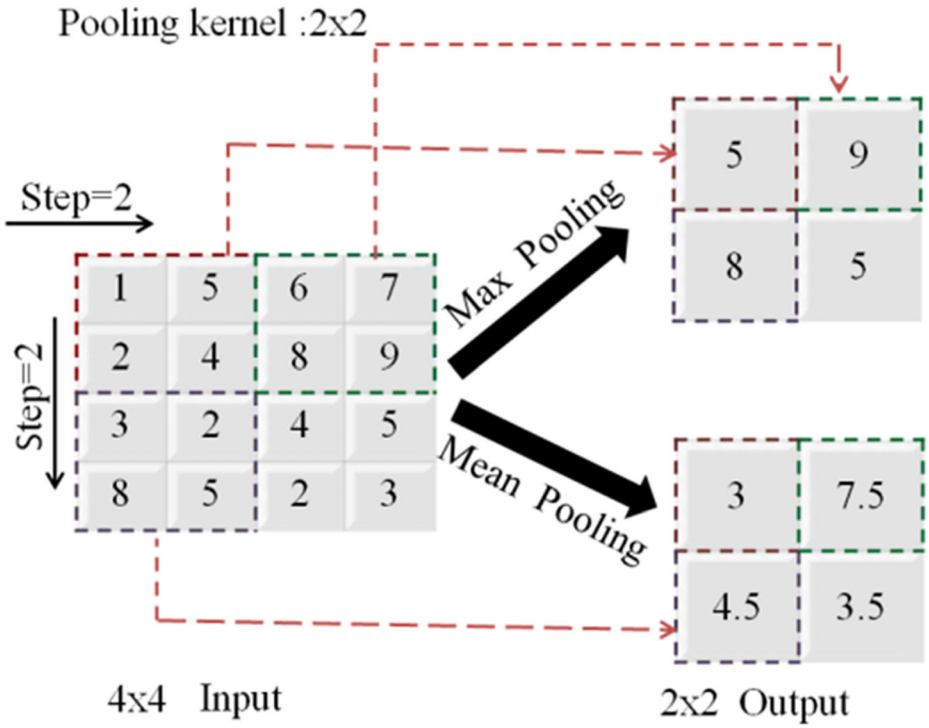

Fig. 2 Modes of the pooling operation

characterized by the equation

$$O_j = f\left(\sum_{k=1}^{n} i_{j-1}^{k} W_{j-1}^{k}\right) \quad (1)$$

In Eq. (1), O_j stands for the output of the j[th] perceptron in a neural net where function f is an activation function. The activation function receives input from the previous layer which is a summation of inputs of $(j\text{-}1)^{th}$ layer i multiplied with respective weight W.

3.2 Dropout

Dropout refers to the technique that is mostly used to address the overfitting problem in a neural network by making the fraction of neurons inactive which forces the model to rely on only a fraction of neurons to learn effectively. Dropout is characterized by 1 parameter which is the fraction of inactive neurons in a dense layer. As the neurons might be randomly inactive in each epoch, the model relies upon the others to train accurately and helps generalize the model (Srivastava et al., 2014).

3.3 Long short-term memory (LSTM)

LSTM is a type of recurrent neural network that has a feedback loop built in unlike other feed-forward networks. LSTM can remember the previous state and takes that into account while making a prediction. This helps in text generation where context is derived from the previous words and sentence structure and also in sequential data like time series forecasting. LSTM has 4 gates i.e., input gate, forget gate, modulation gate, and output gate which helps the model process the information from previous states and uses it with current input to get the next state.

3.4 Methodology

We propose a hybrid CNN stacked on top of LSTM and dense layers for predicting the future prices of the stock market index on NSE: Nifty50. Though optimized for Indian stock markets, it can be trained on other data as well. In image processing where CNN is broadly used, 3 × 3 and 5 × 5 filters are widely used where spatial variance is needed to be captured across a rectangular array of pixels. The inputs dimension of the kernel is modified to a size of 1 × 48 to accommodate the input feature set consisting of a total of 48 input features. Each input feature is represented by a series of variables such as close price, technical indicator values, commodities, etc. The inputs are fed to a CNN network which extracts the features on daily

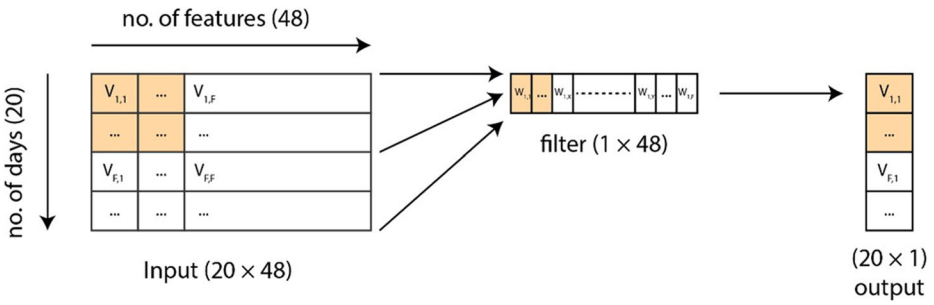

Fig. 3 Convolution operation on raw input features across 20 days

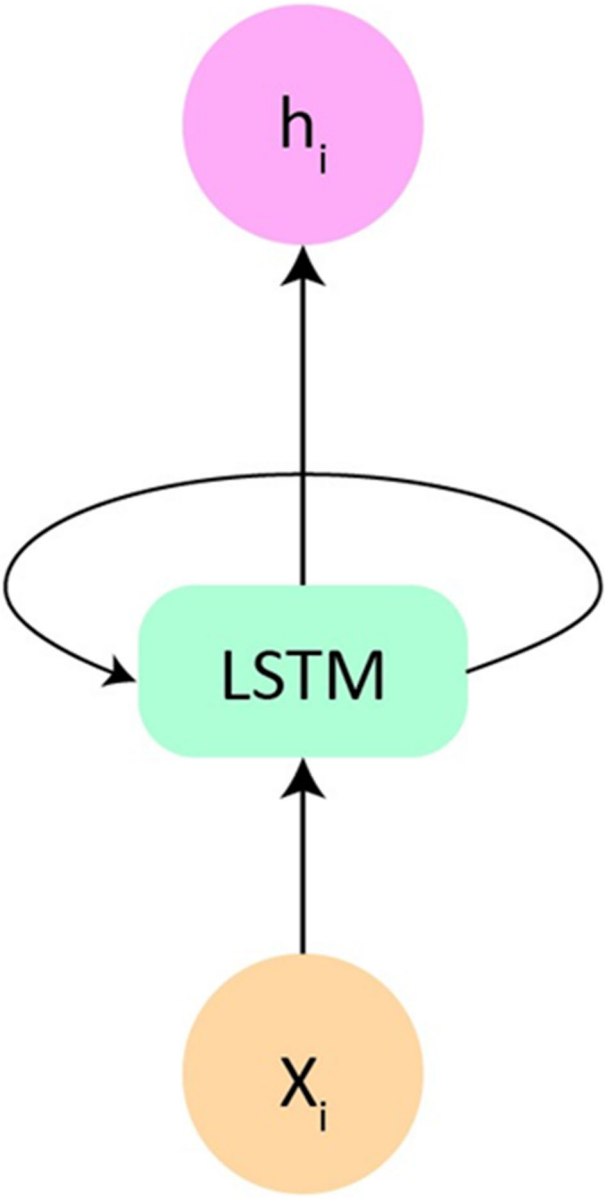

Fig. 4 LSTM visualization

price levels and outputs a tensor which is then passed on to the LSTM network. LSTM takes an input of previous 20-time steps and tries to predict the next time step of data.

3.5 Data collection and merging

The dataset includes the daily closing price of the Nifty 50 stock market index. In addition to this, the input variables consist of 32 technical indicators which can be classified into 4

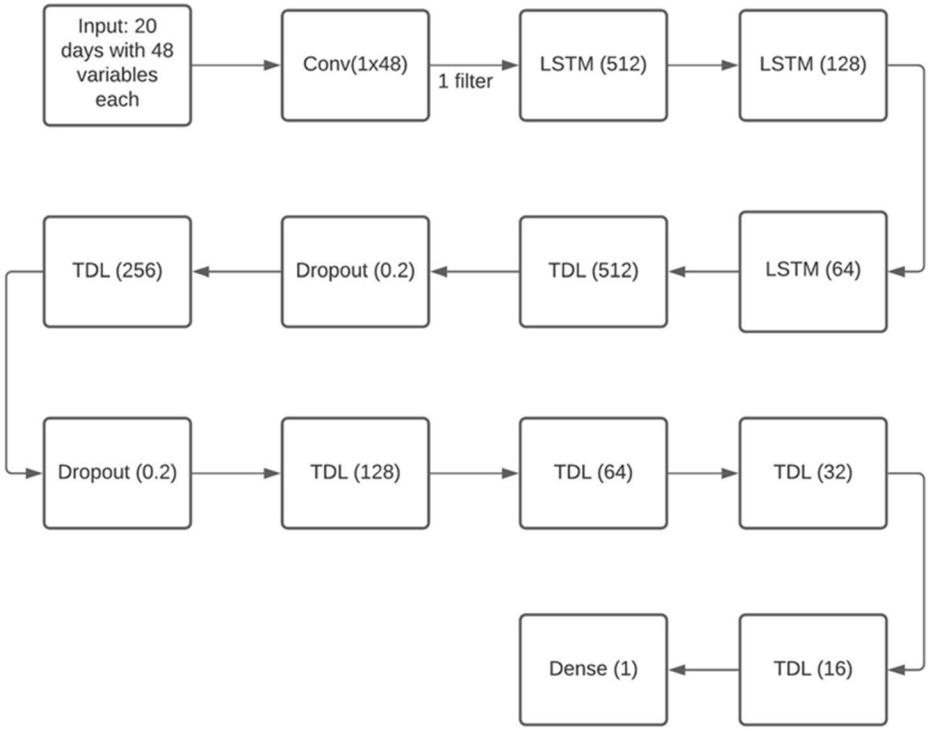

Fig. 5 Model architecture

categories i.e., strength indicator, momentum indicator, trend indicator, and volatility indicator. The other features are drawn by the similarity of the market movements and co-dependencies of the NIFTY50 index on other commodities and indices. Each sample consists of 48 variables that are continuous in nature. The sources from which the data is gathered are mentioned in Table 6. The data is not only bound to the local domain but gathered from domains across the world. The challenge in modeling around that data is the discontinuity that is present as markets of Shanghai are closed on Chinese holidays but the Indian markets had continued trading. The only way to tackle those discontinuities is to merge data on dates with a left join as nifty is our target dataset and use the backfill method on the missing values. In a total of 2286 entries of our dataset, there were 134 missing values for shanghai markets which was the highest number amongst all features. Neural networks unlike other rule-based models of ML are not sensitive to different scales of measurements and hence the data is fed in raw form and not normalized.

3.6 Model architecture

3.6.1 CNN

As we mentioned above, the input data is given onto a convolutional layer where 1×48 filters are used to capture daily price patterns where N represents the number of features. The convolutional neural network layer extracts the trends and the information which capture

Table 2 Parameter optimization setting values

Parameters	Values
Learning rate	0.0001, 0.0005, 0.001, **0.005**, 0.1
Activation function	ReLu, **Tanh**
LSTM layers	1,2,**3**,4
CNN layers	**1**,2,3
Number of neurons in the first layer of LSTM and TDL	128,256,**512**
Number of filters	**1**,2,4,8…32

and further the most of variance of the dependent variable. The convolution operation present in the model can be visualized in Fig. 3.

3.6.2 LSTM

The output data is not pooled to retain the intricate details as well as retain the time dimension of the input which is 20 days of lag data for our analysis. This is mainly because LSTM, a specific kind of RNN, performs better on time series analysis models as compared to other models. Further, the data is passed through multiple layers of LSTM. This enables the model to understand the basic trend of the market not just based on the variables used for the modeling for a single instance but also across consecutive days and incorporate the durational information. Figure 4 represents the LSTM architecture looping as a basic model element.

3.6.3 Time distributed layers

The output from the LSTM layer is later passed through the Time distributed layer. TDL is used to maintain the flexibility of multivariate modeling. Currently, for the discussed experimentations, the goal is to have the prediction based on Nifty, but keeping in mind the future experimentations, we would like to incorporate the Bank Nifty index as well. Hence for predicting the direction of Nifty and bank nifty from this single model time distributed layer can be used in the future as well without much of a change in the model architecture. Since this would add an additional dimension in the input data as well, 3D CNN would be used in place of the currently used 2D CNN as the initial layer for our model. This layer would do a similar task of feature extraction based on the input features used on an additional dimension as well for Bank Nifty. The complete model architecture is visualized in Fig. 5.

Table 3 Training error metrics

Metric	Value
Train R-Square	0.989
Train mean absolute error	168.558
Train mean absolute percentage error	0.0234
Train root mean square error	199.076

Table 4 Testing error metrics

Metric	Value
Test R-Square	0.943
Test mean absolute error	242.418
Test mean absolute percentage error	0.0310
Test root mean square error	413.902

3.7 Dataset and parameter optimization

70% of the dataset is used as a training set while the rest 30% is reserved as the test set and is completely unexposed to the model. Further over the test set, we have also reserved 30% of the training dataset as a validation split while model training that would help the model to adjust the weights in a more generalized manner and not overfit on the training period. The model is trained with minimizing "Mean absolute error" as a metric in focus. The model was trained iteratively by changing different parameters in the model. The learning rates, activation functions, number of CNN layers, number of filters, number of LSTM layers, number of neurons in each layer were experimented with to optimize the model architecture (Fig. 5). The bold values represent the used values that were found to be optimum for the given model. It must be noted that changing the number of CNN layers in the model increased the accuracy a little bit but not significant enough to justify the additional computing cost. Similar was the case with the number of filters as well. The parameter optimization settings can be seen in Table 2 and training and testing error metrics can be seen in Tables 3 and 4 respectively.

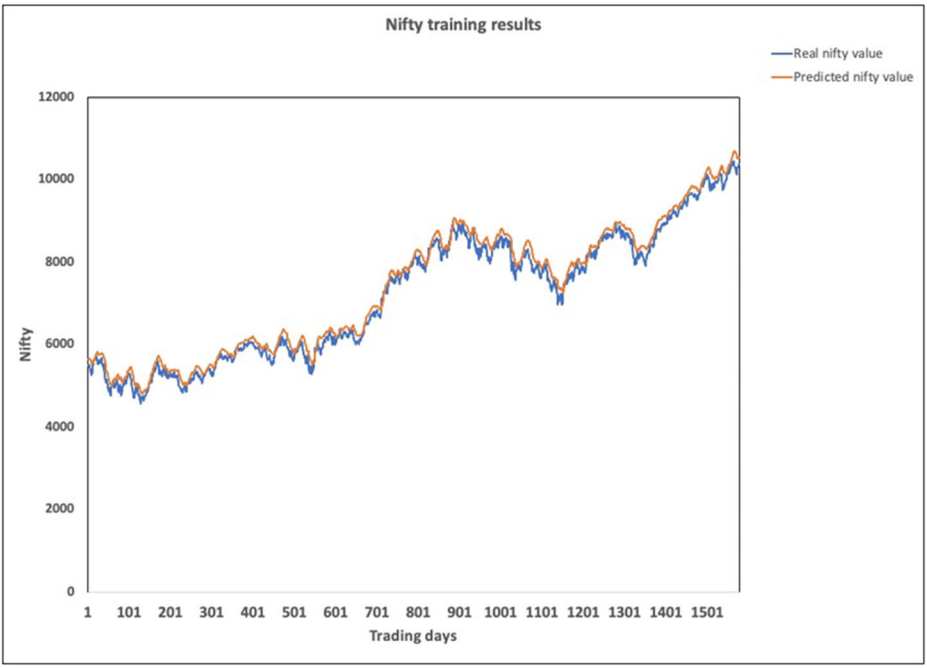

Fig. 6 Model performance on the training set

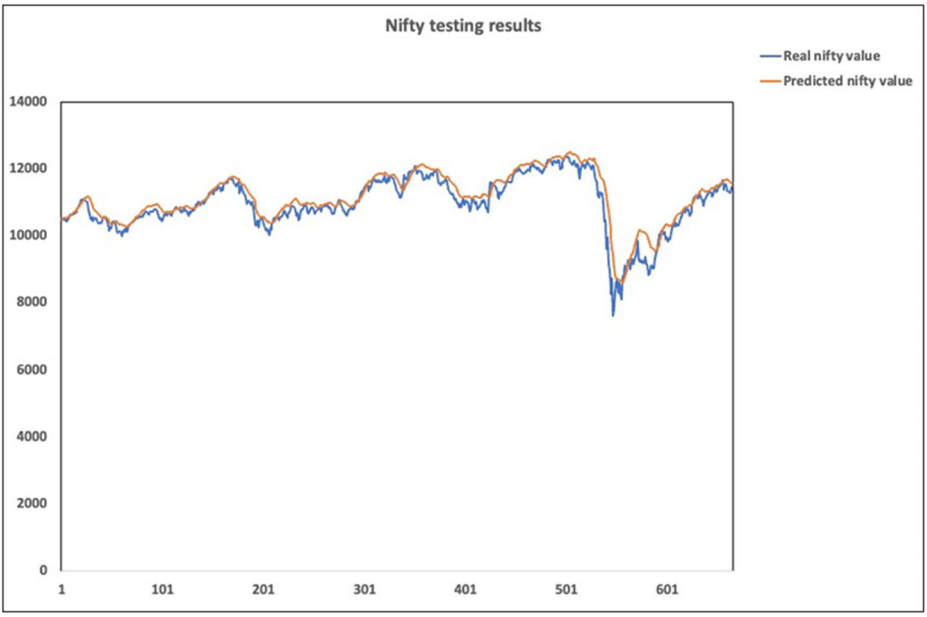

Fig. 7 Model performance on the testing set

3.8 Results and discussion

The model data is evaluated on the returns the model generates. Stop-losses are accounted into the trading philosophy as well which is quite a common practice among traders. The stop loss is set as the average move in percentage in the opposite direction of our trade taken. Stoploss is day's low to open when a long trade is suggested by model and day's high to open when short trade is suggested. Stoploss is employed to protect the capital against any sudden move which might not be anticipated by the model.

3.9 Trading methodology

The model suggests the next day's closing price and the trades are taken accordingly. If the closing price of the previous day is lower than the predicted value, a long trade is taken and vice versa. The stop loss is kept at −0.38% of the capital as it is just above the mean value of price movement in the direction opposite to our trade before moving in the predicted direction. That means, if at any given time, the trade makes a loss greater than 0.38%, the loss is booked for the day. When the model predicts the closing price of the next day is going to be higher, the model invests in that index and doesn't sell until the close price is predicted to be lower for the

Table 5 Metric of errors

	Hoseinzade, 2019		[23]		Present article	
Metrics	Train	Test	Train	Test	Train	Test
F1-score	0.046	0.16	0.009	0.07	0.572	0.578
Accuracy	0.473	0.496	0.382	0.366	0.537	0.531

Table 6 Data description and collection sources

	Variable	type	Source
1	Open	Price	NSE
2	High	Price	NSE
3	Low	Price	NSE
4	Close	Price	NSE
5	Adjusted Close	Price	NSE
6	Volume	Primitive	NSE
7	Trend	Technical indicator	TA-Lib
8	Wilder's RSI	Technical indicator	TA-Lib
9	RSI	Technical indicator	TA-Lib
10	RSI fast K	Technical indicator	TA-Lib
11	RSI fast D	Technical indicator	TA-Lib
12	Exponential RSI	Technical indicator	TA-Lib
13	MACD	Technical indicator	TA-Lib
14	MACD sign	Technical indicator	TA-Lib
15	MACD histogram	Technical indicator	TA-Lib
16	20 days MA	Moving average	TA-Lib
17	50 days MA	Moving average	TA-Lib
18	65 days MA	Moving average	TA-Lib
19	200 days MA	Moving average	TA-Lib
20	BB lower band	Technical indicator	TA-Lib
21	BB middle band	Technical indicator	TA-Lib
22	BB upper band	Technical indicator	TA-Lib
23	PPO	Technical indicator	TA-Lib
24	PPO sign	Technical indicator	TA-Lib
25	PPO hist	Technical indicator	TA-Lib
26	MOM	Technical indicator	TA-Lib
27	ROC	Technical indicator	TA-Lib
28	20 days EMA	Exponential moving average	TA-Lib
29	50 days EMA	Exponential moving average	TA-Lib
30	65 days EMA	Exponential moving average	TA-Lib
31	200 days EMA	Exponential moving average	TA-Lib
32	DPO	Technical indicator	TA-Lib
33	3 months bond yield	Dynamic returns	Investing.com
34	1-year bond yield	Dynamic returns	Investing.com
35	5 years bond yield	Dynamic returns	Investing.com
36	10 years bond yield	Dynamic returns	Investing.com
37	NASDAQ close	Price	Yahoo Finance
38	SSE close	Price	Yahoo Finance
39	Bovespa close	Price	Yahoo Finance
40	Gold	Commodity	Yahoo Finance
41	INR to CAD	Exchange rate	Yahoo Finance
42	INR to CNY	Exchange rate	Yahoo Finance
43	INR to SGD	Exchange rate	Yahoo Finance
44	INR to HKD	Exchange rate	Yahoo Finance
45	INR to AUD	Exchange rate	Yahoo Finance
46	INR to JPY	Exchange rate	Yahoo Finance
47	INR to USD	Exchange rate	Yahoo Finance
48	INR to EUR	Exchange rate	Yahoo Finance

next day. Upon selling, the model also creates a fresh short position and follows until closing price prediction is higher. Our model performance on the training set and test set can be seen below in Figs. 6 and 7 respectively.

While none of the models discussed so far have introduced a stoploss in their trading system, the need for capital protection is evident. Looking at the testing performance of the

Table 7 List of input features

Abbreviation	Full form
RSI	Relative strength index
MACD	Moving average convergence divergence
BB	Bollinger band
PPO	Percentage price oscillator
MOM	Momentum
ROC	Rate of change
DPO	Detrended price oscillator

model, it is seen that model falsely predicts a major up move between 550 to 600 days. The move was mostly due to a fiscal stimulus by the government but the model couldn't accommodate the information timely.

3.10 Baseline algorithms

As mentioned earlier, the article introduces a novel way of identifying the movement of stock markets. The model is optimized by various parameter settings as mentioned in Table 2. The model discussed in the present article is created to predict the closing price of the Nifty 50 index. The algorithms that use CNN architecture to predict the direction of the markets have categorical target variables while the model in this article has a continuous target variable. The models used in [21, 23] have been used to benchmark the model. To tackle the issue of prediction variables, our model predictions are also turned into categorical variables by the methodology discussed in the article of [21].

$$\text{Target variable} = \begin{cases} 1, C_{t+1} > C_t \\ 0, \ else. \end{cases}$$

C_t
Close price on t^{th}*day*

The models were trained with the optimization parameters given in the respective articles. The accuracy of our model was found to be superior. The metrics for train and test for mentioned papers can be found in Table 5.

The classification accuracy of our model that is meant to be a continuous variable prediction model is better than our baseline benchmark. The model has a better recall and precision which is evident by the f1 score metric presented above. In the case of [23], this is because technical indicators only provide the possibility of a possible reversal. The markets don't react to the technical indicators but it is the other way around. Prices move solely due to supply and demand which is created due to behavioral traits and reactions to a certain price point of the stock. As the price fluctuates, the technical indicators fluctuate with them. Hence a model which is relying on technical indicators maybe lagging due to the nature of input. [21] have included a richer variable set but the LSTM model that is essential for capturing the trend is absent in their model architecture. LSTM model looks at 20 days data and understands the pattern before predicting the output. As CNN looks at the numbers as a static image, the

outputs will be same if numbers are identical even in a bull trend and in a bear trend. LSTM model on top of a CNN model eliminates this issue and hence makes the model smarter. The addition of a sophisticated feature extraction model to a pattern recognition model with a rich and diverse feature set makes the model robust (Tables 6 and 7).

3.11 Returns

The model is also benchmarked against traditional buy and hold across 10 years of trading in the Nifty-50 index. The traditional buy and hold method generates a 107% return from 11th may 2011 to 11th September 2020 while our model with stop-losses generates 342% return in the same period. The model offers a far better return in comparison to the passive buy-and-hold strategy.

3.12 Challenges and future scope

The model we have presented is a predictive model which doesn't take into account economic events, policy changes, and overall perception of the investors which can affect the model [6]. The market discounts the information instantly while the model takes in that information at the end of the day in terms of prices and other features. This can be the scope of future work where the model learns to react to events and change its trades accordingly. All the above-discussed approaches work only when the market is deprived of any external noise. Noise may be a sudden surge of cash flows and liquidity or any other news. As the markets are not always rational and efficient, there cannot be a precise time when the stock is about to make move. Consumer behavior, unpredictable events like the pandemic can sway the market without any logic as it is upon the consumers who buy and sell equities. Hence it becomes essential that our model doesn't always rely on the raw prices and lagging features but also takes account of happenings and adjust accordingly. The limitation of the model imposed by this randomness requires us to not predict prices at every second but rather predict over a coarse daily timeframe.

Another challenge that the model might face is that the model is assuming a fixed sum of money present in the market but the markets are subject to constant injections and stimulus from foreign funds without any concrete reason for buying. The efficient market theory might fail in such cases and the model must take into account the cash flows at every point [32]. While major money injections can still be accounted for, cash flows from active interests from retailers and individual investors are very complex to model as they have no concrete data collection.

Another challenge that the model faces is data collection as the model has many input parameters and most of them have time zone differences or can be closed for trading on certain days. Even though while training the model, we have applied the backfill method to get rid of the missing values, this still would give an error in prediction in live trading. The future work calls for tackling above mentioned problems, demands further investigation in a reactive model on top of a predictive model and alternate data gathering models based on similarity of input features.

4 Conclusion

It is certainly a difficult task for creating a predictive analysis model on a noisy and non-linear dataset like the stock market. This study attempted to predict the closing prices for the next day of the Nifty 50 index with the help of deep learning architectures. A rich variable set which

includes technical indicators, price of commodities, price data of foreign markets along with raw price data of target market was used as a source of information to train a CNN-LSTM model for a continuous variable prediction. A 20-day data array was given along with 48 input features to CNN which extracted high-level features and fed them to an LSTM network. The model performed well over a large dataset with reasonable accuracies which suggests that the model could generalize well. The mean absolute percentage error in training was found to be 2.3% over 7 years of training data and the mean absolute percentage error in testing was 3.1% over 3 years of testing data. The suggested framework was also benchmarked against two similar studies and it was found to be superior. CNN-LSTM approach shows a huge gain on returns over the traditional buy and hold method which gave 107% return over 10 years while our approach provided 342% additional returns with stop-losses included. Quantification of the prediction values, which is the target variable of our model, can help traders identify the trade setup, especially with the options derivatives. A better prediction model can be obtained if more granular data is available but the inherent randomness of events across the world would still give errors to the model. The main purpose of this paper was to predict the movements of the market to maximize the returns by using deep learning methods. The model can be further converted to a sophisticated trading setup upon which a computer actively trades based on model predictions.

Acknowledgments The authors are grateful to the Department of Mechanical Engineering, Nirma University, Department of Chemical Engineering, School of Technology, Pandit Deendayal Petroleum University, and Quinbay Technology for the permission to publish this research.

Availability of data and material All relevant data and material are presented in the main paper.

Authors' contributions All the authors make a substantial contribution in writing the manuscript. AS, MG, MS, and MaS participated in drafting the manuscript. AS MG MS and MaS wrote the main manuscript, all the authors discussed the review and implication of the manuscript at all stages.

Declarations

Competing interests The authors declare that they have no competing interests.

Consent for publication Not applicable.

Ethics approval and consent to participate Not applicable.

References

1. Agarwal A (2021) Study of machine learning algorithms for potential stock trading strategy frameworks. International Journal of Financial, Accounting, and Management 3(3):275–287. https://doi.org/10.35912/ijfam.v3i3.604
2. Arévalo R, García J, Guijarro F, Peris A (2017) A dynamic trading rule based on filtered flag pattern recognition for stock market price forecasting. Expert Systems with Applications 81:177–92
3. Atsalakis GS, Valavanis KP (2009) Surveying stock market forecasting techniques - Part II: soft computing methods. Expert Systems with Applications 36(3 PART 2):5932–5941. https://doi.org/10.1016/j.eswa.2008.07.006

4. Avci E (2007) Forecasting daily and sessional returns of the ISE-100 index with neural network models. Journal of Dogus University 8(2):128–142
5. Beck, T., & Levine, R. (2004). Stock markets, banks, and growth: panel evidence. J Bank Financ, 28(3), 423–442. https://doi.org/10.1016/S0378-4266(02)00408-9
6. Birz G, Lott JR (2011) The effect of macroeconomic news on stock returns: new evidence from newspaper coverage. J Bank Financ 35(11):2791–2800. https://doi.org/10.1016/j.jbankfin.2011.03.006
7. Cai X, Hu S, Lin X (2012) Feature extraction using restricted Boltzmann machine for stock price prediction. In: CSAE 2012 - proceedings, 2012 IEEE international conference on computer science and automation engineering, 3, pp 80–83. https://doi.org/10.1109/CSAE.2012.6272913
8. Chen K, Franko K, Sang R (2021) Structured model pruning of convolutional networks on tensor processing units. CoRR, abs/2107.04191. Opgehaal van https://arxiv.org/abs/2107.04191
9. Chollet F (2017) Deep learning with Python. Simon and Schuster
10. Chong E, Han C, Park FC (2017) Deep learning networks for stock market analysis and prediction: methodology, data representations, and case studies. Expert Syst Appl 83:187–205. https://doi.org/10.1016/j.eswa.2017.04.030
11. Choudhry R, Garg K (2008) A hybrid machine learning system for stock market forecasting. July, 315–318
12. Di Persio L, Honchar O (2016) Artificial neural networks architectures for stock price prediction: comparisons and applications. International Journal of Circuits, Systems and Signal Processing 10:403–413
13. Egeli B, Ozturan M, Badur B (2003) Stock market prediction using artificial neural networks. In: Proceedings of the third Hawaii international conference on business, Honolulu, Hawai
14. Ghorbani M, Chong E (2018) Stock Price prediction using principle components. PLoS One 15. https://doi.org/10.1371/journal.pone.0230124
15. Gu S, Kelly B, Xiu D (2021) Autoencoder asset pricing models. J Econ 222(1):429–450. https://doi.org/10.1016/j.jeconom.2020.07.009
16. 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
17. Gunduz H, Yaslan Y, Cataltepe Z (2017) Intraday prediction of borsa Istanbul using convolutional neural networks and feature correlations. Knowl-Based Syst 137:138–148
18. Gupta A, Dhingra B (2012) Stock market prediction using Hidden Markov Models. Students Conference on Engineering and Systems 2012:1–4. https://doi.org/10.1109/SCES.2012.6199099
19. Guresen E, Kayakutlu G, Daim TU (2011) Using artificial neural network models in stock market index prediction. Expert Syst Appl 38(8):10389–10397. https://doi.org/10.1016/j.eswa.2011.02.068
20. Hiransha M, Gopalakrishnan EA, Menon VK, Soman KP (2018) NSE stock market prediction using deep-learning models. Procedia Computer Science 132(Iccids):1351–1362. https://doi.org/10.1016/j.procs.2018.05.050
21. Hoseinzade E, Haratizadeh S (2019) CNNpred: CNN-based stock market prediction using a diverse set of variables. Expert Syst Appl 129:273–285. https://doi.org/10.1016/j.eswa.2019.03.029
22. Hussain W, Merigó J, Raza M (2021) Predictive intelligence using ANFIS-induced OWAWA for complex stock market prediction. Int J Intell Syst. https://doi.org/10.1002/int.22732
23. Kara Y, AcarBoyacioglu M, Baykan ÖK (2011) Predicting direction of stock price index movement using artificial neural networks and support vector machines: the sample of the Istanbul stock exchange. Expert Syst Appl 38(5):5311–5319. https://doi.org/10.1016/j.eswa.2010.10.027
24. Khare K, Darekar O, Gupta P, Attar VZ (2017) Short term stock price prediction using deep learning. In: RTEICT 2017 - 2nd IEEE international conference on recent trends in electronics, information and communication technology, proceedings, 2018-January, pp 482–486. https://doi.org/10.1109/RTEICT.2017.8256643
25. Kimoto T, Asakawa K, Yoda M, Takeoka M (1990) Stock market prediction system with modular neural networks. In: Proceedings of the international joint conference on neural networks, San Diego, California, pp 1–6
26. Lecun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444. https://doi.org/10.1038/nature14539
27. Liu, S., Zhang, C., & Ma, J. (2017). CNN-LSTM neural network model for quantitative strategy analysis in stock markets. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 10635 LNCS, 198–206.
28. Manojlović T, Štajduhar I (2015) Predicting stock market trends using random forests: a sample of the Zagreb stock exchange. In: 2015 38th international convention on information and communication technology, electronics and microelectronics, MIPRO 2015 - proceedings, may, pp 1189–1193. https://doi.org/10.1109/MIPRO.2015.7160456
29. Mbah TJ, Ye H, Zhang J, Long M (2021) Using LSTM and ARIMA to simulate and predict limestone Price variations. Mining, Metallurgy & Exploration 38(2):913–926. https://doi.org/10.1007/s42461-020-00362-y

30. Nabipour M, Nayyeri P, Jabani H, Shahab S, Mosavi A (2020) Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis. IEEE Access 8:150199–150212. https://doi.org/10.1109/ACCESS.2020.3015966
31. Nelson DMQ, Pereira ACM, De Oliveira RA (2010) 2011 international joint conference on neural networks. IEEE Trans Neural Netw 21(8):1378–1378. https://doi.org/10.1109/tnn.2010.2063350
32. Neri S (2002) Monetary policy and stock prices : theory and evidence*. March
33. Nikou M, Mansourfar G, Bagherzadeh J (2019) Stock price prediction using DEEP learning algorithm and its comparison with machine learning algorithms. Intelligent Systems in Accounting, Finance and Management 26(4):164–174. https://doi.org/10.1002/isaf.1459
34. Olson D, Mossman C (2003) Neural network forecasts of Canadian stock returns using accounting ratios. Int J Forecast 19(3):453–465
35. Pang X, Zhou Y, Wang P, Lin W, Chang V (2020) An innovative neural network approach for stock market prediction. J Supercomput 76(3):2098–2118. https://doi.org/10.1007/s11227-017-2228-y
36. 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 Syst Appl 42(1):259–268. https://doi.org/10.1016/j.eswa.2014.07.040
37. Qian B, Rasheed K (2007) Stock market prediction with multiple classifiers. Appl Intell 26(1):25–33. https://doi.org/10.1007/s10489-006-0001-7
38. Radeerom M (2014) Building a Trade System by Genetic Algorithm and Technical Analysis for Thai Stock Index. In: Nguyen NT, Attachoo B, Trawiński B, Somboonviwat K (eds) Intelligent Information and Database Systems. ACIIDS 2014. Lecture notes in computer science, vol 8398. Springer, Cham. https://doi.org/10.1007/978-3-319-05458-2_43
39. Selvin S, Vinayakumar R, Gopalakrishnan EA, Menon VK, Soman KP (2017) Stock price prediction using LSTM, RNN and CNN-sliding window model. In: 2017 international conference on advances in computing, communications and informatics, ICACCI 2017, 2017-January, pp 1643–1647. https://doi.org/10.1109/ICACCI.2017.8126078
40. Singh R, Srivastava S (2017) Stock prediction using deep learning. Multimed Tools Appl 76(18):18569–18584. https://doi.org/10.1007/s11042-016-4159-7
41. Wang J, Wang J (2015) Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks. Neurocomputing 156:68–78. https://doi.org/10.1016/j.neucom.2014.12.084
42. White H (1988) Economic prediction using neural networks: the case of IBM daily stock returns. In: Proceedings of the IEEE international conference on neural networks, San Diego, CA, USA, pp 451–458
43. Yoon Y, Swales G (1991) Predicting stock price performance: a neural network approach. In: Proceedings of the 24th annual Hawaii international conference on systems sciences, Hawaii, pp 156–162
44. Yoshihara A, Fujikawa K, Seki K, Uehara K (2014) Predicting stock market trends by recurrent deep neural networks. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol 8862, pp 759–769. https://doi.org/10.1007/978-3-319-13560-1
45. Zhong X, Enke D (2017) Forecasting daily stock market return using dimensionality reduction. Expert Syst Appl 67:126–139. https://doi.org/10.1016/j.eswa.2016.09.027
46. Nousi C, Tjortjis C (2021) A Methodology for Stock Movement Prediction Using Sentiment Analysis on Twitter and StockTwits Data, 2021 6th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM), pp. 1–7. https://doi.org/10.1109/SEEDA-CECNSM53056.2021.9566242