Svedberg Open
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

Stock Price Prediction Using Optimized Generative Adversarial Network (Gan) with Attention Bidirectional Gated Recurrent Unit and Convolution Neural Networks (Attbigru-Cnn)

Bhuvaneshwari P V[1], Radhakrishnan Vignesh[2]

[1]Presidency School of Artificial Intelligence and Advance Computing, Presidency University, India,
Email: BHUVANESHWARI.20233CSE0032@presidencyuniversity.in
[2]Presidency School of Artificial Intelligence and Advance Computing, Presidency University, India,
Email:vigneshr@presidencyuniversity.in
*Corresponding author: Bhuvaneshwari P V

Abstract
Forecasting stock price variations is one of the important problems for economists. A robust and precise forecast substantially mitigates investment risks for stakeholders. Currently, the fastest growth in Artificial Intelligence (AI) and Deep Learning (DL) models shed light on the intricacies of Stock price prediction (SPP). While DL models has demonstrated remarkable efficacy in long term SPP, it does handle the challenges such as intricate model structures, large training processes, substantial computational demands, and high cost of utilization. This study proposes a SPP model using optimized Generative Adversarial Network (GAN). It consists of Attention Bidirectional Gated Recurrent Unit (AttBiGRU) model as a discriminator that separates the generated stock price from the true stock price, and as a generator that uses past stock price inputs to produce future stock prices. Moreover, the hyperparameters of GAN are optimized by applying the Walrus Optimization Algorithm (WaOA). The proposed GAN:AttBIGRU-CNN model has been applied over 4 bench mark datasets. Experimental results state that GAN:AttBIGRU-CNN model attains minimized statistical error measures while predicting the stock prices, when compared to the existing models.

Keywords: Stock price prediction (SPP), Generative Adversarial Network (GAN), Attention Bidirectional Gated Recurrent Unit (AttBiGRU) , Convolutional Neural Network (CNN), Walrus Optimization Algorithm (WaOA)

1. Introduction

Stock price prediction (SPP) movements pose a significant challenge within the realm of economics. The stock market, spanning a rich history of four centuries, serves as a vital arena for the exchange, trading, and circulation of stocks [1]. It stands as a pivotal platform through which companies can raise essential funds by issuing stocks, channelling a significant influx of capital into the market. This influx not only fosters the consolidation of capital but also enhances the organic composition of enterprise capital, thereby exerting a profound influence on the advancement of the commodity economy [2].

Given its pivotal role, the stock market is widely recognized as a barometer, offering valuable insights into the economic and financial pulse of a country or region. Its fluctuations and trends reflect the broader landscape of economic and financial activities, serving as a key indicator for policymakers, investors, and analysts alike [3]. The inception of the Chinese stock market occurred in the early 1990, trailing behind the establishment of Western stock markets. Despite its relatively late start, the Chinese stock market has swiftly grown in both scale and organizational structure, comparable to its Western counterparts [4]. As China's economy experiences rapid expansion, the stock market has undergone substantial growth, attracting an increasing number of participants keen on engaging in stock investments [5].

Investors place significant emphasis on tracking the shifting trends of stock prices within the market. The variations in the factors that impact stock prices include a different factors, including alterations in national policies, the economic environments at home and abroad, and worldwide affairs. It is noted that stock price variations often exhibit nonlinearity, adding to the complexity of forecasting [6]. Forecasting stock price variations is one of the important problems for economists. A robust and precise forecast concerning these changes holds importance as it substantially mitigates investment risks for

stakeholders [7]. Such forecasts empower investors to incorporate projected stock prices into their investment strategies, thereby optimizing their potential returns and aiding in the maximization of investment gains [8].

Currently, Artificial Intelligence's quick development, particularly the DL models, shed light on the intricacies of SPP [9]. Increasingly, financial organizations are moving towards employing DL models for investment decisions, aiming to reduce intervention of human.

Given the intricate nature pertaining to financial markets,the fusion of DL models forecasting financial time series stands out as a highly promising direction of research [10]. Numerous studies have showcased the efficacy of DL models in predicting financial market trends. Long Short-term Memory (LSTM), CNN, and Recurrent Neural Networks (RNN) are among the prevailing DL models that exhibit superior predictive capabilities in contrast to conventional methods for machine learning [11-12]

1.1 Motivation and Problem Identification

In recent times, DL models has made significant strides in domains like natural language processing and image processing, garnering considerable success. Researchers have extended its application to forecast financial time series.

While DL models has demonstrated remarkable efficacy in long term SPP, it does handle the challenges such as intricate model structures, large training processes, substantial computational demands, and high cost of utilization. Consequently, for short term SPP, conventional models have garnered increasing attention from scholars owing to their simpler architectures, ease of training, and lower computational requirements. The DL models have better non-linear fitting capabilities, making it well suited to forecast in the stock market, characterized by its non-linear and highly volatile nature.

The DL model AttBiGRU offers advantages such as smaller model size, reduced complexity, and enhanced robustness, making it adaptable for short term SPP. However, the effectiveness of AttBiGRU hinges significantly on the initial weights and threshold settings, necessitating optimization for improved predictive performance.

1.2 Objectives

The main objectives of this research work are:

- ✓ To design an automated SPP model using GAN
- ✓ To apply advanced DL models as generator and discriminator modules of GAN
- ✓ To fine tune the hyperparameters of GAN by applying optimization techniques.

1.3 Main Contributions

One of the most effective models for making predictions is the GAN. The model's adversarial discriminator and generator contribute to the correctness of the outcome. Hence this study introduces a new model termed GAN:OBiGRU-CNN for stock price prediction.

The main contributions of the proposed GAN:AttBIGRU-CNN model are highlighted below:

- The generator module of GAN is trained with AttBiGRU model that creates future stock prices by entering previous stock prices.
- The discriminator module of GAN is trained with CNN to distinguish between the generated and real stock prices.
- The prediction performance is enhanced by hyperparameter tuning, using WaOA.

2. Related works

Liu et al. [13] presented ESSA (enhanced Sparrow search algorithm) has been introduced to enhance the optimization of the threshold parameters and initial weights of the BPNN (Backpropagation Neural Network). This improved algorithm is then deployed in the forecasting of stock market indices.

Wang et al. [14] presented deep Transformer model for SPP. By integrating the encoder decoder based structure and multi head attention module, the Transformer model proves accomplishing at capturing the intricate rules governing stock market behavior. MAPE and MSE values achieved were 0.954 and 0.007 on the CSI 300 dataset.

Rezaei et al. [15] presented DL model frequency decomposition for SPP. Initially, the CEEMD (Complete Ensemble Empirical Mode Decomposition) was used for decomposing the frequencies. Then, the CNN with the LSTM was used for extracting features and time series prediction. MAPE and MAE values achieved were 1.05 and 164 on Nikkei225 dataset.

Gülmez and Burak [16] identified stock price forecasting using LSTM with ARO (artificial rabbit's optimizer). Here, the parameters of the LSTM were optimized by the ARO for enhancing the prediction

performance. The experimentation was demonstrated on the DJIA dataset and achieved better R^2 and MAPE values of 0.956 and 0.020.

Zhao et al. [17] leveraged user generated comments from online platforms to supplement traditional stock market data. ECNN (enhanced CNN) was employed to extract sentiment representations from these comments. Subsequently, DAE (denoised autoencoder) was utilized to extract pertinent features from the stock market data, thereby enhancing prediction accuracy.

Jiang et al. [18] presented the HMM-A-LSTM (Hidden Markov Model-Attentive-LSTM) for modeling SPP. This existing framework harnesses the insights gleaned from the HMM to guide the learning of hidden states within A-LSTM. Experimentation was carried out on the six real-time datasets and achieved 10% improvement than other models.

Venkateswararao and Venkata Ramana Reddy [19] presented hybrid model for SPP. Here, an enhanced butterfly optimizer was used for removing noise and the BPO (brown planthopper optimizer) was used for selecting optimal features. The experimentation was performed on the social media and 11 stock market indices data.

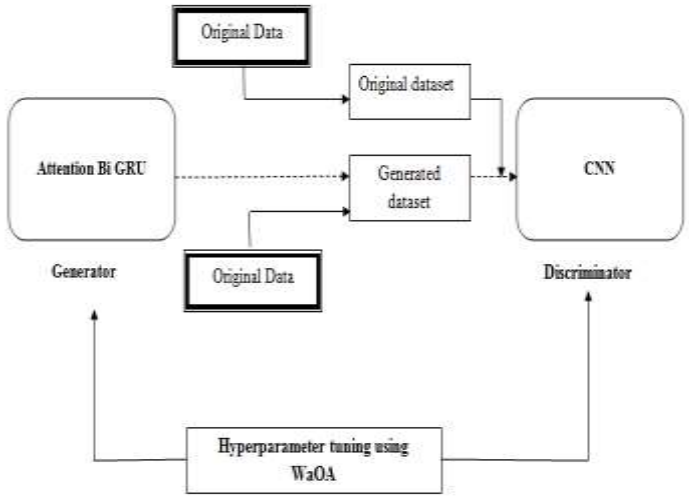
Sivadasan et al. [20] presented SPP model using the GRU with LSTM models. The GRU with LSTM was used for extracting essential features and characteristics. Initially, the sliding window was utilized for analyzing the day-wise and forecasting the future. Here, the MAPE value achieved was 1.191.

3. Proposed Solution

3.1 Overview

With CNN acting as a discriminator to classify the generated and real stock prices, and AttBiGRU acting as a generator that takes historical stock prices as inputs to produce future stock prices, this study suggests a stock prediction model utilizing GAN:AttBIGRU-CNN. Here, WaOA does the best procedure. The architecture of the suggested GAN:AttBiGRU-CNN model is presented in Figure 1.

Figure 1 Architecture of proposed GAN:AttBiGRU-CNN model



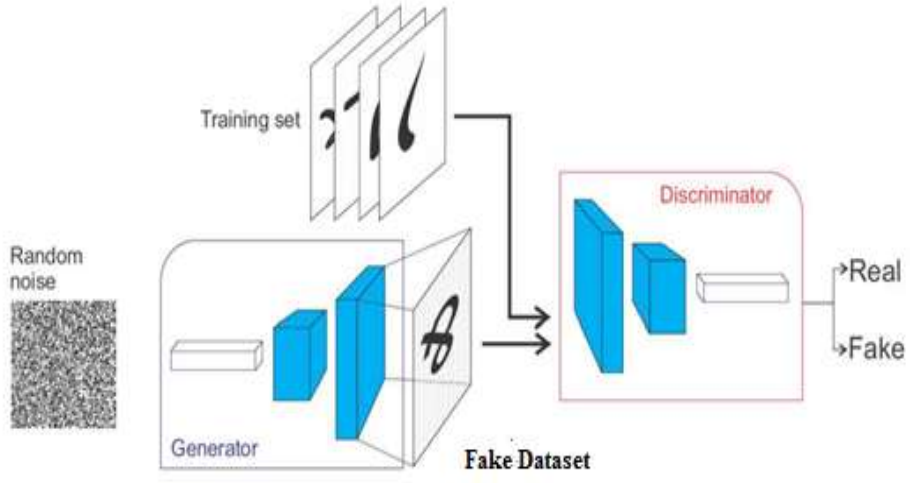
3.2 Basic GAN

GAN relies on zero-sum non-cooperative games and is formulated as a minimax optimization problem. A discriminator (D) and a generator (G) are the two primary parts of a GAN. G generates data which closely resembles the original data. On the other hand, D classifies the data as generated or original.

Training D is similar to training any other neural network. Real data and generated data from G are classified by D. D has loss (d_loss) and accuracy (d_acc) functions that must be validated. The loss function punishes D for misclassifying either a phony sample as real or a true sample as fake. It also utilizes back-propagation to update the weights of the discriminator. In a similar manner, D classifies samples produced by the generator as genuine or fraudulent. Then, the data are sent into a loss function, which uses back-propagation to adjust the generator's weights and penalises it for not being able to trick the discriminator. Due to the D's inability to discern between real and false, its performance deteriorates as the G gets better with training.

The basic GAN architecture is illustrated in Figure 2.

Figure 2 Basic GAN Architecture



In the basic GAN framework, Kullback-Leibler (KL) and Jensen-Shannon (JS) divergences are utilized to derive the loss function. During training, the model uses cross-entropy loss for reducing the discrepancy between the generated and real data distributions, which efficiently corresponds to reducing the KL-JS divergence. The discriminator's goal is to increase the likelihood of correctly labelling every sample. The mathematical objective function $\hat{V}$ for the discriminator is given by:

$$\hat{V} = \frac{1}{n}\sum_{j=1}^{n} log\, D(y^j) + \sum_{j=1}^{n}\left(1 - log\, D\left(D(x^j)\right)\right) \quad (1)$$

Then, the generator is trained for minimizing its objective function, which is given by,

$$\hat{V} = \frac{1}{n}\sum_{j=1}^{n}\left(1 - log\, D\left(D(x^j)\right)\right) \quad (2)$$

where x and y represent the generator's input data and the matching target from the actual dataset, respectively. The generator's output is denoted by $G(x_j)$. To demonstrate the calculation during a GAN's training phase, the discriminator's loss function is provided by,

$$-\frac{1}{n}\sum_{j=1}^{n} log\, D(y^j) - \sum_{j=1}^{n}\left(1 - log\, D\left(D(x^j)\right)\right) \quad (3)$$

The loss function of generator is given by,

$$\sum_{j=1}^{n}\left(log\, D\left(D(x^j)\right)\right) \quad (4)$$

Through the training process, the loss function should be minimized to get the better result.

In the proposed model, AttBiGRU is used as CNN is used as D to distinguish between the generated and real stock prices, while G enters the historical stock price and produces the future stock price.

3.3 AttBiGRU as Generator

GRU is a variation of traditional RNN, which can learn short as well as long term dependencies. GRU has two layers layer 1 and layer 2. There are forward and backward GRUs in every stratum, and each GRU comprises a self-connected hidden layer and an input layer.

In BiGRU, there are 256 secret units and one GRU unit is initialized with zero-valued states. The forward GRU processes the sequence X = (X_1, X_2,..., X_{18}), as a set of sequential inputs. The forward GRU's input layer receives the t^{th} vector x_t from sequence X at each time step t ($0< t \leq 18$). x_t and the prior hidden state h_{t-1} are combined in the hidden layer for performing a set of linear transformations which is followed by activation functions.

Each GRU cell consists of 2 gates: the reset gate (r_t) and the update gate (z_t), where z_t controls the amount of the previous state should be preserved in the present situation, whereas the r_t controls how much historical data should be overlooked efficiently filtering out unrelated byte sequences. It will ensure that only the meaningful sequences are stored and passed to the following time steps. The complete relationships that handle the forward GRU operations are defined as:

$$z_t = \sigma\left(W_z\left[\overrightarrow{h_{t-1}}, x_t\right]\right) \quad (5)$$

$$r_t = \sigma\left(W_r\left[\overrightarrow{h_{t-1}}, x_t\right]\right) \quad (6)$$

$$\tilde{h}_t = tanh\left(W\left[r_t\overrightarrow{h_{t-1}}, x_t\right]\right) \quad (7)$$

$$\vec{h}_t = (1 - z_t)\overrightarrow{h_{t-1}} + z_t\tilde{h}_t \quad (8)$$

While σ denotes the sigmoid activation function that converts the intermediate state into the 0–1 range to operate as a gating signal, W_z, W_r, and W stand for weight matrices. At t, the hidden state is denoted by h_t. The input for the backward GRU is X in reverse. A hidden state h_t is produced at t by a sequence of computations akin to the forward GRU, and it is spliced with h_t to produce the final output h_t.

H = (h_1, h_2,..., h_{18}) of layer 1 is the final output following the last time step. The BiGRU approach uses forward and backward propagation to precisely extract features by analyzing the correlations of packet vectors. In order to capture deeper information as layer 1, H is passed into layer 2. The attention layer uses the output of layer 2 as its input.

$$h_t = \left[\overrightarrow{h_t} + \overleftarrow{h_t}\right] \quad (9)$$

It is vital to allocate greater attention to the more informative vectors. For every hidden state h_t generated a_t by the second BiGRU layer, a weight is learned by the attention layer t ranges from 1 to 18 and the attention weights are characterized dependent on H, as a vector a = (a_1, a_2,..., a_{18}). A weighted aggregate of these concealed states is used to determine the final attention vector, s. It is given by,

$$F = \sum_{t=1}^{18} a_t h_t \quad (10)$$

$$a_t = \frac{exp(k_t^T k_w)}{\sum_t exp(k_t^T k_w)} \quad (11)$$

$$u_t = tanh(W_w h_t + b_w) \quad (12)$$

.

where b_w is the bias and W_w and k_w are weight matrices. A fully connected layer receives the attention layer's output and uses a c-way softmax function is used for producing a probability distribution over c traffic class labels in the dataset.

3.4 CNN as Discriminator

The discriminator is a CNN model intended to ascertain whether its input data is authentic or fraudulent. The original dataset or the generator's output is fed to the discriminator. This architecture has 3 layers of 1D convolution with 32, 64, and 128 filters, which is afterward, three dense layers having 220, 220, and 1 neuron, respectively. All layers utilize the Leaky ReLU as the activation function, with the exception of the output layer. The layer of output in the Basic GAN setup utilizes a function of sigmoid activation to generate a scalar value between 0 and 1, representing whether the input is real or fake. In the WGAN-GP version, the output layer uses a linear activation function that generates a continuous scalar score.

3.5 Hyperparameter tuning using WaOA

In machine learning, a hyperparameter is a parameter whose value guides the learning process. However, training is employed to ascertain the values of node weights. Hyperparameters can be considered as algorithmic hyperparameters, which impact the speed and superiority of learning. The performance of the resulting model can be greatly impacted by the selection of hyper-parameters, although choosing appropriate values can be challenging.

Choosing the appropriate collection of values while creating a ML model is known as hyperparameter tuning. Tuning process with optimization requires a term to be minimized.

In this work, the following hyperparameters of G and D of GAN are optimized by applying the WaOA.

- ✓ Learning rate (LR) of both G and D
- ✓ Batch size (BS) of both G and D
- ✓ Number of units (NU) in G.

This mechanism aids in refining the prediction accuracy by emphasizing the most relevant features from historical data, thereby enriching the predictive capabilities of the model.

3.5.1 Mathematical Model of WaOA

In WaOA, the position updating mechanism is inspired by the natural behaviours of walruses and is classified into three main phases. The initial phase concentrates on the feeding strategy that represents the exploration stage. Walruses intake a wide range of marine organisms—more than sixty species. Though, they favour benthic bivalve mollusks, mainly clams. Walruses feed on the seafloor using strong flipper motions and keen vibrissae to identify potential prey. The dominating walrus in this process, which is usually distinguished by its long tusks, guides the group in the search for food.

The walrus's tusk length is a representation of the candidate solution's objective function value quality. Henceforth, the best-performing The strongest walrus is thought to be the candidate answer. This leader-based search behaviour inspires exploration across numerous regions of the solution space, thus enhancing the global search capability. The position update phase is represented by copying this food supplying strategy in which the best candidate leads the generation of new positions. When a newly

generated position produces a better objective function (OF) value, it substitutes the earlier one. This update mechanism is captured in the following equations.

$$x_{mn}^{P1} = x_{mn} + rand_{mn}(SW_n - I_{mn}x_{mn}) \quad (13)$$

$$X_m = \begin{cases} X_m^{P1} & F_m^{P1} < F_m \\ X_m & Else \end{cases} \quad (14)$$

where X_m^{P1} is the newly generated position of walrus m in Phase 1, whereas x_{mn}^{P1} indicates its n-th dimension, with the value of its objective function being F_m^P1. The term $rand_{mn}$ are randomly generated numbers within [0,1]. The top-performing potential solution is SW. This solution is the most powerful walrus in the group. The integer I_{mn} is randomly chosen as 1 or 2. It will improve the exploratory capability of the algorithm. When I_{mn} = 2, it activates more considerable and extensive positional changes in the walrus than the standard case of I_{mn} = 1. This design selection reinforces the algorithm's capability to escape local optima and enables exploration of the true global optimum in the search space.

Phase 2-Migration: The main distinctive the way walruses behave is their seasonal migration to rocky shorelines or outcrops characteristically prompted by increasing temperatures in late summer.
This migratory behaviour is combined with WaOA for guiding the search process toward successful parts of the solution space. This behaviour is formulated in the following equations.
In this model, each walrus moves toward where another walrus selected at random is located in a diverse region of the area under search. Firstly, a new candidate place is generated. Then, this new location replaces the corresponding walrus's old position when it results in an enhanced objective function value.

$$x_{mn}^{P2} = \begin{cases} x_{mn} + rand_{mn}(x_{kn} - I_{mn}x_{mn}) & F_k < F_n \\ x_{mn} + rand_{mn}(x_{mn} - x_{kn}) & Else \end{cases} \quad (15)$$

$$X_m = \begin{cases} X_m^{P2} & F_m^{P2} < F_m \\ X_m & Else \end{cases} \quad (16)$$

where X_m^{P2} is the newly generated position of walrus m in Phase 2 and its n-th dimension is x_{mn}^{P2}. F_m^{P2} matches the value of OF at this new position X_k, where k ≠ I and k ∈ {1, 2,..., N} specifies the position of another randomly selected walrustoward which the migration of the mth walrus and x_{kn} indicates its j-th dimension. F_k is the related objective function value of the k-th walrus.

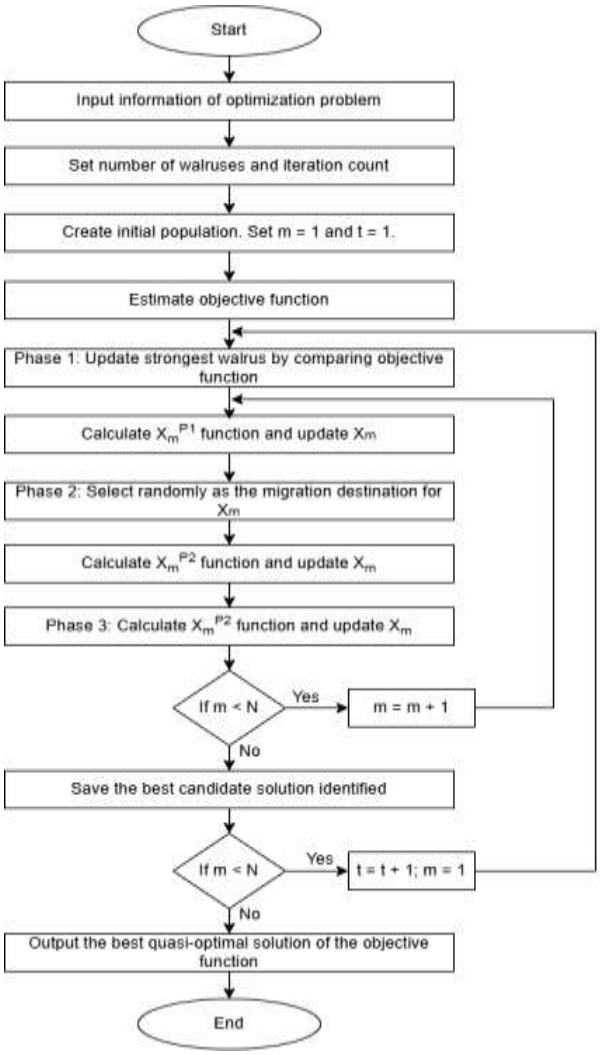
Phase 3-Using Exploitation to Flee and Fight Predators: Usually, walruses are susceptible to predators like polar bears and killer whales. Their natural response—fleeing or defending themselves—leads to rapid movements in the neighbourhood of their current location. This behaviour is matched in WaOA for strengthening its exploitation capability during local search, concentrating on refining candidate solutions in their immediate neighbourhood. As these actions happen near current position of every walrus, the algorithm models it within a neighbourhood based on walruses, as indicated by a dynamic radius. In the early iterations of WaOA, the concentration is on global exploration. Therefore, the neighbourhood radius is primarily set to a larger value and progressively reduced in later stages for improving precision.
For simulating this behaviour, local lower and upper bounds are introduced that dynamically adjust with each iteration, thus modifying the neighbourhood radius. A new position is dynamically determined within this neighbourhood. When this new position leads to an improved OF, it substitutes the earlier position based on the condition specified below.

$$x_{mn}^{P3} = \left\{x_{mn} + \left(L_{ln}^t + (U_{ln}^t - rand \cdot L_{ln}^t)\right), Local\ bound = \begin{cases} L_{ln}^t = \frac{L_n}{t} \\ U_{ln}^t = \frac{U_n}{t} \end{cases}\right. \quad (17)$$

$$X_m = \begin{cases} X_m^{P3} & F_m^{P3} < F_m \\ X_m & Else \end{cases} \quad (18)$$

where L_n and U_n represent the lower and upper thresholds of the nth term, L_{ln}^t and U_{ln}^t are local lower and local upper thresholds acceptable for the nth term, respectively, $x_{mn}^{P3}$is its nth dimension; F_m^{P3} is its OF value; and t is the iteration contour.
Figure 2 displays the suggested flowchart WaOA approach.

Figure 3 Flowchart of WaOA Process



Algorithm: Pseudocode of WaOA

1. Get WaOA started.
2. Enter all optimization problem details.
3. Decide on the total number of iterations (T) and the number of walruses (N).
4. Set up walruses' locations.
5. For t = 1:T
6. Update strongest walrus depending on objective function value criterion.
7. For m = 1:N

Phase 1: Strategy for feeding

Estimate new where the nth walrus is located

Update the m-th walrus location

Phase Two: Transition

10. Choose the m-th walrus's immigration destination.
11. Determine the n-th walrus's new location. Update the m^{th} walrus location.

//Phase 3: Escaping and fighting against predators//

Determine a new role in the neighbourhood of the m^{th} walrus

Update the m^{th} walrus location

14. End
15. Keep the top-ranked potential solution.
16. End
17. Provide the best quasi-optimal solution for the given problem that WaOA was able to obtain.
18. End

---

4. Experimental Results

The proposed GAN-AttBIGRU-CNN model is implemented in Python 3.The following datasets are used in the experiments:

- **Nikkei 225 index [22]:** It is most widely used in Japanese stock markets.
- **Chinese Stock Index (CSI) 300 [23]**: This dataset includes 300 stocks and 5,088 time steps from the CSMAR database.
- **Hang Seng Index (HSI) [24]:** The historical records of the HSI dataset start from January 2000 to July 2015.
- **S&P index 500 [25]:** The historical stock prices of last 5 years for all companies are provided in the S&P 500 index dataset.

The proposed GAN:AttBIGRU-CNN model is compared with the existing BiGRU, CNN, GAN-BiGRU, GAN-CNN models.

4.1 Training & validation Curves

This section presents the Training & Validation (TV) curves of the proposed model for the 4 datasets.

A. Results for Nikkei 225 dataset

Figure 4 TV loss curves for Nikki 225 dataset

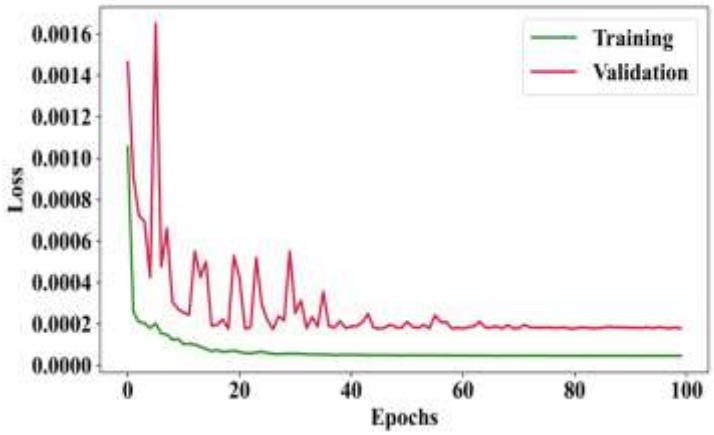


Figure 4 shows the TV loss curves of the proposed GAN-AttBiGRU-CNN model for the Nikki 225 dataset, by varying the epochs from 0 to 100.

Figure 5 TV accuracy curves for Nikki 225 dataset

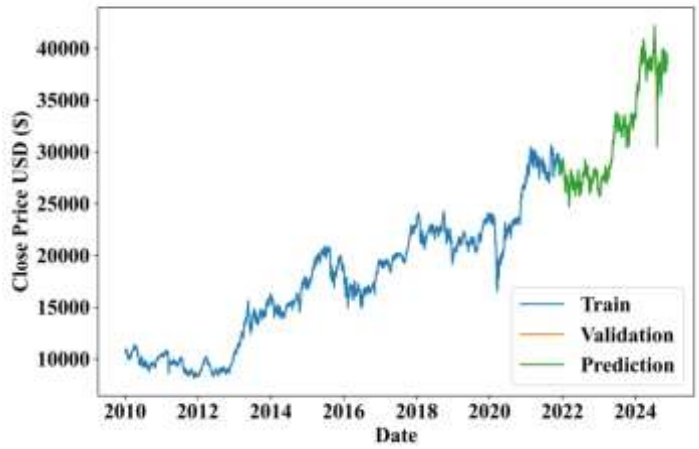


Figure 5 shows the TV accuracy curves of the proposed GAN-AttBiGRU-CNN model for the Nikki 225 dataset by varying the years from 2010-2014.

Figure 6 Comparison of TV accuracy curves Nikki 225 dataset

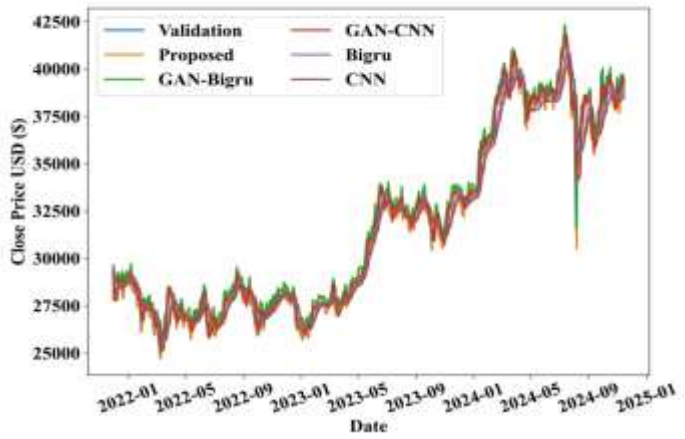


Figure 6 shows the TV accuracy curves of the proposed and existing models for the Nikki 225 dataset by varying the years from 2010-2014.

4.2 Results for CSI 300 dataset

Figure 7 TV loss curves for CSI 300 dataset

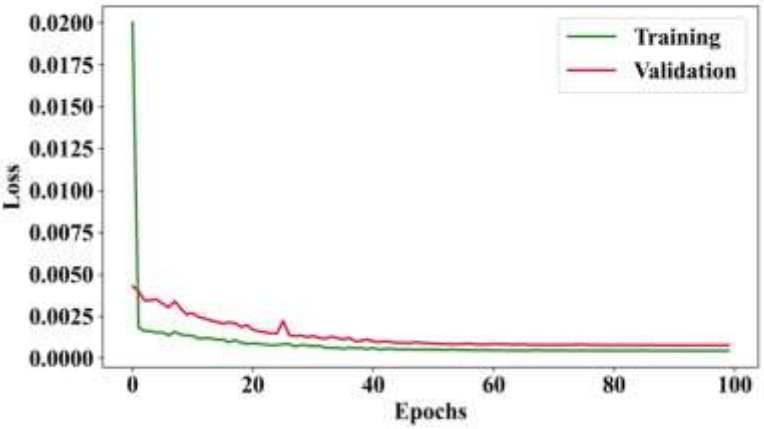


Figure 7 shows the TV loss curves of the proposed GAN-AttBiGRU-CNN model for the CSI 300 dataset, by varying the epochs from 0 to 100.

Figure 8 TV accuracy curves for CSI 300 dataset

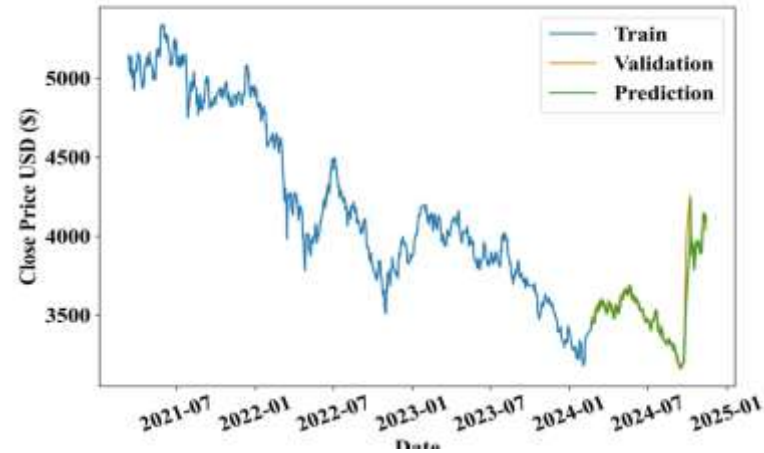


Figure 8 shows the TV accuracy curves of the proposed GAN-AttBiGRU-CNN model for the CSI 300 dataset by varying the years from 2010-2014.

Figure 9 Comparison of TV accuracy curves for CSI 300 dataset

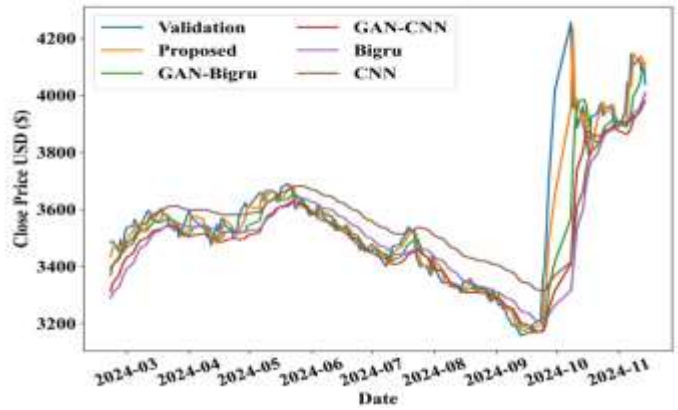


Figure 9 shows the TV accuracy curves of the proposed and existing models for the CSI 300 dataset by varying the years from 2010-2014.

4.3 Results for HSI dataset

Figure 10 TV loss curves for HSI dataset

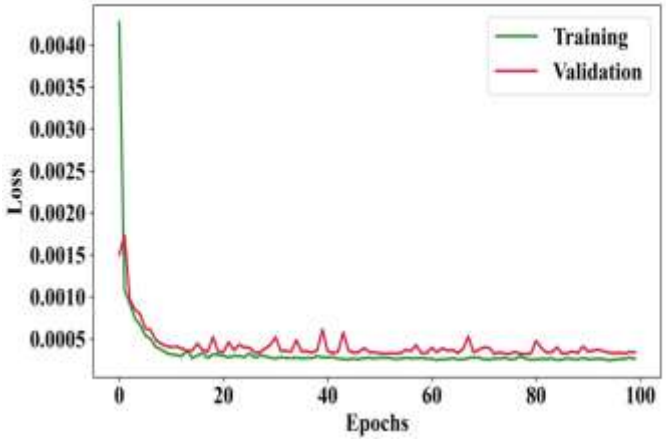


Figure 10 shows the TV loss curves of the proposed GAN-AttBiGRU-CNN model for the HSI dataset, by varying the epochs from 0 to 100.

Figure 11 TV accuracy curves for HSI dataset

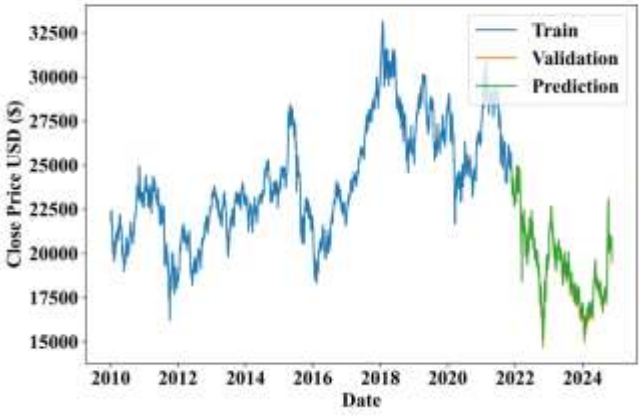


Figure 11 shows the TV accuracy curves of the proposed GAN-AttBiGRU-CNN model for the HSI dataset by varying the years from 2010-2014.

Figure 12 Comparison of TV accuracy curves for HSI dataset

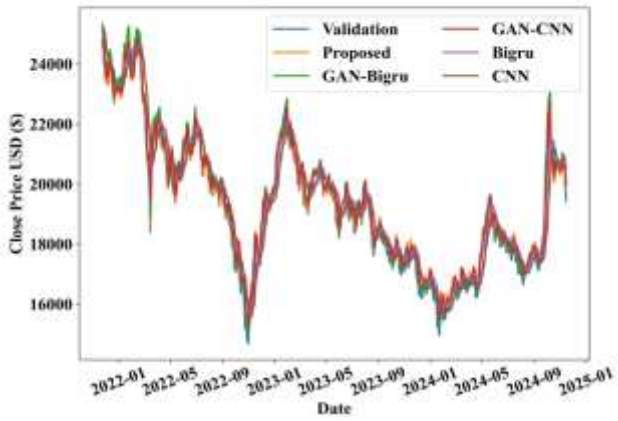


Figure 12 shows the TV accuracy curves of the proposed and existing models for the HSI dataset by varying the years from 2010-2014.

4.4 Results for S&P 500 index dataset

Figure 13 TV loss curves for S&P 500 dataset

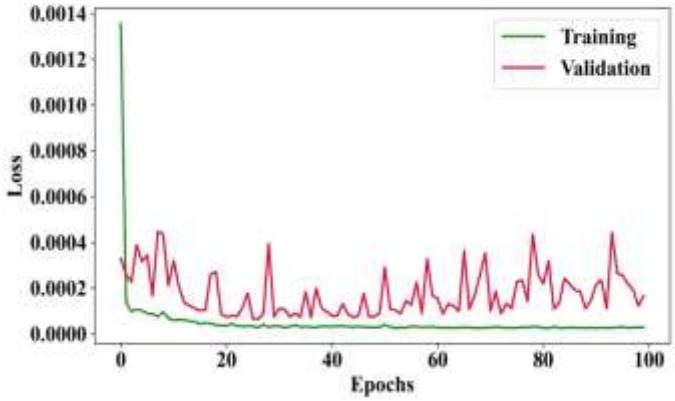


Figure 13 shows the TV loss curves of the proposed GAN-AttBiGRU-CNN model for the S&P 500 dataset, by varying the epochs from 0 to 100.

Figure 14 TV accuracy curves for S&P 500 dataset

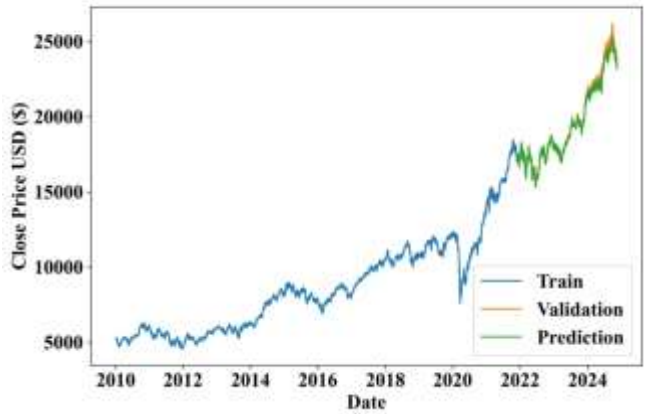


Figure 14 shows the TV accuracy curves of the proposed GAN-AttBiGRU-CNN model for the S&P 500 dataset by varying the years from 2010-2014.

Figure 15 Comparison of TV accuracy curves for S&P 500 dataset

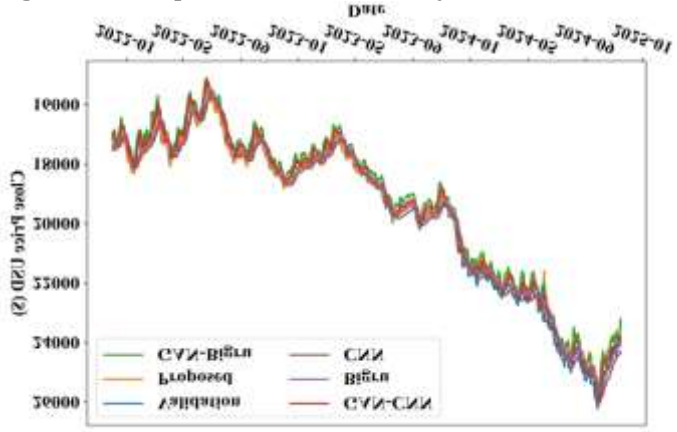


Figure 15 shows the TV accuracy curves of the proposed and existing models for the S&P 500 dataset by varying the years from 2010-2014.

4.5 Performance Comparison

In this section, the accuracy of models are measured for the 4 datasets. The model accuracy is represented in terms of the statistical error measures, which are defined in Table 1.

Table 1 Statistical error measures for model accuracy

Measure	Definition	Formula
Mean Square Error(MSE)	It is computed by averaging the squared residuals, which is the difference between each data point's actual value and its anticipated value.	$\text{Mean Squared Error} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$
Root Mean Square Error(RMSE)	The discrepancy between the measured and expected values are squared and then averaged over the sample for which, the square root is taken.	$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{n}(A_t - F_t)^2}{n}}$
Mean Absolute Percentage Error (MAPE)	percentage error of the difference between the actual and estimated values .	$\text{MAPE} = \frac{\sum_{t=1}^{n}\left\lvert\frac{A_t - F_t}{A_t}\right\rvert}{n} \times 100$
R2	It is a statistical measure that represents the proportion of variance in the dependent variable	$R^2 = 1 - (SSE / SST)$ SSE- Sum of squared Error SST – Total sum of Squares

Figure 16 Comparison of MAPE results for all models

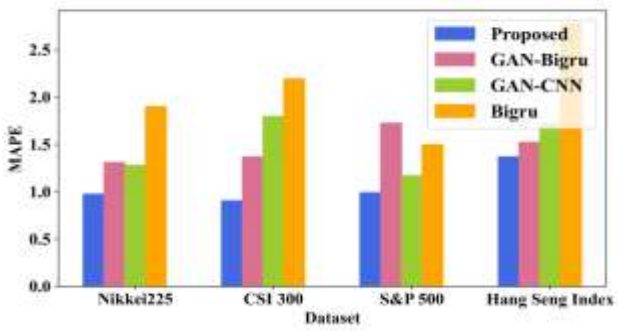


Figure 16 shows the results of MAPE measured over the 4 datasets for the existing and proposed models. As we can see from the figure, the proposed GAN-AttBIGRU-CNN model attains the least MAPE values for all datasets, than the other models. Table 2 shows the percentage of improvement of GAN-AttBIGRU-CNN over the other models.

Table 2 Percentage of improvement of proposed over existing models for MAPE

Dataset	GAN-BiGRU (%)	GAN-CNN (%)	BiGRU (%)	CNN (%)
Nikkei225	25.6	23.8	48.7	54.9
CSI 300	33.8	49.6	58.7	66.4
S&P 500	42.6	15.2	33.9	33.5
HIS	10.1	20.4	93.9	49.8
Avg	**28.0**	**27.3**	**58.8**	**51.1**

Figure 17 Comparison of MSE results for all models

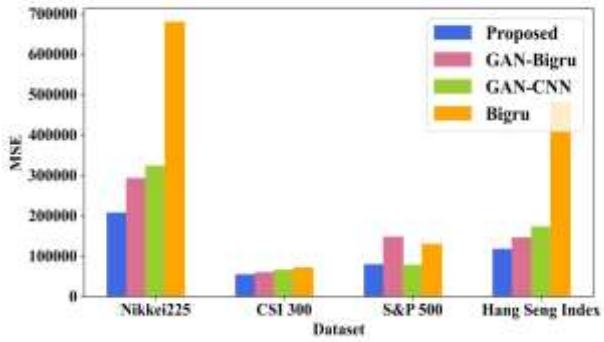


Figure 17 shows the results of MSE measured over the 4 datasets for the existing and proposed models. As we can see from the figure, the proposed GAN-AttBIGRU-CNN model attains the least MSE values for all datasets, when compared to the other models. Table 3 shows the percentage of improvement GAN-AttBIGRU-CNN over the other models.

Table 3 Percentage of improvement of proposed over existing models for MSE

Dataset	GAN-BiGRU (%)	GAN-CNN (%)	BiGRU (%)	CNN (%)
Nikkei225	29.3	36.0	69.6	75.9
CSI 300	9.5	17.7	24.1	22.8
S&P 500	46.5	3.1	39.0	38.8
HIS	19.6	32.1	75.5	76.8
Avg	**26.2**	**22.2**	**52.1**	**53.6**

Figure 18 Comparison of RMSE results for all models

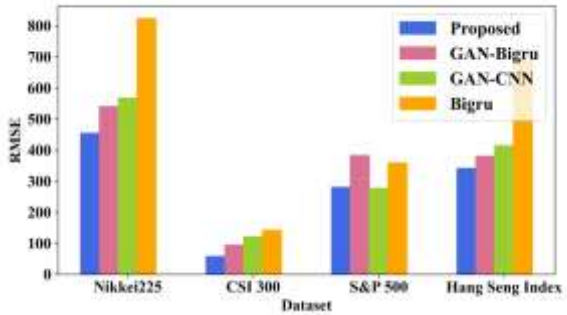


Figure 18 shows the results of RMSE measured over the 4 datasets for the existing and proposed models. As we can see from the figure, the proposed GAN:AttBIGRU-CNN model attains the least RMSE values for

all datasets, than the other models. Table 4 shows the percentage of improvement of GAN-AttBIGRU-CNN over the other models.

Table 4 Percentage of improvement of proposed over existing models for RMSE

Dataset	GAN-BiGRU (%)	GAN-CNN (%)	BiGRU (%)	CNN (%)
Nikkei225	15.9	20.0	44.9	50.9
CSI 300	38.5	52.3	59.2	57.9
S&P 500	26.9	5.8	21.9	21.8
HIS	10.3	17.6	50.7	51.8
Avg	**22.9**	**23.9**	**44.2**	**45.6**

Figure 19 Comparison of R2 results for all models

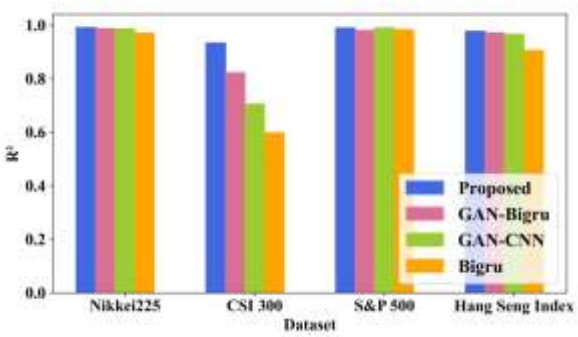


Figure 19 shows the results of R2 values measured over the 4 datasets for the existing and proposed models. It essentially denotes how the model's predictions match the actual data. As we can see from the figure, the proposed GAN:AttBIGRU-CNN model attains the highest R2 values for all datasets, than the other models.
Table 5 shows the percentage of improvement of GAN-AttBIGRU-CNN over the other models for R2

Table 5 Percentage of improvement of proposed over existing models for R2

Dataset	GAN-BiGRU (%)	GAN-CNN (%)	BiGRU (%)	CNN (%)
Nikkei225	0.37	0.51	2.06	2.83
CSI 300	11.83	24.39	35.92	33.46
S&P 500	0.90	0.99	0.66	0.65
HIS	0.57	1.11	7.32	7.76
Avg	**3.42**	**6.75**	**11.49**	**11.18**

5. Conclusion

This study proposes a SPP model using optimized GAN. It consists of AttBiGRU model as a generator and CNN model as a discriminator. The hyperparameters of GAN are optimized by applying the WaOA. The proposed GAN:AttBIGRU-CNN model has been applied over 4 bench mark datasets and its performance has been compared with the existing BiGRU, CNN, GAN-BiGRU, GAN-CNN models. Experimental results state that GAN:AttBIGRU-CNN model attains minimized MSE , RMSE, MAPE and maximum R^2 measures, while predicting the stock prices, when compared to the existing models.

References

[1] Mehtab, S., Sen, J., & Dutta, A. (2021). Stock price prediction using machine learning and LSTM-based deep learning models. In P. Siarry (Ed.), Machine learning and metaheuristics algorithms, and applications: Second Symposium, SoMMA 2020, Chennai, India, October 14–17, 2020, revised selected papers 2 (pp. 88–106). Springer. https://doi.org/10.1007/978-981-16-1811-0_6

[2] Mintarya, L. N., Halim, J. N. M., Angie, C., Achmad, S., & Kurniawan, A. (2023). Machine learning approaches in stock market prediction: A systematic literature review. Procedia Computer Science, 216, 96–102. https://doi.org/10.1016/j.procs.2022.12.060
[3] Abdullah, M., Sulong, Z., & Chowdhury, M. A. F. (2024). Explainable deep learning model for stock price forecasting using textual analysis. Expert Systems with Applications, 249, 123740. https://doi.org/10.1016/j.eswa.2023.123740
[4] Mohanty, D. K., Parida, A. K., & Khuntia, S. S. (2021). Financial market prediction under deep learning framework using auto encoder and kernel extreme learning machine. Applied Soft Computing, 99, 106898. https://doi.org/10.1016/j.asoc.2020.106898
[5] Khan, W., Ghazanfar, M. A., Azam, M. A., Karami, A., Alyoubi, K. H., & Alfakeeh, A. S. (2022). Stock market prediction using machine learning classifiers and social media, news. Journal of Ambient Intelligence and Humanized Computing, 1–24. https://doi.org/10.1007/s12652-022-03792-8
[6] Swathi, T., Kasiviswanath, N., & Rao, A. A. (2022). An optimal deep learning-based LSTM for stock price prediction using twitter sentiment analysis. Applied Intelligence, 52(12), 13675–13688. https://doi.org/10.1007/s10489-021-02907-4
[7] Ni, J., & Xu, Y. (2023). Forecasting the dynamic correlation of stock indices based on deep learning method. Computational Economics, 61(1), 35–55. https://doi.org/10.1007/s10614-021-10208-2
[8] Chen, M.-Y., Chiang, H.-S., Lughofer, E., & Egrioglu, E. (2020). Deep learning: Emerging trends, applications and research challenges. Soft Computing, 24, 7835–7838. https://doi.org/10.1007/s00500-020-04811-0
[9] Pokhrel, N. R., Dahal, K. R., Rimal, R., Bhandari, H. N., Khatri, R. K. C., Rimal, B., & Hahn, W. E. (2022). Predicting NEPSE index price using deep learning models. Machine Learning with Applications, 9, 100385. https://doi.org/10.1016/j.mlwa.2022.100385
[10] Kanwal, A., Lau, M. F., Ng, S. P. H., Sim, K. Y., & Chandrasekaran, S. (2022). BiCuDNNLSTM-1dCNN—A hybrid deep learning-based predictive model for stock price prediction. Expert Systems with Applications, 202, 117123. https://doi.org/10.1016/j.eswa.2022.117123
[11] Singh, T., Kalra, R., Mishra, S., Satakshi, & Kumar, M. (2023). An efficient real-time stock prediction exploiting incremental learning and deep learning. Evolving Systems, 14(6), 919–937. https://doi.org/10.1007/s12530-023-09566-2
[12] Shen, J., & Shafiq, M. O. (2020). Short-term stock market price trend prediction using a comprehensive deep learning system. Journal of Big Data, 7, 1–33. https://doi.org/10.1186/s40537-020-00299-6
[13] Singh, R., & Srivastava, S. (2017). Stock prediction using deep learning. Multimedia Tools and Applications, 76, 18569–18584. https://doi.org/10.1007/s11042-016-4159-7
[14] Liu, X., Guo, J., Wang, H., & Zhang, F. (2022). Prediction of stock market index based on ISSA-BP neural network. Expert Systems with Applications, 204, 117604. https://doi.org/10.1016/j.eswa.2022.117604
[15] Wang, C., Chen, Y., Zhang, S., & Zhang, Q. (2022). Stock market index prediction using deep Transformer model. Expert Systems with Applications, 208, 118128. https://doi.org/10.1016/j.eswa.2022.118128
[16] Rezaei, H., Faaljou, H., & Mansourfar, G. (2021). Stock price prediction using deep learning and frequency decomposition. Expert Systems with Applications, 169, 114332. https://doi.org/10.1016/j.eswa.2020.114332
[17] Gülmez, B. (2023). Stock price prediction with optimized deep LSTM network with artificial rabbits optimization algorithm. Expert Systems with Applications, 227, 120346. https://doi.org/10.1016/j.eswa.2023.120346
[18] Zhao, Y., & Yang, G. (2023). Deep learning-based integrated framework for stock price movement prediction. Applied Soft Computing, 133, 109921. https://doi.org/10.1016/j.asoc.2023.109921
[19] Jiang, J., Wu, L., Zhao, H., Zhu, H., & Zhang, W. (2023). Forecasting movements of stock time series based on hidden state guided deep learning approach. Information Processing & Management, 60(3), 103328. https://doi.org/10.1016/j.ipm.2023.103328
[20] Venkateswararao, K., & Reddy, B. V. R. (2023). LT-SMF: Long term stock market price trend prediction using optimal hybrid machine learning technique. Artificial Intelligence Review, 56(6), 5365–5402. https://doi.org/10.1007/s10462-023-10542-0
[21] Sivadasan, E. T., Sundaram, N. M., & Santhosh, R. (2024). Stock market forecasting using deep learning with long short-term memory and gated recurrent unit. Soft Computing, 28(4), 3267–3282. https://doi.org/10.1007/s00500-023-08788-1

[22] Trojovský, P., & Dehghani, M. (2023). A new bio-inspired metaheuristic algorithm for solving optimization problems based on walruses behaviour. Scientific Reports, 13, 8775. https://doi.org/10.1038/s41598-023-35863-5
[23] Nikkei Inc. (n.d.). Nikkei stock average dataset. Nikkei Indexes. https://nkbb.nikkei.co.jp/en/dataset/
[24] Liqiang, L. (2022). CSI 300 index fully data 2017–2022. Kaggle. https://www.kaggle.com/datasets/liqiang2022/csi-300-index-fully-data-20172022
[25] Hang Seng Indexes Company Limited. (n.d.). Hang Seng Index. https://www.hsi.com.hk/eng
[26] Nugent, C. (2017). S&P 500 stock data. Kaggle. https://www.kaggle.com/datasets/camnugent/sandp500