EAI Endorsed Transactions
on Intelligent Systems and Machine Learning Applications Research Article EAI.EU

Deep Learning Approaches for Stock Price Prediction: A Comparative Study on Nifty 50 Dataset

Sushma P Kallimath[1], Narayana Darapaneni[2], Anwesh Reddy Paduri[3,*]

[1]PES University, Bangalore, India, 560085
[2]NorthWestern University, Evanston, IL, USA 60208
[3]Great Learning, Hyderabad, India 500089

Abstract

Stock price prediction is essential for investors and traders in financial markets. Deep learning methods have emerged as promising tools for capturing intricate patterns in stock market data. In this paper, we explore a comprehensive comparative study of various deep learning architectures for stock price prediction using the Nifty 50 dataset. The models evaluated include Linear regression, LSTM, GRU, CNN, RNN, Temporal Convolutional Network (TCN), as well as combination models such as LSTM+GRU, CNN+RNN, CNN+TCN, and LSTM+TCN.

Our study aims to evaluate how well they perform and suitability of these methodologies in capturing the dynamics of stock price movements. Utilizing historical Nifty 50 data spanning multiple years, we evaluate the models' predictive capabilities using standard evaluation metrics such as MSE, R2 Score, RMSE, MAE, and MAPE.Results from our experiments unveil distinct strengths and weaknesses among the different deep learning architectures. While linear regression provides a baseline for comparison, deep learning models like LSTM, GRU, CNN, RNN, and TCN exhibit superior performance in capturing the nonlinear and time-varying nature of stock market data. Additionally, hybrid architectures demonstrate promising results by leveraging the complementary strengths of individual models.

This comparative study offers meaningful perspectives on the effectiveness of various deep learning approaches for stock price prediction, which can benefit researchers, practitioners, and stakeholders in the financial domain. By understanding the performance characteristics of these models, stakeholders can make informed decisions in their investment strategies.

Received on 05 October 2024; accepted on 27 February 2025; published on 28 February 2025

Keywords: NSE50, regression models, R2 score, RMSE, MAPE, MAE

doi:10.4108/eetismla.7481

1. Introduction

In finance Market, accurately predicting stock prices is an enduring goal for both investors and traders. The volatility and complexity of stock market data pose significant challenges, requiring sophisticated analytical tools to uncover meaningful patterns and trends. Deep learning, a subset of machine learning techniques, has garnered considerable attention for its ability to extract intricate features and relationships from complex datasets. In particular, deep learning models such as Linear regression, Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Temporal Convolutional Network (TCN), as well as hybrid architectures like LSTM+GRU, CNN+RNN, CNN+TCN, and LSTM+TCN, have shown promise in capturing the dynamics of stock price movements.

The Nifty 50 index mirrors the combined performance of the leading 50 companies listed on the National Stock Exchange of India (NSE), serves as a

*Corresponding author. Email: anwesh@greatlearning.in

benchmark for Indian equity markets. Analyzing the Nifty 50 dataset gives valuable insights into the broader Indian stock industry shifts and is of significant interest to investors and analysts alike.

This paper delves into an extensive comparative examination of deep learning methodologies for stock price prediction using the Nifty 50 dataset. We aim to evaluate the efficacy of various deep learning architectures in capturing the complexities inherent in stock market data and to provide insights into their performance characteristics.

The models under investigation encompass a spectrum of deep learning techniques tailored for sequential data analysis. Linear regression, a conventional statistical method, serves as a baseline for comparison, while LSTM, GRU, CNN, RNN, and TCN represent state-of-the-art deep learning models. Additionally, hybrid architectures combining different models, such as LSTM+GRU, CNN+RNN, CNN+TCN, and LSTM+TCN, are explored to harness the complementary strengths of individual architectures.

To gauge how well these models perform, we utilize commonly used evaluation criteria, including MSE, R2 Score, RMSE, MAE, and MAPE. These metrics provide quantitative measures of predictive accuracy, making room for a comprehensive evaluating how well the models fare in comparison across different architectures.

Through this comparative study, we strive to enrich the collective knowledge on predicting stock prices and enhance our comprehension of the strengths and weaknesses of deep learning methods in financial forecasting. Our discoveries hold relevance for investors, financial experts, and scholars, providing valuable perspectives on the effectiveness of different deep learning techniques for forecasting stock market trends.

2. Related Work

Structured Learning and Market Trend Prediction (CKRM) algorithm engages in a process of merging an image of the candlestick chart pattern with regression analysis to predict and recognize the market trends. CKRM uses historical price data and regression methods to identify patterns, Anticipate market movements and execute trading decisions guided by regression outcomes. This approach enables the investors to understand the market sentiment, make risk decisions and create strategies that may be based on the signals observed from the candlesticks patterns and the statistical modelling. One of the main functions of CKRM is figuring out and forecasting the possible tendencies of the market, i.e., bullish, bearish or sideways through the provision of the crucial information that helps traders analyse and take necessary actions. CKRM integrates technical analysis with the quantitative techniques that offer a complete approach for stock market analysis. Hence, the investors will be able to address the market dynamics and seize this chance.[1]

The Stock Market Prediction based on supervised machine learning algorithms makes use of the past stock data to train and generate future predictions which allow for gaining valuable insights. Various machine learning approaches including regression or classification can be used to train models to find patterns and relationships within the market data which eventually allows to predict market movements and find trading opportunities. Analyzing variables like the price history, trading volume as well as market sentiments, the models can give relevant data to investors, traders and financial analysts. The method allows to improve decision-making process through implementation of the capabilities of machine learning to process market trends and to adapt investment strategies so as to achieve high return level with the lowest risk.[2]

Convolutional Neural Networks (CNNs) are used in concert with BiLSTM networks, and the Attention Mechanism (AM) for stock price prediction are constituted by the CNN-BiLSTM-AM method. CNNs allow for the capturing of spatial features in stock price data, while BiLSTM deal with the handling of the temporal aspects. The attention mechanism highlights crucial time steps, which contribute to the model interpretability favorably. Therefore, this type of structure Enables the model to capture the subtle nuances and correlations within stock price data over time, thereby enhancing the likelihood of making accurate forecasts. Through the use of the spatial and temporal details in the nature of CNN-BiLSTM-AM along with the attention based mechanism, a firm foundation for the prediction of stock prices can be created that will account for both the short-term fluctuations and the long-term trend.[3]

Stock price prediction involving deep neural networks models utilizes structures such as RNN, LSTM and CNN to analyze historical stock data and forecast future prices. These models work with the sequential nature of stock prices and find such complex patterns from masses of data. Through looking into the past trends and market signals, deep neural networks help buyers as well as investors to take decision which are based on facts. They provide the relevant information that helps managers navigate risks and make investment decisions. Nevertheless, deep neural networks are still endowed with compelling instruments for predicting stock market prices where the accuracy is not far off the mark.[4]

Predicting share prices using news sentiment analysis involves using NLP techniques with machine learning models which are aimed at extrapolating market

sentiment from textual news data. Through the process of sentiment analysis, the data extracted from the news articles, social media and financial reports is analyzed to obtain the market sentiments, the investor emotions and the public opinion about specific stocks or companies. Sentiments, detected from textual data, are then used as features to make predictive models like regression, decision trees or neural networks. These models apply sentiment scores and previous data for stock prices in order to forecast future price movements. This methodology aims to address market mood oscillations, news-relevant events, and investors' sentiment trends, that significantly increase prediction precision of stock prices. Through news sentiment analysis application which traders and investors gain valuable information on market behaviors and consequently they are able to make more authentic decisions and reduce risks.[5]

The forecasting stock price in the Indian stock market by using LSTM networks relies on tactfully using the sequential nature of the stock price data. LSTM networks, a kind of RNN, learn temporal effects and features in time-series data for stock prices. These LSTM models produce forecasts by analyzing past price fluctuations and taking into account market trends, thus being able to make predictions for future stock prices with some degree of accuracy. This approach encompasses the analysis of past trading information as well as the application of technical indicators and market sentiment to give intricate insights to the Indian stock market investors and traders. LSTM-based forecasts support data-informed decision making, risk mechanism, and best investment strategies in unpredictable market environments.[6]

By Using Deep Learning and Natural Language Processing (NLP) as components of the model, a Robust Predictor for Stock Price Prediction is able to investigate the price trends of shares by analyzing both numerical and textual information. By virtue of Deep Neural Networks, this model fulfills its task through LSTM and CNNs and it captures temporal patterns and spatial features of stock price respectively. NLP pipeline ranging over news articles, social media and financial reports parses them and groups the sentiment and market related data. Through the combination of multiple cognitive factors, the model gives the trading more robust and efficiency to investors and traders. This combination of deep learning and NLP enforces all market participants with the full features of the market environment that contribute greatly in the unlocking of more effective investment decisions.[7]

The approach for predicting stock prices using Convolutional Neural Networks focuses on the implementation of CNNs for price prediction. The CNNs, which excel in the field of image analysis, interpret price and volume data as seemingly two-dimensional images, thus visualizing spatial patterns and temporal correlations. The method is based upon the conversion of historical price data into a data structure format, which is then fed into CNNs for feature extraction. On top of the hierarchical nature of CNNs, the model is also able to discover and identify complex patterns in stocks prices, thus making its predictions accurate. This researches bring forth deep learning applicability in financial forecasting and develops more robust models for stock market predictions as well as produce investors and financial analysts.[8]

The Study on Stock Price Prediction using the LSTM's Associated Network Model reports on the employment of LSTM networks for stock price prediction. LSTM, which is a kind of RNN, detects long-term dependencies and sequential patterns through the stock market data stream. The incorporated network model takes into account interrelations among various stocks or market indices, hence improving the accuracy of prediction. While incorporating the LSTM's ability to glean insights from past data and identify complex relationships, the study intends to achieve better accuracy in forecasting stock prices. This study contributes to the development of forecasting methods in financial markets and provides key information to investors, traders, and financial researchers.[9]

Stock Prediction with a Modified Firefly Algorithm and Support Vector Regression (SVM-MFA) suggests the integration of SVR with a modified version of the firefly algorithm to predict stock prices. The learning SVR is a supervised method of analysis that efficiently detects nonlinear relationships between features and the target variable via historical data. Modified Firefly Algorithm emerges as a better optimization technique for SVR hyperparameters investigation that result in improved prediction accuracy. This model can handle complex tasks such as searching and exploiting of search space efficiently and then provides effective forecasts in stock markets. This research adds to the improvement of forecasting models that help investors and traders to make concrete decisions in favor of their interest and to minimize the risks in an unstable market.[10]

The Predictions on Stock Prices using the Combined Model of ARI-MA-LS-SVM is the result of a model that utilizes Autoregressive Integrated Moving Average (ARI), Moving Average (MA), and Locally Stationary Support Vector Machines (LS-SVM) techniques. ARI, MA and LS-SVM approach the time sequence, look at historical data and deal with the nonlinear relationships in stock data at the same time. The layer model merges these algorithms to get ARI for analysis of time series, MA for trend detection, and LS-SVM for precise forecasting. It can provide broad forecasts through multifactored approach which improve traders and investors choices. This work is useful for enhancing

forecasting techniques, pointing at market tendencies, and redirecting risk management methods in the stock market.[11]

The Hybrid Model for Predicting Stock Prices using Variational Mode Decomposition (VMD) and LSTM Network aims to combine the advantage of VMD signal decomposition with LSTM sequential learning, hoping for the development of stock price prediction. VMD transforms stock price data into intrinsic modes, thereby capturing prime patterns in the system, and LSTM discovers time-lagged dependencies among implotted historical sequences. The hybrid model combine both methods into one to increase the forecasting precision level, which brings the ability to highly capture whatever trends or variability happens in stock prices. Through the integration of the deep signal processing and the deep learning methodologies, the model improves its forecasting accuracy to enable investors and analysts with the useful market dynamics data to help them with the right stock market-related decisions.[12]

Deep Learning-Based Stock Price Prediction employs LSTM and Bi-Directional LSTM models as part of its forecasting processes for stock prices. LSTM is the one that captures the sequential relationship and dependences from the historical data of stocks and Bi-Directional LSTM brings more information involving past to future contexts. They scrutinize the large data sets and uncover the complicated parameters, which help to develop these models to give precise forecasts of future stock quotes. Through the combination of LSTM and Bi-Directional LSTM techiques, it is possible to build a forecasting approach that is reliable enough and supplies investors and traders with exciting information about market tendencies and dynamics. This method helps to develop informed decision-making and risk management in the end, resulting in an effective stock marketplace.[13]

Forecasting stock prices using more sophisticated Integrated Artificial Neural Network (ANN) and metaheuristic algorithms differ in distinct fashion compared with traditional time series models. The method includes use of ANN's nonlinear relationship identification features as well as metaheuristic optimization algorithms. This integration serves to grow forecasting precision by determining the causalities in the stock data. Different from time series models which may have problems with irregular and complex variations in market activities, the combined approach gives place to changing economic conditions. Through employment of the ANN features such as flexibility and metaheuristic algorithms' adaptability, the method is robustly guaranteed. This study offers new ways of forecasting, which would be a huge benefit for the investors and analysts who would be able to make informed decision and reduce risk in the volatile stock markets.[14]

A Forecasting Model for Stock Market Volatility, blending Time Series Data with a Combination of ARIMA and XGBoost is a combination of ARIMA and XGBoost algorithms which predict the stock market volatility. Capturing linear trends and Identifying Seasonal Patterns in Time Series Data using ARIMA Models, the Model is empowered with high accuracy XGBoost feature by gradient boosting. Combining the 2 models can allow the approach to not only capture the complex patterns but also the nonlinear relationships between the data of volatility which increases the method forecasts precision. By combining both fundamentals and technical analysis, investors and analysts gain key insights into the market dynamics and ultimately take the informed decisions that enable them to manage their risks under varying market conditions.[15]

Stocks Closing Price Prediction Adoption of Sentiment Analysis and LSTM models to anticipate the movement of stock prices. Sentiment analysis involves Analysis of sentiments from textual data such as news articles and social networks whereas LSTM networks are characterized by learning sequential patterns from historical stock data. Pairing sentiment data with the past movements in price will enable the model to forecast closing prices with elevated precision. This fusion method of data analysis includes textual and numerical data. It is able to give market dynamics a more comprehensive and detail-oriented coverage, which is beneficial to investors because they get a great amount of information which can be used for better decision-making as well as risk management. Sentiment analysis and LSTM contribute effectively to a more precise and comprehensive prediction of stock price.[16]

We blend Wavelet Transform and LSTM algos to increase the accuracy of stock price prediction with Digital Transformation. By Wavelet Transform, raw data is preprocessed and breaks the data into different frequency components, which separately reflect both short-term and long-term aspects of a given term. LSTM, with its special ability to handle sequential learning, is subsequently imbibed with these features to make stock price predictions for the future. Therefore, that combination gives an up-to-date method of stock market dynamics which is a useful instrument by which investors can take wiser decisions. Using these two methods together, the model reaches levels of precision in accurately detecting all the complex patterns and diurnal patterns that drive the stock price move.[17]

The Data Science Approach entails using LSTMs networks in Indonesian stock prices predicting during the Covid-19 pandamic. LSTM, a type of RNN, is efficient in preserving the order (temporal dependency) and distinguishing sequence patterns. Through portraying historical stock market data as well as the effect of

the COVID-10 pandemic, LSTM models learn complex relationships and make precise forecasts of stock prices in the future. This information-driven methodology employs advanced deep learning techniques and data science methods, enabling financial agents like investors, traders, and financial analysts to get insights into stock market dynamics in Indonesia, especially during the pandemic, which in turn helps them to rationally take decisions and mitigate risks.[18]

The paper presents an innovative Hybrid Deep Learning Model aimed at Forecasting Stock Price Movement which is an innovation that reveals new knowledge. The model has also been enhanced by incorporating various deep learning architectures to effectively process complex patterns present in stock market data. The model accounts for the combination of CNNs, RNNs, and other techniques, thus, improving the prediction accuracy. This approach mixes theoretical element among dynamic market conditions and also takes learnings from historical trends which give the forecasts powerful estimate for stock price movement. So, the research brings innovation in forecasting techniques; it can also help investors and analysts make better decisions as in stock market, it plays an important role in risk management. This hybrid model gives assurance that the model can provide accurate predictions concerning the dynamic market situation.[19]

The CSE Stock Market Price Forecasting study contrasts ARIMA and Artificial Neural Network (ANN) for Colombo Stock Exchange (CSE). ARIMA, a conventional time series method, and ANN, a machine learning technique, are being considered to find out their relative forecasting accuracy. The research entails backtesting the historical stock data from the CSE to establish the robustness of every model in predicting the future of stock prices. The accuracy and robustness of the ARIMA and ANN are analyzed in this research. It provides an insight into their applicability in stock market forecasting. The above comparative analysis helps to investors and analysts in choosing wittingly the best method to make a better decision in the CSE.[20]

The Innovative CNN-LSTM with Leading Indicators augmented with Graph-based techniques, is meant to fill the existing gap in the Stock Price Prediction domain. Within this combination, the leading indicators allow to enhance forecast to make it more accurate. CNN takes cues from stationgraphs to identify spatial features, while LSTM picks up on the temporal relationships. Through incorporating predictive markers, the algorithm gives comprehensive information on market relationship and dynamics. The graph-based architecture allows for the development of more complicated links between various stocks. Real-time analysis, adaptive learning, and artificial intelligence integration are paving the way for more accurate forecasts enabling investors and analysts to have credible decision-making and risk management tools that can be used effectively in fluent stock markets.[21]

Financial Sentiment Index and Stock Return Predictability method using of BERT, a recently advanced natural language processing model that has achieved cutting-edge performance, with LSTM networks for stock return prediction. BERT develops a financial sentiment analysis from textual data by creating a sentiment index. LSTM models model dependencies in stock data and learn temporal relations. Sentiment analysis, coupled with stock return forecasting, offers the method a greater predictability. It uses the deep learning and natural language procession techniques to deliver investors and analysts with a very important information which helps them in taking informed decisions and risk management.[22]

The research presents an Innovative Enhanced Particle Swarm Optimization (PSO) combined with LSTM Hybrid Model for Predicting Stock Market Indices. This hybrid approach merges PSO optimization techniques with LSTM networks to enhance stock index prediction accuracy. PSO optimizes LSTM parameters, facilitating improved model performance. By integrating the strengths of PSO and LSTM, the model effectively captures complex patterns and trends in stock index data. This innovative methodology offers robust forecasting capabilities, empowering investors and analysts with valuable insights into market dynamics. The hybrid model contributes to advancing forecasting methodologies, aiding in more informed decision-making and risk management strategies in financial markets.[23]

Utilizing a blend of Soft Computing Models, the Stock Price Prediction system incorporates fine-tuning of parameters and selection of input variables to improve forecast precision. This approach combines various soft computing techniques, such as ANN, SVM, and fuzzy logic systems, to analyze historical stock data. Through meticulous parameter tuning and careful selection of input variables, the model effectively captures intricate patterns and relationships within the data. By leveraging the strengths of multiple methodologies, it provides robust predictions, empowering investors and analysts with valuable insights for decision-making and risk management strategies in dynamic financial markets.[24]

The Hybrid Two-Stage Financial Stock Forecasting Algorithm combines clustering and ensemble learning for enhanced prediction accuracy. In the first stage, clustering techniques group similar stocks based on historical data patterns. In the second stage, ensemble learning methods, such as Random Forest or Gradient Boosting, are applied to each cluster to predict future stock movements. By leveraging clustering to organize data and ensemble learning to refine predictions, the

model achieves robust forecasting results. This two-stage approach offers a comprehensive methodology for analyzing diverse stock datasets, empowering investors and analysts with valuable insights into market dynamics and aiding in informed decision-making and risk management strategies.[25]

This study explores the efficacy of LSTM and GRU models in stock price prediction, focusing on optimization techniques to enhance their performance. LSTM and GRU are widely recognized RNN designs acclaimed for their knack in grasping temporal patterns within sequential data, rendering them ideal for predicting stock market trends.The research begins by preprocessing the data, including normalization and feature engineering, to prepare it for modeling. Next, LSTM and GRU models are implemented and trained on historical stock market data. To improve model performance, optimization techniques such as hyperparameter tuning and regularization methods are applied.Through extensive experimentation and evaluation, the study identifies optimal configurations for LSTM and GRU models, maximizing their predictive accuracy. Performance metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE) are used to assess the models' effectiveness.The results demonstrate that the optimized LSTM and GRU models outperform baseline models and exhibit strong predictive capabilities in stock price forecasting. The study concludes by highlighting the potential of LSTM and GRU architectures, alongside optimization techniques,by enhancing the accuracy of stock market predictions, providing valuable insights for investors and financial analysts.[26]

This paper introduces a novel cryptocurrency price prediction model leveraging the power of Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and Bidirectional LSTM (bi-LSTM) machine learning algorithms. By harnessing the temporal dependencies within cryptocurrency data, the model aims to forecast future price movements accurately. Through meticulous experimentation and evaluation, the study compares the performance of GRU, LSTM, and bi-LSTM architectures in cryptocurrency price prediction tasks. The results indicate that the bi-LSTM model, which processes data in both forward and backward directions, exhibits superior predictive capabilities compared to standalone GRU and LSTM models. By combining the strengths of these advanced architectures, the proposed model offers promising insights into cryptocurrency price dynamics, facilitating informed decision-making for investors and stakeholders in the cryptocurrency market.[27]

The paper presents a methodology for predicting price changes in ultra-high frequency financial data using Temporal Convolutional Networks (TCNs). TCNs are utilized because they are effective at capturing temporal relationships in sequential data. The study focuses on modeling price changes at a granular level, leveraging the high frequency of financial data. Through extensive experimentation and evaluation, the efficacy of TCNs in predicting price changes is demonstrated. The proposed approach offers a promising solution for forecasting price movements in volatile financial markets,empowering traders and investors to make timely, well-informed decisions. By harnessing the power of TCNs, the model contributes to the advancement of predictive analytics in financial trading, offering insights into market dynamics at ultra-high frequencies.[28]

The paper presents a fresh approach to forecasting stock prices, leveraging a blend of BiLSTM and an enhanced Transformer architecture. By integrating the temporal modeling capabilities of BiLSTM with the attention mechanism of the Transformer, the model aims to capture both short-term and long-term dependencies in stock price data effectively. The study focuses on enhancing the Transformer architecture to better handle sequential data and extract meaningful features for prediction. Through extensive experimentation and evaluation, the proposed method demonstrates improved predictive accuracy compared to traditional approaches. The fusion of BiLSTM and the enhanced Transformer offers a robust framework for stock price forecasting, providing valuable insights for investors and financial analysts. This innovative approach contributes to advancing the field of predictive analytics in financial markets, facilitating more informed decision-making processes.[29]

The paper presents a novel stock price prediction model that combines Dual Attention Mechanism and TCN. The Dual Attention Mechanism allows the model to consider both overarching trends and specific details in stock price data, thereby improving its capability to grasp intricate patterns and relationships. By integrating TCN, the model further leverages the temporal information inherent in sequential data, allowing for more accurate forecasting of price movements. Through rigorous experimentation and evaluation, the proposed model demonstrates superior predictive performance compared to traditional methods. The synergy between Dual Attention Mechanism and TCN offers a robust framework for stock price prediction, providing valuable insights for traders and investors. This innovative approach contributes to advancing the field of financial forecasting, facilitating more informed decision-making processes in the stock market.[30]

The paper delves into time series forecasting techniques with a focus on applications in finance. It explores various methods such as ARIMA, Exponential Smoothing (ETS), and machine learning models like LSTM and Gradient Boosting Machines (GBM). These techniques are applied to financial datasets to predict

future stock prices, market trends, and other financial indicators. The study evaluates the performance of each method using metrics like MSE, RMSE, and MAPE. Through comprehensive experimentation and analysis, The paper sheds light on the advantages and constraints of various prediction methods within financial settings. It offers valuable insights into the practical implications and challenges of time series forecasting in finance, aiding investors, analysts, and policymakers in making informed decisions.[31]

3. Methodology

3.1. Model Architecture of Linear Regressor

The setup of a Linear Regression model seems pretty straightforward compared to fancier models like neural networks. Linear Regression is a basic statistical tool used to figure out how one or more things you're interested in (like sales numbers) are related to other stuff (like advertising spending) by fitting a straight line to your data. Let's break it down:

1. Input Variables (Features): - In Linear Regression, you've got one or more things you're measuring (features), usually called $X_1, X_2, ..., X_n$. These are the bits of data you're going to use to guess the thing you're interested in (the target).

2. Target Variable: - The thing you're trying to figure out or predict (the target) is called Y. That's what you're aiming for based on the stuff you're measuring.

3. The Equation: - The heart of Linear Regression is a simple equation that shows how the things you're measuring relate to what you're trying to predict. It looks like this:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon$$

- Here's what all those parts mean: - Y is your guess for the thing you're interested in. - β_0 is where your guess starts (like if you spent zero on advertising, how many sales would you get?). - $\beta_1, \beta_2, ..., \beta_n$ are how much each bit of data you're measuring affects your guess. - $X_1, X_2, ..., X_n$ are the bits of data you're measuring. - ϵ is just a fancy way of saying "stuff we don't know or can't measure" that affects our guess.

4. Training: - When you train a model, you're teaching it the best values for all those βs by comparing its guesses to what really happened. It's like adjusting the settings on your guess machine until it gets things right.

5. Making Predictions: - Once it's trained, your Linear Regression model can take new data and use the equation to make a guess about what will happen.

6. Checking How Well It Works: - We evaluate the model by seeing how close its guesses are to what really happened, using measures like R^2 or how much error there is.

So, in a nutshell, Linear Regression is about finding the relationship between what you're measuring and what you want to know, teaching a model to make good guesses about that relationship, and then using those guesses to predict new stuff. It's a basic but essential tool in both statistics and machine learning, often used as a starting point for figuring out more complex relationships in data.

3.2. Model Architecture of Long Short-Term Memory (LSTM)

LSTM networks are a unique type of neural network designed specifically to address the "vanishing gradient problem," enabling them to effectively capture long-term patterns in sequential data. They excel in various tasks such as language understanding, time-based data analysis, and speech recognition. Let's explore their structure:

1. Input Layer: This is where sequential data is fed into the network, such as event sequences, time-based data, or words in a sentence for language tasks.

2. LSTM Cells: These are the fundamental units of an LSTM network, structured to handle sequential data efficiently. Each LSTM cell comprises several key components: - Cell State (C_t): Acts as the memory of the cell, crucial for retaining context over extended periods. - Hidden State (h_t): Represents the output of the cell, containing relevant information for predictions or subsequent cells in the sequence. - Forget Gate (f_t): Determines what information to discard from the cell's memory based on the previous hidden state (h_{t-1}) and the current input (x_t). - Input Gate (i_t): Controls what new information to store in the cell state. - Output Gate (o_t): Regulates which parts of the cell state to utilize for computing the hidden state (h_t).

3. Output Layer: Tailored according to the network's specific task. For sequence prediction, it may consist of layers followed by activation functions. In classification tasks, a softmax layer might be employed to determine probabilities.

4. Training: During training, the network adjusts its internal parameters, such as weights and biases, using a technique called backpropagation through time (BPTT). This method extends regular backpropagation to handle sequential data effectively.

5. Activation Functions: LSTMs employ specialized functions like sigmoid and hyperbolic tangent (tanh) to regulate information flow and compute the hidden state.

In summary, LSTM networks leverage specialized cells adept at retaining critical information over extended periods, making them ideal for tasks involving sequential data. Their versatility has led to widespread adoption across various fields due to their efficacy in understanding sequential order.

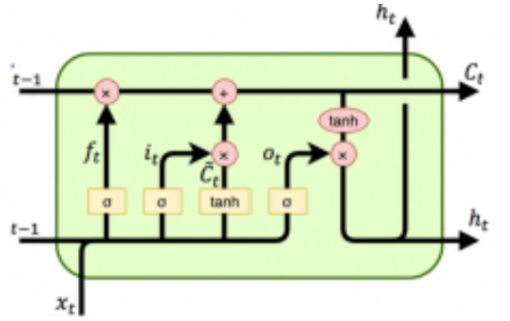

Figure 1. Architecture of LSTM Model

3.3. Model Architecture of Gated Recurrent Unit (GRU)

GRU networks are another flavor of neural networks, particularly geared towards handling sequences of data over time, just like LSTM networks. They're pretty neat because they're designed to catch onto those long-term patterns in data sequences, and they do it in a way that's a bit more efficient than traditional LSTM networks. People use GRUs all over the place, from understanding language to analyzing time-based data and even recognizing speech. Let's dig into how a GRU network is set up:

1. Input Layer: - Similar to other setups with sequences of data, the input layer here takes in that sequential data. Think of things like sequences of events, time-based data, or even words in a sentence for language tasks.

2. GRU Cell: - This is where the magic happens. The GRU network is made up of these GRU cells, organized in a way that's good for handling sequences. Each cell has some key parts: - Update Gate (z_t): This gate decides how much of the old stuff to keep and how much of the new stuff to let in. It takes the previous hidden state (h_{t-1}) and the current input (x_t) and figures out a gate vector (z_t) to decide what to update. - Reset Gate (r_t): This gate figures out what old stuff to forget. It looks at the previous hidden state (h_{t-1}) and the current input (x_t) to come up with a reset gate vector (r_t). - Current Memory Content (h'_t): This is like the new stuff we're considering for this time step. It's a mix of the old hidden state (h_{t-1}) and the current input (x_t), calculated using the reset gate and current input. - Hidden State (h_t): This is what the GRU cell spits out. It's a blend of the previous hidden state (h_{t-1}) and the current memory content (h'_t), controlled by the update gate.

3. Output Layer: - Similar to LSTM networks, this part changes based on what the network is doing. For predicting sequences, it might have some layers followed by activation functions. For classifying things, it could use a softmax layer to figure out probabilities.

4. Training: - When it's learning, the network tunes its internal bits (like weights and biases) using something called BPTT, which is kind of like a fancier version of regular backpropagation, tweaked for sequences.

5. Activation Functions: - GRUs use special functions like the sigmoid and hyperbolic tangent (tanh) to help with managing information flow and figuring out the hidden state.

So, GRU networks are made up of these special cells that are really good at picking up on long-term patterns in sequences of data. They're simpler than LSTM networks but still super effective at understanding the order of things, which is why they're so popular in all sorts of fields where sequences matter.

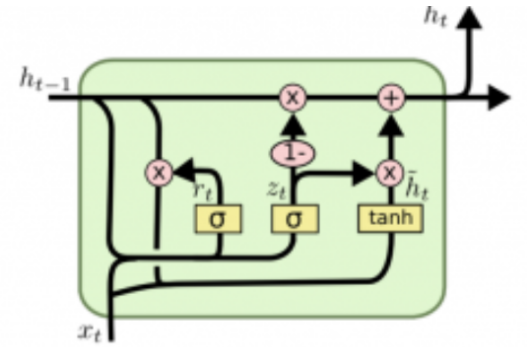

Figure 2. Architecture of GRU Model

3.4. Model Architecture of Convolutional Neural Networks (CNN)

CNNs are a sophisticated class of neural networks predominantly utilized in computer vision tasks like image categorization, object localization, and segmentation. Let's delve into their architecture:

1. Input Layer: Typically, CNNs receive images or tensors representing multiple images.

2. Convolutional Layer: This layer employs adaptable filters (kernels) to extract specific features from input images via convolution operations, preserving spatial correlations among pixels.

3. Activation Function: Following convolution, an activation function (often ReLU) introduces non-linearity, aiding in learning intricate patterns and correlations.

4. Pooling Layer: These layers reduce spatial dimensions (width and height) of feature maps through downsampling techniques like max pooling, curbing complexity and overfitting.

5. Fully Connected (Dense) Layer: Extracted features from convolutional and pooling layers are flattened and fed into one or more fully connected layers for classification/regression.

6. Output Layer: Generates final predictions, with neuron count varying based on task. For instance, in image classification, it aligns with class count, often with softmax activation for probability estimation.

Optional Components: - Dropout Layer: A regularization method deactivating random neurons during training to prevent overfitting. - Batch Normalization: Standardizes layer outputs, aiding convergence and reducing initialization dependency. - Skip Connections: Featured in architectures like ResNet to mitigate vanishing gradients and enhance learning efficiency.

These elements form the core of CNN architecture, with numerous variations and adaptations tailored to diverse tasks and domains.

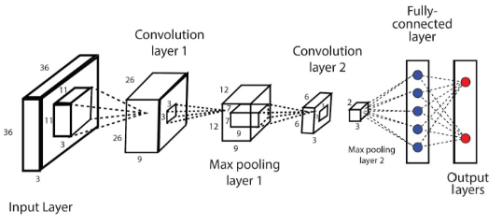

Figure 3. Architecture of CNN Model

3.5. Model Architecture of Recurrent Neural Networks (RNN)

RNNs represent a category of neural networks tailored for handling sequential data by capturing temporal correlations. The RNN architecture integrates recurrent connections, facilitating the retention of information across time. Below is an overview of the typical RNN architecture:

Input Layer: Analogous to other neural networks, RNNs initiate with an input layer. However, for sequential data such as text or time series, the input constitutes a sequence of vectors, with each vector denoting a single timestep within the sequence.

Recurrent Connections: The pivotal characteristic of an RNN lies in its recurrent connections, enabling the transmission of information from one timestep to the subsequent one. At each timestep, the input combines with the output from the preceding timestep to yield the current output. Mathematically, this can be expressed as:

$$h_t = f(W_{ih} \cdot x_t + W_{hh} \cdot h_{t-1} + b_h)$$

Where: - h_t signifies the hidden state at timestep t. - x_t denotes the input at timestep t. - W_{ih} denotes the weight matrix linking the input to the hidden state. - W_{hh} denotes the weight matrix linking the hidden state to itself (recurrent weights). - b_h symbolizes the bias vector. - f represents the activation function, frequently a non-linear function like tanh or ReLU.

Output Layer: The output layer's configuration in an RNN varies depending on the task at hand. For sequence prediction tasks, the output can be generated at each timestep. Conversely, for tasks such as sequence classification, sentiment analysis, or language modeling, the output typically emerges after processing the entire sequence. In sequence classification scenarios, for instance, the output can be derived by applying a softmax activation function to the final hidden state.

Optional Components: - Bidirectional RNNs: In certain scenarios, processing the input sequence in both forward and backward directions proves advantageous. Bidirectional RNNs accomplish this feat by employing two distinct recurrent layers: one handling the sequence forward and the other in reverse. This approach enables the model to assimilate information from both past and future contexts. - GRUs and LSTM: Conventional RNNs often grapple with the vanishing gradient problem, impeding their ability to learn prolonged dependencies. GRUs and LSTMs represent RNN variants that leverage gating mechanisms to tackle this challenge. Their widespread adoption across diverse applications stems from their efficacy in capturing long-term dependencies.

In essence, the RNN architecture revolves around its recurrent connections, facilitating the processing of sequential data while retaining memory of prior timesteps. A plethora of adjustments and enhancements have been proposed to bolster the performance and efficacy of RNNs across various tasks and domains.

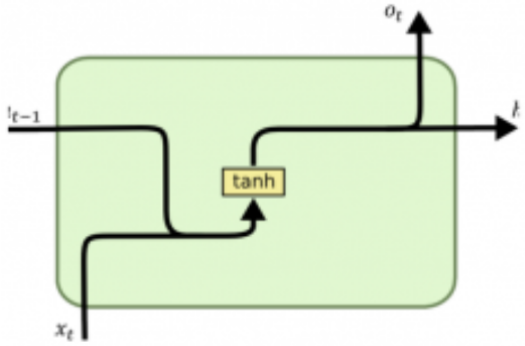

Figure 4. Architecture of RNN Model

3.6. Model Architecture of Temporal Convolutional Networks (TCNs)

TCNs represent a category of neural networks tailored for modeling temporal sequences. Known for their

ability to grasp long-term connections within sequences while maintaining computational efficiency, TCNs have garnered considerable interest. The hallmark of a TCN's architecture lies in its stacked dilated causal convolutional layers. Below is an outline of the typical TCN architecture:

Input Layer: TCNs commence with an input layer akin to other neural networks. Typically, the input for a TCN comprises a sequence of vectors representing temporal data, with each vector signifying a distinct timestep within the sequence.

Dilated Causal Convolutional Layers: At the heart of a TCN resides a succession of dilated causal convolutional layers. These layers administer convolutional filters to the input sequence in a causal and dilated manner.

Causal Convolution: Causal convolution ensures that each timestep's output relies solely on the current and preceding inputs, mirroring the autoregressive trait of sequential data. This mechanism averts information leakage from subsequent timesteps.

Dilated Convolution: Dilated convolution entails skipping input values by a designated step size (dilation rate) during the convolutional filter application. This capability enables TCNs to efficiently capture extended dependencies across the input sequence, surpassing conventional convolutional networks in efficacy.

Residual Connections: To facilitate the training of deep TCNs and alleviate the vanishing gradient issue, residual connections find utility. These connections expedite gradient flow through the network by integrating skip connections that bypass one or more convolutional layers.

Temporal Pooling: Integration of temporal pooling layers into the TCN architecture enables temporal resolution reduction of feature maps while retaining crucial information. Operations such as max pooling or average pooling across temporal dimensions facilitate feature map downsampling.

Output Layer: The configuration of a TCN's output layer varies based on the task at hand. For sequence prediction tasks, outputs may be generated at each timestep or solely at the final timestep post-entire sequence processing. In the context of sequence classification or regression tasks, output generation may involve pooling operations succeeded by a fully connected layer.

Activation Functions: Post-convolutional layer applications typically entail non-linear activation functions such as ReLU or tanh, serving to instill non-linearity into the network.

Optional Components: - Temporal Dropout: Analogous to dropout in traditional neural networks, temporal dropout may be applied to convolutional layer outputs to forestall overfitting and enhance generalization. - Multi-Head Attention: Certain TCN variants incorporate multi-head attention mechanisms inspired by Transformer architectures to capture intricate temporal dependencies.

In summation, the hallmark of a TCN lies in its stacked dilated causal convolutional layers, which efficaciously enable extended dependency modeling within temporal sequences. Diverse adaptations and refinements may be introduced in alignment with the unique requisites of a given task or application.

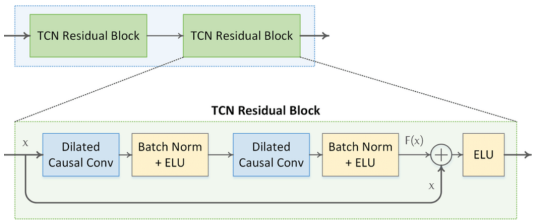

Figure 5. Architecture of TCN Model

4. Experiments and Results

4.1. Datasets

We utilized the Nifty 50 dataset sourced from Kaggle, encompassing features such as strike price, callOpen, callHigh, callLow, callClose, callVolume, callOI, putOpen, putHigh, putLow, putClose, putVolume, putOI, open, high, low, close, and volume as input variables. We applied the LSTM and GRU algorithms for predicting the 'spot price', employing optimizers including Adadelta, Adagrad, Adamax, Nadam, Adam, SGD, and RMSprop. Our analysis involves comparing the results, including R2 Score, RMSE, and MAEe, across Linear Regression, LSTM, GRU and other models.

4.2. Evaluation Matrics

When evaluating predictive models, it's important to include key evaluation metrics that provide understanding how well the model performs. Here's an overview of the evaluation metrics along with their formulas that is used to check the performance of the model : **R-squared (R2) Score:**

- R2 score quantifies the proportion of variance in the dependent variable that can be explained by the independent variables. - Formula:

$$R^2 = 1 - \frac{\text{Sum of Squared Residuals}}{\text{Total Sum of Squares}}$$

- R2 score ranges between 0 and 1, with 1 indicating a perfect fit and 0 suggesting no linear relationship between independent and dependent variables.

Root Mean Squared Error (RMSE): - RMSE computes the average magnitude of errors between predicted and actual values. - Formula:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

- Lower RMSE values signify better model fit, offering an absolute measure of model accuracy.

Mean Absolute Error (MAE): - MAE calculates the average of absolute errors between predicted and actual values. - Formula:

$$MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n}$$

- MAE is less affected by outliers compared to RMSE and provides a straightforward interpretation of average prediction error.

Mean Absolute Percentage Error (MAPE) : - MAPE quantifies the average percentage error of predictions relative to actual values. - Formula:

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

- Particularly useful when comparing variables with widely varying scales.

Mean Squared Error (MSE): Mean Squared Error (MSE) is a metric used to evaluate regression models. It calculates the average squared difference between actual (y_i) and predicted ($\hat{y}_i$) values. The formula is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where n is the number of data points. MSE penalizes larger errors more than smaller ones. Lower MSE values indicate better model performance.

Below are the performance of different models for Nifty 50 data set

Metrics	Linear Regression
R2 Score	72.7
RMSE	63.5011
MAPE	0.08%
MAE	35.5581

Figure 6. Performance of Linear Regression Model

LSTM					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	75854.9631	0.75885	297.952	201.8921	0.6227
Adagrad	54933.1184	0.8732	234.3782	193.7319	0.4259
Adamax	5762.9469	0.9867	75.9141	49.6948	0.1084
Nadam	5216.3963	0.988	72.2246	47.673	0.1042
SGD	15330.539	0.9646	123.8166	98.7318	0.2161
RMSprop	7020.9062	0.9838	83.7908	56.5806	0.1233

Figure 7. Performance of LSTM Model

GRU					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	93554.6415	0.94741	305.867	260.9861	0.5686
Adagrad	24178.6716	0.9442	155.4949	126.8906	0.2781
Adamax	5642.4037	0.987	75.1159	50.2076	0.1097
Nadam	5658.7294	0.9869	75.2245	49.4914	0.108
SGD	6829.0096	0.9842	82.6378	56.8203	0.1242
RMSprop	6927.6384	0.984	83.2324	56.5498	0.1233

Figure 8. Performance of GRU Model

LSTM+GRU					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	7050.8044	0.9837	83.9691	59.384	0.1294
Adagrad	5895.8859	0.9864	76.7847	50.7318	0.1108
Adamax	5860.6262	0.9865	76.5547	54.0951	0.118
Nadam	5272.8249	0.9878	72.6142	48.7395	0.1065
SGD	5375.669	0.9876	73.319	46.9763	0.1025
RMSprop	4478.8718	0.9897	66.9244	42.2145	0.0921

Figure 9. Performance of LSTM+GRU Model

CNN					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	52934.955	0.8778	230.076	190.8861	0.4177
Adagrad	12939.7441	0.9701	113.753	90.6965	0.1985
Adamax	5859.4039	0.9865	76.5467	49.2663	0.1074
Nadam	5767.4179	0.9867	75.9435	52.3304	0.1142
SGD	6347.5376	0.9853	79.6714	53.3964	0.1167
RMSprop	5074.8851	0.9883	71.2382	44.9767	0.0981

Figure 10. Performance of CNN Model

RNN					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	69844.658	0.8388	264.2814	225.3968	0.4931
Adagrad	10910.4228	0.9748	104.453	74.299	0.1624
Adamax	5355.2958	0.9876	73.1799	47.9462	0.1047
Nadam	6873.1313	0.9841	82.9044	58.9563	0.1289
SGD	6881.045	0.9841	82.9521	56.4831	0.1235
RMSprop	14452.6627	0.9666	120.2192	96.6457	0.2109

Figure 11. Performance of RNN Model

CNN+RNN					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	25640.53	0.9408	160.1266	131.3903	0.2872
Adagrad	13014.575	0.97	114.0814	90.216	0.1976
Adamax	4214.4755	0.9903	64.919	40.9265	0.0894
Nadam	4581.0067	0.9894	67.6831	49.255	0.1075
SGD	5609.7136	0.9871	74.898	49.3195	0.1078
RMSprop	4107.5462	0.9905	64.0901	41.4234	0.0906

Figure 12. Performance of CNN+RNN Model

TCN					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	77885.8381	0.8202	279.0803	229.758	0.5047
Adagrad	12303.342	0.9716	110.9204	82.2482	0.1799
Adamax	5878.8272	0.9864	76.6735	49.8137	0.1085
Nadam	4263.4586	0.9902	65.2952	40.5133	0.0885
SGD	6754.5275	0.9844	82.1859	55.4822	0.1212
RMSprop	5924.7396	0.9863	76.9723	55.1807	0.1207

Figure 13. Performance of TCN Model

5. Discussion and Conclusion

Linear regression, LSTM, GRU, CNN, RNN, TCN, LSTM+GRU, CNN+RNN, TCN+CNN, TCN+LSTM Model were trained and tested with 2350, and 235 data points for 50 iterations. Based on the performance metrics obtained from the evaluation of different models on the Nifty 50 dataset, the following conclusions can be drawn:

TCN+CNN					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	16254.0159	0.9625	127.4912	86.5392	0.1903
Adagrad	5313.8676	0.9877	72.8963	46.8938	0.1026
Adamax	7958.5993	0.9816	89.211	67.7208	0.1486
Nadam	5924.6277	0.9863	76.9716	55.0229	0.12
SGD	5643.228	0.987	75.1214	49.2614	0.1078
RMSprop	4045.9538	0.9907	63.6078	42.515	0.0929

Figure 14. Performance of TCN+CNN Model

TCN+LSTM					
Optimizer	MSE	R2 Score	RMSE	MAE	MAPE
Adadelta	36096.6868	0.9167	189.9913	141.2014	0.3106
Adagrad	7426.1175	0.9829	86.1749	58.9033	0.1288
Adamax	3468.1775	0.992	58.8912	36.8767	0.0806
Nadam	3932.9992	0.9909	62.7136	43.3225	0.0946
SGD	4843.3642	0.9888	69.5943	44.8232	0.098
RMSprop	7200.1425	0.9834	84.8537	61.9108	0.1348

Figure 15. Performance of TCN+LSTM Model

Comparison of Optimizers: - Across different models (LSTM, GRU, CNN, RNN, TCN) and their combinations (LSTM+GRU, CNN+RNN, TCN+CNN, TCN+LSTM), various optimizers were evaluated. - Adadelta consistently yields relatively lower R2 scores compared to other optimizers across most models. It performs relatively poorer especially in models like LSTM, RNN, and TCN, which indicates that it might not be the most suitable optimizer for these architectures. - Optimizers such as Adamax, Nadam, and RMSprop consistently yield higher R2 scores across different models, indicating their effectiveness in optimizing model parameters and improving predictive performance. - Adagrad shows competitive performance, particularly in models like CNN and TCN. - SGD performs relatively well but is outperformed by more advanced optimizers like Adamax, Nadam, and RMSprop.

Model Performance: - LSTM and GRU models tend to exhibit higher R2 scores compared to RNN, CNN, and TCN, indicating their ability to capture temporal dependencies more effectively. - The combination models (LSTM+GRU, CNN+RNN, TCN+CNN, TCN+LSTM) show varying performance across different optimizers, suggesting that the choice of architecture and optimizer combination significantly impacts predictive performance. - TCN architecture, while showing potential because of its effective handling of long-term connections, tends to show relatively lower R2 scores compared to LSTM and GRU models across most optimizers.

Overall Implications: - The selection of optimizer significantly influences the predictive accuracy of deep learning models on the dataset at hand. - Advanced optimizers such as Adamax, Nadam, and RMSprop generally yield better results compared to Adadelta, especially in models like LSTM, RNN, and TCN. - LSTM and GRU architectures demonstrate superior performance in capturing temporal dependencies and predicting target variables compared to other architectures like RNN, CNN, and TCN. - The combination of different architectures (LSTM+GRU, CNN+RNN, TCN+CNN, TCN+LSTM) offers potential for further improvement in predictive performance, and the choice of optimizer should be carefully considered when designing such hybrid models. - Further experimentation and tuning, including hyperparameter optimization and architectural modifications, may lead to even better performance and should be explored to maximize predictive accuracy.

.

References

[1] M Ananthi and K Vijayakumar. Retracted article: stock market analysis using candlestick regression and market trend prediction (ckrm). *Journal of Ambient Intelligence and Humanized Computing*, 12(5):4819–4826, 2021.

[2] Kunal Pahwa and Neha Agarwal. Stock market analysis using supervised machine learning. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pages 197–200. IEEE, 2019.

[3] Wenjie Lu, Jiazheng Li, Jingyang Wang, and Lele Qin. A cnn-bilstm-am method for stock price prediction. *Neural Computing and Applications*, 33:4741–4753, 2021.

[4] Pengfei Yu and Xuesong Yan. Stock price prediction based on deep neural networks. *Neural Computing and Applications*, 32:1609–1628, 2020.

[5] Saloni Mohan, Sahitya Mullapudi, Sudheer Sammeta, Parag Vijayvergia, and David C Anastasiu. Stock price prediction using news sentiment analysis. In *2019 IEEE fifth international conference on big data computing service and applications (BigDataService)*, pages 205–208. IEEE, 2019.

[6] Achyut Ghosh, Soumik Bose, Giridhar Maji, Narayan Debnath, and Soumya Sen. Stock price prediction using lstm on indian share market. In *Proceedings of 32nd international conference on*, volume 63, pages 101–110, 2019.

[7] Sidra Mehtab and Jaydip Sen. A robust predictive model for stock price prediction using deep learning and natural language processing. *arXiv preprint arXiv:1912.07700*, 2019.

[8] Lounnapha Sayavong, Zhongdong Wu, and Sookasame Chalita. Research on stock price prediction method based on convolutional neural network. In *2019 international conference on virtual reality and intelligent systems (ICVRIS)*, pages 173–176. IEEE, 2019.

[9] Guangyu Ding and Liangxi Qin. Study on the prediction of stock price based on the associated network model of lstm. *International Journal of Machine Learning and Cybernetics*, 11:1307–1317, 2020.

[10] Jun Zhang, Yu-Fan Teng, and Wei Chen. Support vector regression with modified firefly algorithm for stock price forecasting. *Applied Intelligence*, 49:1658–1674, 2019.

[11] Chenglin Xiao, Weili Xia, and Jijiao Jiang. Stock price forecast based on combined model of ari-ma-ls-svm. *Neural Computing and Applications*, 32:5379–5388, 2020.

[12] Hongli Niu, Kunliang Xu, and Weiqing Wang. A hybrid stock price index forecasting model based on

variational mode decomposition and lstm network. *Applied Intelligence*, 50:4296–4309, 2020.

[13] Md Arif Istiake Sunny, Mirza Mohd Shahriar Maswood, and Abdullah G Alharbi. Deep learning-based stock price prediction using lstm and bi-directional lstm model. In *2020 2nd novel intelligent and leading emerging sciences conference (NILES)*, pages 87–92. IEEE, 2020.

[14] Milad Shahvaroughi Farahani and Seyed Hossein Razavi Hajiagha. Forecasting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models. *Soft computing*, 25(13):8483–8513, 2021.

[15] Yan Wang and Yuankai Guo. Forecasting method of stock market volatility in time series data based on mixed model of arima and xgboost. *China Communications*, 17(3):205–221, 2020.

[16] Zhigang Jin, Yang Yang, and Yuhong Liu. Stock closing price prediction based on sentiment analysis and lstm. *Neural Computing and Applications*, 32:9713–9729, 2020.

[17] Xiaodan Liang, Zhaodi Ge, Liling Sun, Maowei He, and Hanning Chen. Lstm with wavelet transform based data preprocessing for stock price prediction. *Mathematical Problems in Engineering*, 2019, 2019.

[18] Widodo Budiharto. Data science approach to stock prices forecasting in indonesia during covid-19 using long short-term memory (lstm). *Journal of big data*, 8:1–9, 2021.

[19] Yang Zhao and Zhonglu Chen. Forecasting stock price movement: New evidence from a novel hybrid deep learning model. *Journal of Asian Business and Economic Studies*, 29(2):91–104, 2021.

[20] GWRI Wijesinghe and RMKT Rathnayaka. Stock market price forecasting using arima vs ann; a case study from cse. In *2020 2nd International Conference on Advancements in Computing (ICAC)*, volume 1, pages 269–274. IEEE, 2020.

[21] Jimmy Ming-Tai Wu, Zhongcui Li, Norbert Herencsar, Bay Vo, and Jerry Chun-Wei Lin. A graph-based cnn-lstm stock price prediction algorithm with leading indicators. *Multimedia Systems*, 29(3):1751–1770, 2023.

[22] Joshua Zoen Git Hiew, Xin Huang, Hao Mou, Duan Li, Qi Wu, and Yabo Xu. Bert-based financial sentiment index and lstm-based stock return predictability. *arXiv preprint arXiv:1906.09024*, 2019.

[23] Yi Ji, Alan Wee-Chung Liew, and Lixia Yang. A novel improved particle swarm optimization with long-short term memory hybrid model for stock indices forecast. *Ieee Access*, 9:23660–23671, 2021.

[24] Mustafa Göçken, Mehmet Özçalıcı, Aslı Boru, and Ayşe Tuğba Dosdoğru. Stock price prediction using hybrid soft computing models incorporating parameter tuning and input variable selection. *Neural Computing and Applications*, 31:577–592, 2019.

[25] Ying Xu, Cuijuan Yang, Shaoliang Peng, and Yusuke Nojima. A hybrid two-stage financial stock forecasting algorithm based on clustering and ensemble learning. *Applied Intelligence*, 50:3852–3867, 2020.

[26] Ya Gao, Rong Wang, and Enmin Zhou. Stock prediction based on optimized lstm and gru models. *Scientific Programming*, 2021:1–8, 2021.

[27] Mohammad J Hamayel and Amani Yousef Owda. A novel cryptocurrency price prediction model using gru, lstm and bi-lstm machine learning algorithms. *Ai*, 2(4):477–496, 2021.

[28] Wei Dai, Yuan An, and Wen Long. Price change prediction of ultra high frequency financial data based on temporal convolutional network. *Procedia Computer Science*, 199:1177–1183, 2022.

[29] Shuzhen Wang. A stock price prediction method based on bilstm and improved transformer. *IEEE Access*, 2023.

[30] Yifeng Fu and He Xiao. Stock price prediction model based on dual attention and tcn. *Available at SSRN 4282842*, 2022.

[31] Viswapriya Misra. Time series forecasting with applications to finance. 2021.