Journal of Scientific Research and Reports

Volume 30, Issue 10, Page 1-11, 2024; Article no.JSRR.123063
ISSN: 2320-0227

Enhancing Agricultural Commodity Price Forecasting Using Generative Models: A Deep Learning Approach

G. Avinash [a], G. H. Harish Nayak [b], Moumita Baishya [a*], B. Samuel Naik [a] and Karthik V.C. [a]

[a] The Graduate School, ICAR-Indian Agricultural Research Institute, New Delhi, India.
[b] Department of Agricultural Statistics, College of Agriculture, UAS, Dharwad, India.

Authors' contributions

This work was carried out in collaboration among all authors. All authors read and approved the final manuscript.

Article Information

DOI: https://doi.org/10.9734/jsrr/2024/v30i102424

Open Peer Review History:
This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here: https://www.sdiarticle5.com/review-history/123063

Original Research Article

Received: 02/07/2024
Accepted: 04/09/2024
Published: 16/09/2024

ABSTRACT

Predicting stock market prices is a critical yet challenging task in finance. Technical analysis, a widely used methodology in investment theory, involves forecasting price movements by analyzing historical market data. Recently, deep learning has gained prominence for its exceptional ability to process complex data, making it a popular tool for financial applications such as stock prediction, portfolio optimization, financial information analysis, and trade execution strategies. In this study, we propose a novel deep learning architecture that integrates a Generative Adversarial Network (GAN) with a Convolutional Neural Network (CNN) as the discriminator and Gated Recurrent Units (GRU) as the generator. This framework generates distributions of daily stock prices through adversarial

**Corresponding author: E-mail: moumitabaishya1194@gmail.com;*

Cite as: *Avinash, G., G. H. Harish Nayak, Moumita Baishya, B. Samuel Naik, and Karthik V.C. 2024. "Enhancing Agricultural Commodity Price Forecasting Using Generative Models: A Deep Learning Approach". Journal of Scientific Research and Reports 30 (10):1-11. https://doi.org/10.9734/jsrr/2024/v30i102424.*

learning to predict stock closing prices. Using daily trading data from Ruchi Soya Industries Limited, an empirical analysis was performed across a broad time frame. Results show that the proposed GAN-based architecture significantly outperforms traditional deep learning models, including GRU, LSTM, and Bi-LSTM, in predicting stock closing prices. These findings demonstrate the potential of this novel approach for improving stock price forecasting accuracy.

Keywords: Bidirectional LSTM (Bi-LSTM); Convolutional Neural Network (CNN); discriminator; Gated Recurrent Unit (GRU); generative models; generator; Long Short-Term Memory (LSTM).

1. INTRODUCTION

Predicting stock prices is a highly complex task due to the chaotic nature and intricate dynamics of financial markets, influenced by non-decidable, nonstationary stochastic variables (Marszalek and Burczynski, 2014). Various methods have been proposed to analyze historical financial time series, but achieving accurate forecasts often requires expert knowledge, precise input selection, and the application of advanced statistical methods. This complexity can be a barrier for individuals without financial expertise (Huang et al., 2014; Wang et al., 2017; Chong et al., 2017).

In recent years, machine learning (ML) and deep learning (DL) models have shown promising results in predicting agricultural prices for crops, as reported by Avinash et al. [1,2] (2023a, 2023b), Nayak et al. [3,4], Vinay et al. [5], Baishya et al. [6], and Singh et al. [7]. Other AI applications related to the several domains were also studied by Nayak et al. (2023) and Shah et al. [8]. Generative Adversarial Networks (GANs), introduced by Goodfellow et al. [9], have demonstrated success in generating image patches from random noise [10-13]. GANs consist of two networks: a discriminative network (D) that learns to distinguish between real and generated data, and a generative network (G) that aims to produce data indistinguishable from the real data. Although GANs have been applied to image processing and video prediction, their use in stock forecasting is relatively new, as seen in Iizuka et al. (2017). However, there has been no prior application of GANs to agricultural market prediction.

This study introduces a GAN-based model utilizing technical index data, which can be readily obtained from trading platforms, allowing individuals without financial expertise to predict stock prices. This approach tackles challenges in deep generative models, such as intractable probabilistic computations arising from maximum likelihood estimation and the use of piecewise linear units in a generative context [14,15]. The GAN framework drives improvement through competition between the generative model and the discriminative model, where the former attempts to generate realistic data while the latter seeks to detect the generated data as fake [16-18]. This adversarial process continues until the generated data becomes indistinguishable from real data.

The paper is organized as follows: Section 2 reviews the literature on financial market prediction algorithms. Section 3 outlines the GAN framework and problem formulation. The experimental section compares the proposed model's performance with classical prediction models. Sections 4 and 5 present the results, conclusions, and references.

2. MATERIALS AND METHODS

2.1 Genesis of Gan

Related work falls into two categories: econometric and soft computing models. Econometric models like AR, MA, ARMA, and ARIMA (Brockwell & Devis, 2013) forecast by treating new signals as noisy combinations of recent signals but rely on assumptions like i.i.d. noise, which GARCH models address by predicting conditional variances. Soft computing models, inspired by AI, include ANN [19], FL (Hassan et al., 2009), SVM [20], and PSO [21]. Deep neural networks (Rather et al., 2015; Chong et al., 2017) effectively predict high-frequency financial time series, while Chen et al. (2017) use a double-layer network for stock return dependencies. These methods often need expert constraints, unlike the proposed model, which directly uses trading software data. Algorithms like RNN, LSTM, and GRU are widely used in time-series forecasting [22,23-25]. GANs, developed by Goodfellow et al. [9] for image generation, have been adapted for sequential data, improving stock price prediction through the adversarial generator-discriminator relationship. Some of these studies are summarized in Table 1.

Table 1. Review of modelling tasks using GAN models

Applications	Task	Values in task	Model	Evaluation methods
RCGAN (Esteban *et al.,* 2017)	Generation	Medical data	GAN and RNN	TSTR and TRTS
Grid-GAN [26]	Generation	Smart grid data	CGAN and CNN	TSTR and TRTS
EEG-GAN (Hartmann *et al.,* 2018)	Generation	EEG brain signals	WGAN and CNN	IS, FID, and ED
StockGAN (Zhou *et al.,* 2018)	Generation	Stock data	GAN, CNN and LSTM	RMSRE, DPA
GRU-GAN [27]	Imputation	Medical records, meteorologic data	GAN and GRU	Imputation accuracy
ForGAN (Koochali *et al.,* 2019)	Generation	Synthetic series and internet traffic	CGAN and LSTM	KL divergence
TimeGAN (Yoon *et al.,* 2019)	Generation	Sines, stocks, energy and events data	GAN	Diversity, fidelity (*e.g*., RMSRE, DPA)
E2GAN (Luo *et al.,* 2019)	Imputation	Medical records, meteorologic data	GAN and GRU	Imputation accuracy
SimGAN (Golany *et al.,* 2020)	Generation	Heart rate ECG signals	GAN	Prediction accuracy
Ad-Attack (Dang-Nhu *et al.,* 2020)	Generation	Stock prices and electricity data	GAN	Domain metrics (*e.g*., attack success rate, returned of perturbed portfolio)

2.2 Theoretical Background

In time series forecasting, conventional deep learning approaches such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Gated Recurrent Units (GRU), and Long Short-Term Memory (LSTM) models have been widely applied. This section compares these methods with Generative Adversarial Networks (GANs) for various time series applications.

Convolutional Neural Network (CNN): CNNs are deep, feed-forward neural networks commonly used for analyzing visual imagery. A CNN typically consists of an input layer, output layer, and multiple hidden layers. Unlike traditional Multilayer Perceptron (MLP) models, CNNs can develop internal representations of two-dimensional image data, making them effective for tasks involving spatial correlations. Though originally designed for image analysis, CNNs have been successfully adapted for time series forecasting by LeCun et al. (2015), proving effective in capturing temporal patterns in sequential data.

Recurrent Neural Network (RNN): RNNs are a type of neural network where previous outputs are fed as inputs to the current step. The key advantage of RNNs is their ability to capture sequential dependencies through hidden states, which serve as internal memory. Despite their success across various domains, RNNs suffer from the vanishing gradient problem [28], making it difficult to learn long-term dependencies in time series data. To address this issue, two variants of RNNs were developed: Long Short-Term Memory (LSTM) by Hochreiter and Schmidhuber (1997) and Gated Recurrent Units (GRU) by Kyunghyun et al. (2014). These models offer improved performance in learning temporal correlations in time series and spatio-temporal (ST) data.

Gated Recurrent Unit (GRU): GRU is a variant of RNN that incorporates gating mechanisms to control the flow of information. It was introduced in 2014 by Kyunghyun Cho et al. GRUs simplify the LSTM architecture by combining the forget and input gates into a single "update gate" and removing the cell state, storing both long- and short-term memory in the hidden state. GRUs address the vanishing and exploding gradient problems inherent in RNNs, and due to fewer parameters, they tend to perform better on smaller datasets compared to LSTMs. The internal structure of a GRU unit is shown in Fig. 1.

Long Short-Term Memory (LSTM): LSTM, introduced by Hochreiter et al. (1997), is a specific RNN architecture designed to address the vanishing gradient problem. LSTMs incorporate feedback connections and can handle both single-point and sequential data. LSTM units consist of three gates: input, output, and forget gates, which regulate the flow of information through the network. This enables LSTMs to remove or add information to the cell state, making them highly effective at learning long-term dependencies in time series data. LSTMs have become a powerful tool for processing, classifying, and forecasting time series data. The structure of an LSTM unit is shown in Fig. 2.

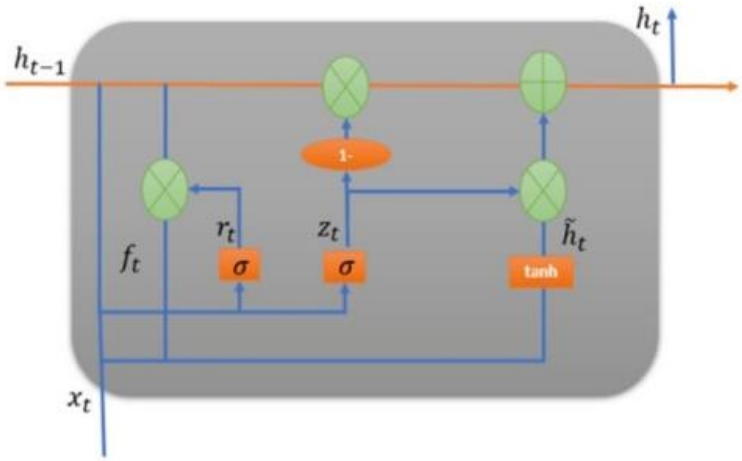

Fig. 1. Architecture of the GRU

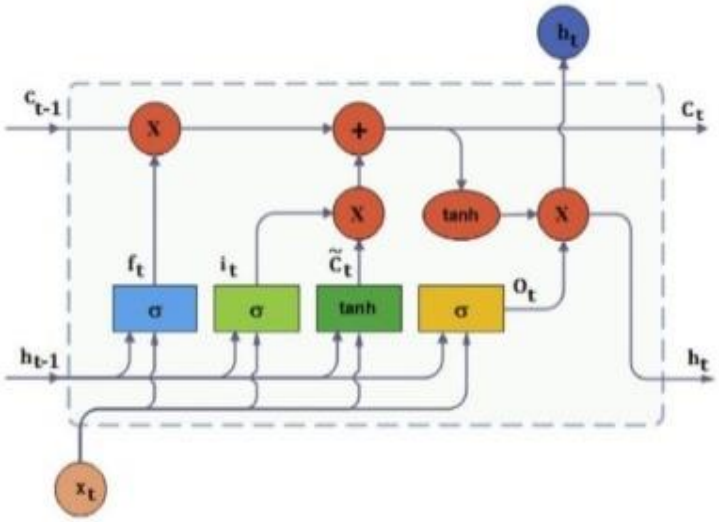

Fig. 2. Architecture of the LSTM

Bidirectional LSTM (Bi-LSTM): The Bidirectional LSTM (Bi-LSTM) model, as shown in Fig. 3, includes both a forward and a backward LSTM layer. The forward LSTM layer processes the input sequence from left to right, while the backward layer processes it from right to left. The hidden vectors of both layers, denoted as $\vec{h}_t$ and $\overleftarrow{h}_t$ are independent of each other and only relate to their respective layers. The final output of Bi-LSTM is obtained by combining the outputs of these two hidden layers through a weighted connection, as described by the following equations:

$$\vec{h}_t = LSTM\left(x_t, \vec{h}_{t-1}\right)$$
$$\overleftarrow{h}_t = \text{LSTM}\ \left(x_t, \overleftarrow{h}_{t+1}\right)$$
$$y_t = \delta\left(W_{\vec{h}_y}\vec{h}_t + W_{\overleftarrow{h}_t}\overleftarrow{h}_t + b_y\right)$$

2.3 Generative Adversarial Networks (GANs)

2.3.1 Basic idea/principles of GANs

GANs, introduced by Goodfellow et al. [9], operate on a framework that involves training two models in a zero-sum game setting. In this adversarial process, the generator (G) acts as the "cheater" by generating data that mimics real data, while the discriminator (D) serves as the "judge," distinguishing between real and generated data. The goal is to reach a point where the discriminator can no longer differentiate between the two, indicating that the generator has successfully captured the true data distribution. This principle underlies the proposed GAN architecture for predicting stock closing prices.

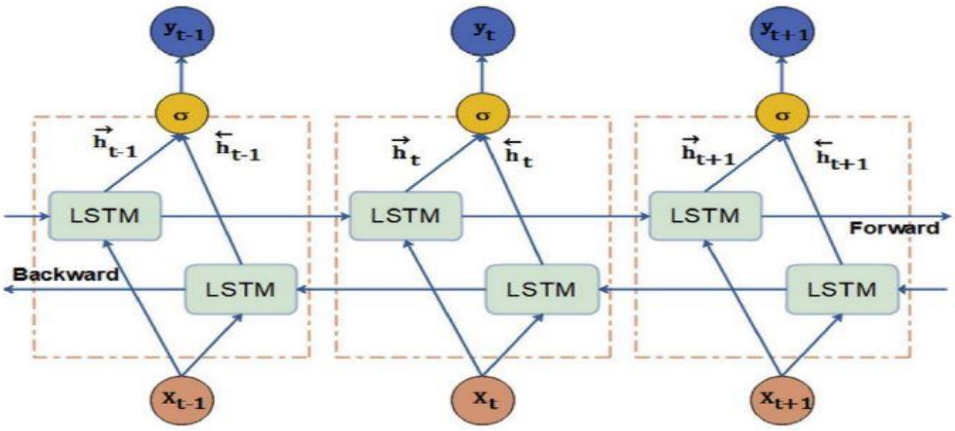

Fig. 3. Architecture of the Bi-LSTM

2.3.2 The architecture of GAN

"The adversarial modeling framework is most straightforward to apply when the models are both multilayer perceptron. To learn the generator's distribution p_g over data x, we define a prior on input noise variables $p_z(z)$, then represent a mapping to data space as G (z; θ_g), where G is a differentiable function represented by a multilayer perceptron with parameters θ_g. We also define a second multilayer perceptron D (x; θ_d) that outputs a single scalar. D(x) represents the probability that x came from the data rather than p_g. We train D to maximize the probability of assigning the correct label to both training examples and samples from G . We simultaneously train G to minimize log (1− D(G(z)))". In other words, D and G play the following two-player minimax game with value function V (G, D):

$$\min_G \max_D V(D, G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_{z(z)}}[\log(1 - D(G(z)))]$$

In practice, adversarial networks aim to optimize the discriminator and generator iteratively. While this game-theoretic approach ensures that the generator eventually learns the data distribution, implementing this game requires an iterative numerical method. Optimizing the discriminator fully within each training step is computationally expensive and can lead to overfitting, especially on finite datasets. Instead, we alternate between several steps of optimizing the discriminator and a single step of optimizing the generator. This maintains the discriminator near its optimal state while allowing the generator to evolve gradually as shown in Fig. 4 with Algorithm 1.

$L_D = -E_{x\sim P_{data}(x)}[\log D(x)] - E_{Z\sim P_{z(Z)}}[\log(1 - D(G(z)))]$ and $L_G = E_{Z\sim P_{z(Z)}}[\log(1 - D(G(z)))]$

In practice, above equation may not provide sufficient gradient for G to learn well. Early in learning, when G is poor, D can reject samples with high confidence because they are clearly different from the training data. In this case, log (1 − D(G(z))) saturates. Rather than training G to minimize log (1 − D(G(z))) we can train G to maximize log D(G(z)). This objective function results in the same fixed point of the dynamics of G and D but provides much stronger gradients early in learning. The generator G implicitly defines a probability distribution p_g as the distribution of the samples G(z) obtained when z ~ p_z. Therefore, we would like Algorithm 1 to converge to a good estimator of p_{data}, if given enough capacity and training time and finally this minimax game has a global optimum for p_g = p_{data}. Ultimately, the generator and discriminator are trained iteratively, continuously improving until the generated data is indistinguishable from the actual data, achieving the goal of accurate data prediction.

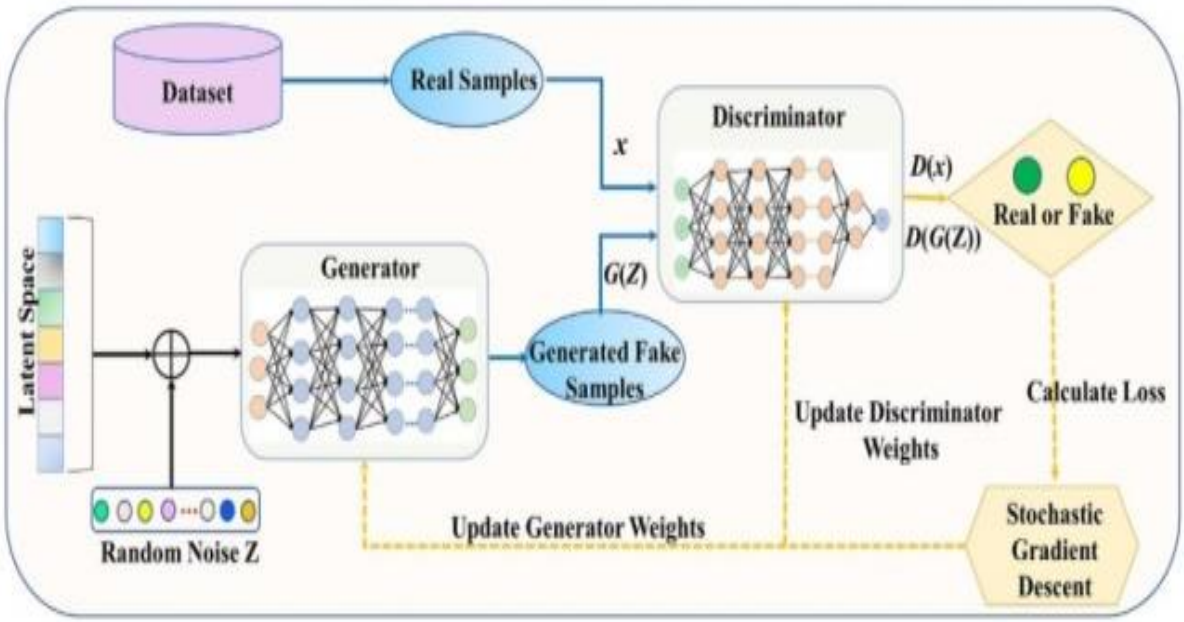

Fig. 4. Architecture of the GAN [29]

Algorithm 1 This algorithm outlines the training procedure for GANs using minibatch stochastic gradient descent. In this process, the generator (G) and discriminator (D) are updated iteratively to improve their performance in generating and distinguishing data, respectively. The number of steps applied to the discriminator, denoted by k, is a hyperparameter. In the experiments conducted, we used k=1, the least computationally expensive option, which still yields effective results.

for number of training iterations do

for k steps do

- Sample minibatch of m noise samples $\{z_{(1)}, \ldots, z_{(m)}\}$ from noise prior $p_g(z)$.
- Sample minibatch of m examples $\{x_{(1)}, \ldots, x_{(m)}\}$ from data generating distribution $p_{data}(x)$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} [\log D\left(x^{(i)}\right) + \log(1 - D(G(z^{(i)})))]$$

end for

- Sample minibatch of m noise samples $\{z_{(1)}, \ldots, z_{(m)}\}$ from noise prior $p_g(z)$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(1 - D(G(z^{(i)})))]$$

end for

The gradient-based updates can use any standard gradient-based learning rule.

3. RESULTS AND DISCUSSION

3.1 Empirical Study

Data description: The dataset used in this study comprises daily prices of soya oil (in INR) from July 30, 2010, to June 30, 2020, sourced from Yahoo Finance. The descriptive statistics of the soya stock price series, shown in Table 2, indicate that prices ranged from ₹4,306 to ₹14,105, with a standard deviation of ₹3,580.47, reflecting significant variability. The data also show a positive skewness of 0.8619 and a platykurtic value of 2.6052, suggesting a non-normal distribution, confirmed by the Jarque-Bera test [30]. A time plot of the series, shown in Fig. 5, further validates the non-stationarity and nonlinearity of the data. The dataset consists of 2,498 observations, divided into a training set (80%) with 1,998 observations and a testing set (20%) with 500 observations used for post-sample predictions.

The study aimed to develop a GAN model for forecasting agricultural commodity prices and compare its performance with models like LSTM, Bi-LSTM, and GRU. The experiments were conducted on a system with an Intel Core i5-10500 CPU, 2.50 GHz, 8 GB RAM, and Intel UHD Graphics 10 GB.

Table 2. Descriptive statistics of Soya stock price series (in INR)

Descriptive statistics	Price (in INR)
Minimum	17.00
Mean	4306.00
Maximum	14105.00
Standard deviation	3580.47
CV (%)	83.15
Skewness	0.88
Kurtosis	2.60
Jarque-Bera	339.95**

3.2 Test for Stationarity

To check the stationarity of the soya price series, the Augmented Dickey-Fuller (ADF), Phillip-Perron (PP), and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) tests were applied. As shown in Table 3, the ADF test failed to reject the null hypothesis, indicating non-stationarity. Similarly, the PP test confirmed non-stationarity. The KPSS test also supported the non-stationarity of the series.

Table 3. ADF, PP and KPSS test results of daily price series of soya stock

Price series	ADF test		PP test		KPSS test	
	Statistic	p-value	Statistic	value	Statistic	p-value
Soya stock price series	-2.2800	0.2260	-3.42	0.28	9.05	0.01

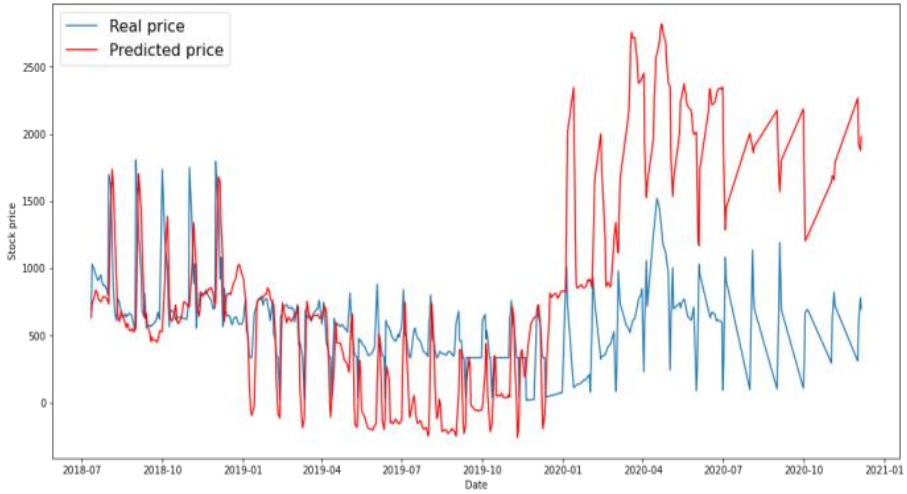

Fig. 5 (a). GRU model on test data

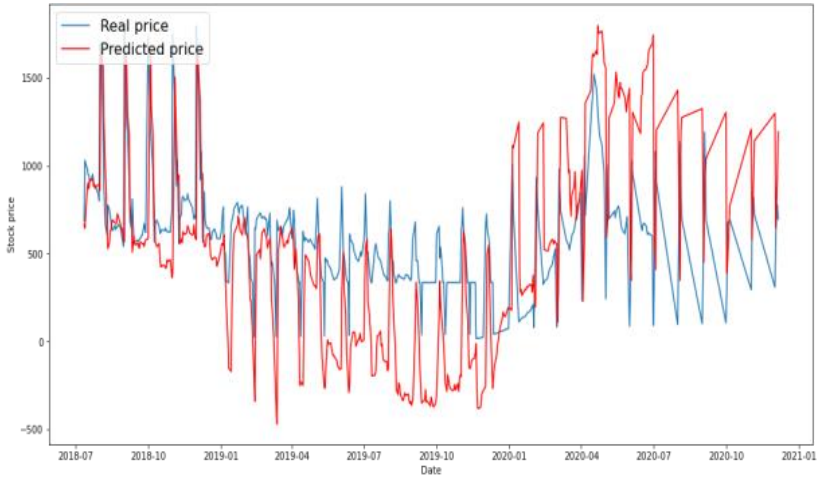

Fig. 5 (b). LSTM model on test data

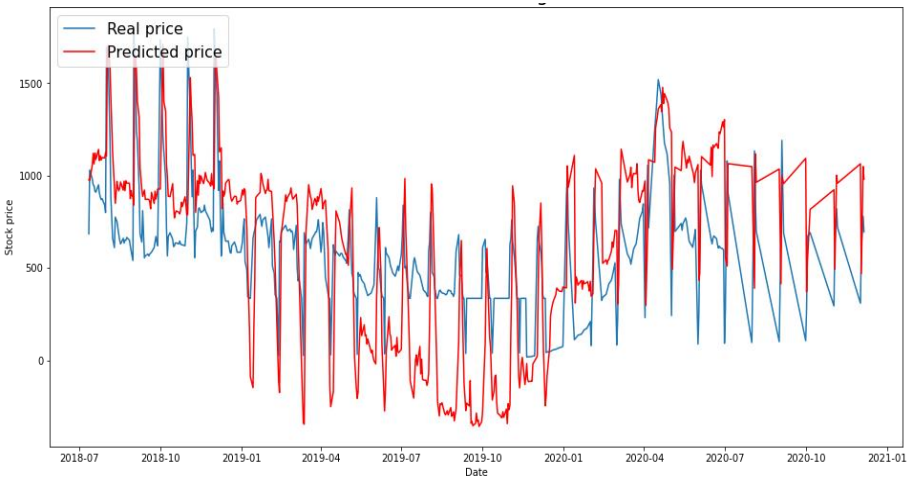

Fig. 5 (c). Bi-LSTM model on test data

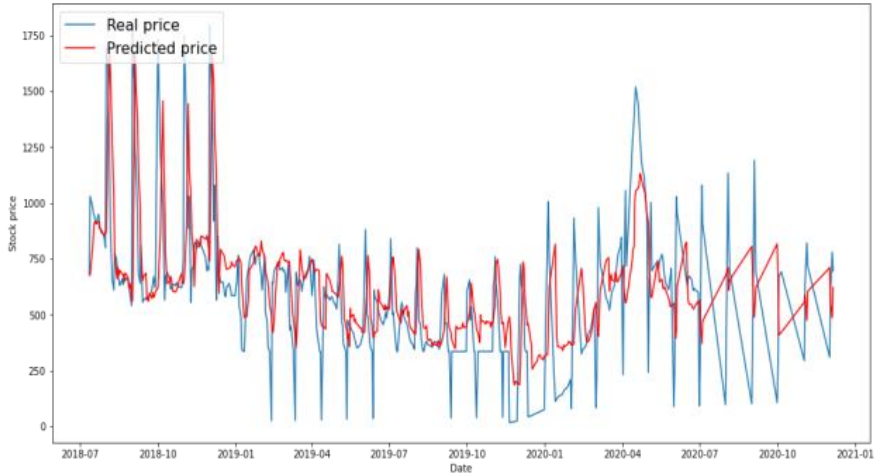

Fig. 5 (d). GAN model on test data

3.3 Brock-Dechert- Scheinkman (BDS) Test for Nonlinearity

The BDS test was applied to assess the nonlinearity of the time series. The test results, summarized in Table 4, confirmed significant nonlinearity in the soya stock market price data, particularly at embedding dimensions of 2 and 3. The p-values were calculated for different levels of proximity and indicated nonlinearity for the dataset.

Table 4. BDS test for non-linearity

Epsilon for close points	Embedding dimensions		p-value
	2	3	
0.5 σ	208.08	344.99	<0.0001
1.0 σ	154.16	186.64	<0.0001
1.5 σ	146.38	160.76	<0.0001
2.0 σ	131.97	132.33	<0.0001

3.4 Data Pre-processing and Normalization

The data contained no missing values, so imputation was unnecessary. However, normalization was performed to ensure effective model fitting. The data were normalized using the MinMaxScaler function from Scikit-learn, which scaled the values between 0 and 1, facilitating smoother training for the neural network models.

3.5 Implementation of Forecasting Models

Given the confirmed non-stationarity and nonlinearity, various forecasting models were developed using Python's TensorFlow and Keras libraries. The GRU model, tuned using grid search, was structured with 128 input neurons, 64 hidden neurons in the first layer, and 32 neurons in the second hidden layer. This model, trained with the Adam optimizer, batch size of 128, and 50 epochs, achieved an RMSE of 741.63 (Fig. 5(a)).

The LSTM model, also optimized through grid search, yielded an RMSE of 449.72 on the test data (Fig. 5(b)). The Bi-LSTM model outperformed both GRU and LSTM, achieving an RMSE of 388.04 on the test data (Fig. 5(c)).

Building on the strengths of these models, a GAN model was designed to incorporate elements from both GRU and Bi-LSTM. The generator in the GAN utilized three layers of GRU with 1024, 512, and 256 neurons, while the discriminator employed a Convolutional Neural Network (CNN) with three 1D convolution layers containing 32, 64, and 128 neurons. The discriminator also featured three dense layers with 220, 220, and 1 neuron. The Leaky ReLU activation function was used throughout, except in the output layer, which employed the Sigmoid function.

Table 5. Optimized results obtained by different models for the Soya price series data

Models	RMSE
GRU	741.63
LSTM	449.72
Bi-LSTM	388.04
GAN	**251.60**

The GAN model significantly outperformed the other models, achieving an RMSE of 251.60 (Fig. 5(d)), making it the best-performing model in the study. As presented in Table 5, the model was further fine-tuned using Bayesian Optimization, which helped to minimize forecast error and direction prediction loss. This combined architecture, termed GAN-FD (GAN for minimizing forecast error and direction prediction loss), demonstrated superior predictive accuracy.

4. CONCLUSIONS

This paper presents a novel GAN model designed specifically for predicting soya stock prices by generating time series data through the generator component of the network. This is a pioneering application of GANs in agricultural time series forecasting. The model successfully integrates the inherent uncertainties of soya stock prices by utilizing a deep neural network-based generator that captures noise sequences in the latent space. Through adversarial training, the GAN model demonstrates superior prediction accuracy over traditional forecasting methods, highlighting its potential in agricultural stock price prediction. However, tuning hyperparameters, especially in GAN models involving RNNs, remains a challenge, as improper adjustments can lead to model instability. Future research should focus on developing advanced hyperparameter optimization techniques, such as reinforcement learning, to enhance the predictive performance of GAN models in this context and further improve the accuracy and robustness of predictions in agricultural stock price forecasting.

DISCLAIMER (ARTIFICIAL INTELLIGENCE)

Author(s) hereby declare that NO generative AI technologies such as Large Language Models (ChatGPT, COPILOT, etc) and text-to-image generators have been used during writing or editing of this manuscript.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

1. Avinash G, Ramesh AG, Rebasiddanavar RM, Shilpa M, Vinay HT. A study on temporal variation of land use pattern in Karnataka through indices; 2022a.
2. Avinash G, Ramasubramanian V, Gopalakrishnan BN. Heterogeneous Auto-regressive modeling based realised volatility forecasting. Stat. Appl. 2022b;21(2):121-140.
3. Nayak GH, Alam W, Singh KN, Avinash G, Ray M, Kumar RR. Modelling monthly rainfall of India through transformer-based deep learning architecture. Modeling Earth Systems and Environment. 2024a;1-18.
4. Nayak GH, Alam MW, Singh KN, Avinash G, Kumar RR, Ray M, Deb CK. Exogenous variable driven deep learning models for improved price forecasting of TOP crops in India. Scientific Reports. 2024b;14(1):17203.
5. Vinay HT, Pavitra VMSJ, Avinash G, Nayak GHH. A Comparative analysis of time series models for onion price forecasting: insights for agricultural economics. Journal of Experimental Agriculture International. 2024;46(5):146-154.
6. Baishya M, Avinash G, Sharma K, Veershetty, Nayak GH. Navigating soybean price volatility: A deep learning perspective. International Journal of Statistics and Applied Mathematics; SP. 2023;8(5):980-984
7. Singh KN, Sharma K, Avinash G, Kumar RR, Ray M, Ramasubramanian V, Lama A, Lal SB. LSTM based stacked autoencoder approach for time series forecasting. *J.* Indian Society of Agricultural Statistics. 2023;77:71-78.
8. Immad A Shah, SukhDev Mishra. Artificial intelligence in advancing occupational health and safety: An encapsulation of developments, Journal of Occupational Health. 2024;66(1):uiad017, Available:https://doi.org/10.1093/joccuh/uiad017
9. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Bengio Y. Generative adversarial nets. Advances in Neural Information Processing Systems. 2014;27.
10. Ding X, Zhang Y, Liu T, Duan J. Deep learning for event-driven stock prediction. In Twenty-fourth International Joint Conference on Artificial Intelligence; 2015.
11. Jaiswal R, Jha GK, Kumar RR, Choudhary K. Deep long short-term memory-based model for agricultural price forecasting. Neural Computing and Applications. 2021;1-16.
12. Jinsung Y, Daniel J, Schaar Mihaela Van Der. In Time-series generative adversarial networks. In International Conference on Advances in Neural Information Processing Systems (5508-5518); 2019.
13. Liu Y, Yu R, Zheng S, Zhan E, Yue Y. NAOMI: Non-autoregressive multiresolution sequence imputation. Advances in Neural Information Processing Systems. 2019;32.
14. Araujo RDA, Oliveira AL, Meira S. A hybrid model for high-frequency stock market forecasting. Expert Systems with Applications. 2015;42(8):4081-4096.
15. Avinash G, Ramasubramanian V, Ray M, Paul RK, Godara S, Nayak GH, Kumar RR, Manjunath B, Dahiya S, Iquebal MA. Hidden Markov guided Deep Learning models for forecasting highly volatile agricultural commodity prices. Applied Soft Computing. 2024a;158, 111557.
16. Avinash G, Ramasubramanian V, Ray M, Paul RK, Dahiya S, Iquebal MA, Godara S, Manjunath B. Price Forecasting of TOP (Tomato, Onion and Potato) Commodities using Hidden Markov based Deep Learning Approach. Statistics and Applications. 2024b;22(2):1-28.
17. Che Z, Cheng Y, Zhai S, Sun Z, Liu Y. Boosting deep learning risk prediction with generative adversarial networks for electronic health records. In 2017 IEEE International Conference on Data Mining (ICDM). 2017;787-792.
18. Chen Z, Jiang C. Building occupancy modeling using generative adversarial network. Energy and Buildings. 2018;174:372-379.

19. Kara Y, Boyacioglu MA, Baykan ÖK. Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. Expert systems with Applications. 2011;38(5): 5311-5319.
20. Huang W, Nakamori Y, Wang SY. Forecasting stock market movement direction with support vector machine. Computers and Operations Research. 2005;32(10):2513-2522.
21. Majhi R, Panda G, Sahoo G, Panda A, Choubey A. Prediction of S&P 500 and DJIA stock indices using particle swarm optimization technique. *In 2008 IEEE Congress on* Evolutionary Computation (IEEE World Congress on Computational Intelligence) (1276-1282). IEEE; 2008.
22. Gao N, Xue H, Shao W, Zhao S, Qin KK, Prabowo A, Salim FD. Generative adversarial networks for spatio-temporal data: A survey. ACM Transactions on Intelligent Systems and Technology (TIST). 2022;13(2):1-25.
23. Mogren, O. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:*1611.09904. (https://arxiv.org/abs/1611.09904 was available for access on May 31, 2022); 2016.
24. Sheta AF, Ahmed SEM, Faris H. A comparison between regression, artificial neural networks and support vector machines for predicting stock market index. Soft Computing. 2015;7(8):2.
25. Zhang K, Zhong G, Dong J, Wang S, Wang Y. Stock market prediction based on generative adversarial network. Procedia Computer Science. 2019;147:400-406.
26. Zhang C, Kuppannagari SR, Kannan R, Prasanna VK. Generative adversarial network for synthetic time series data generation in smart grids. In 2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (Smart Grid Comm) (1-6). IEEE; 2018.
27. Luo Y, Cai X, Zhang Y, Xu J. Multivariate time series imputation with generative adversarial networks. Advances in Neural Information Processing Systems. 2018; 31.
28. Li S, Li W, Cook C, Zhu C, Gao Y. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2018;5457-5466.
29. Huang X, Li Q, Tai Y, Chen Z, Liu J, Shi J, Liu W. Time series forecasting for hourly photovoltaic power using conditional generative adversarial network and Bi-LSTM. Energy. 2022;246:123403.
30. Kundu MG, Mishra S, Khare D. Specificity and sensitivity of normality tests. in proceedings of VI international symposium on optimisation and statistics. Anamaya Publisher, India; 2011.

Peer-review history:
The peer review history for this paper can be accessed here:
https://www.sdiarticle5.com/review-history/123063