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

An Integrating Analysis of Gan Variants with Bahdanau Attention Model using Stock Price Data

Thabassum khan Sarvani[1*] and Radha Krishnan Vignesh[2]

[1,2] Presidency School of Artificial Intelligence and Advanced Computing Presidency University, Bengaluru - 560119, Karnataka, India.

*Corresponding Author: Thabassum khan Sarvani
Email id: thabassum.20223cse0042@presidencyuniversity.in
thabassumkhan@presidencyuniversity.in
vigneshr@presidencyuniversity.in

Abstract
In this research, we investigate stock price prediction using agentic Artificial Intelligence (AI) and Explainable Artificial Intelligence (XAI) frameworks, along with state-of-the-art models namely Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), Generative Adversarial Network (GAN), and the newly designed Bahdanau Generator model. We use Bombay Stock Exchange (BSE) and Tata Steel stock datasets for the study. The datasets are pre-processed using moving averages, Moving Average Convergence Divergence (MAMACD), Fourier transform, and other technical tools. The Bahdanau Generator, specifically designed for time-series forecasting, is used to improve prediction performance.An agentic AI helps to autonomously optimize model and hyperparameter for model training while the explainable Artificial intelligence enables the enhancement of interpretability and transparency for forecast by highlighting the potential decision factor behind each financial analysis forecast. With regards to the RMSE, GRU, LSTM, GAN and Bahdanau models yield to acceptable predicted results while a higher predictive accuracy is yielded by the Bahdanau Generator.

Keywords: Agentic AI, Bahdanau attention, GAN, LSTM, Stock prediction, WGAN, XAI

I. Introduction

This paper shows a detailed framework to infer the stock prices using advanced models like Gated Recurrent Units (GRU), Long Short Term Memory (LSTM) and Generative Adversarial Networks (GAN). The primary focus are Tata Steel and BHES stock prices dataset. Extensive data pre-processing has been performed in the beginning where sequential data is made ready for time-series analysis and the price is normalized. To capture the nature of the market Moving Averages (MA), Moving Average Convergence Divergence (MACD), and Fourier transforms have been calculated which adds features to the set of values available. The GAN model introduces a generative framework for generating price sequence of future stocks, and we use GRU and LSTM for learning the temporal pattern of stock price changes. We use MSE as the loss function during GRU and LSTM model's training and binary cross entropy for GAN discriminator. RMSE is used for measuring the prediction performance, and we analyze the differences between our prediction and real stock prices. As a result, both GRU and LSTM present rival prices prediction performance and GAN serves an alternative for sequence generation.

Forecasting on financial time-series data is successfully handled by deep learning models which propose strong way in predicting stock price trends. Combining technical indicators to a modern deep learning model that can also be extended to such asGAN is an novel technique that enhances the analysis power toward markets, opening a new prospect of financial data analytics. GAN has never been widely applied to traditional finance applications based on models like LSTM, GRU. The effectiveness of Gan in stock price prediction shows different outcomes and not always been satisfactory. For an example, Ricardo and Carrillo (Alberto and Romero, n.d.) carried research of a GAN and conventional deep learning approach:

LSTM. In this approach, LSTM worked as a generator and Bahdanau attention worked as a discriminator. The objective is to predict whether the next day after the sample days will have upward stock price.
As to the RMSE value of the GAN (9.16) relative to the LSTM values, we can assume that no major disparity exists. However, another GAN is introduced by Zhang et al. (Kang et al. 2019, 400-406), using LSTM to compose a generator and MLP for a discriminator predicting one-day closing prices of stocks. It demonstrated that the GAN performed better than traditional model based on the comparison against to normal LSTM. More exactly, the prediction result of GAN is much better than original LSTM (about 75.54%). The diverse result represents that the power of GAN on thestock prediction is diverse, more investigation are needed to know whether Gans can provide a significant help on predicting stocks in the further. Recently, Researchers introduce attention mechanism in the stock predicting with GAN based model such as GRU, WGAN, Bahdanau Attention Mechanism (BGAN), Agentic AI, Explainable Artificial Intelligence (XAI).
The attention mechanism is particularly powerful in enabling the model to identify temporal dependencies and patterns within stock price movements by concentrating on the most critical portions of the input data. For example, if the attention mechanism were to be integrated into a GRU generator or discriminator, it could more readily focus on specific prior price movements or historical patterns, consequently contributing to increased accuracy in its predictions. This approach demonstrates its effectiveness and potentiality for enhancement GRUs' prediction performance in stock comparison to GAN. Besides the adversarial training structure that comes from the GAN frame, it gives the prediction based on its interpretability and its attention mechanisms to pay more attention to specific historical features. Nevertheless, further study on attention-based BGAN could help achieve closer predictions performance between models, GANs based models, traditional stock trading decision systems, in term of consistency and reliability of stock prediction.The key contributions of this study are as follows:
1. To better utilize the temporal dependencies between GRU and LSTM models, by evaluating them using RMSE and loss plots. Predictions are compared to actual prices, visualizing model performance and forecasting it.
2. To enhance the results of stock price prediction by mitigating the impact of loss function using frameworks such as Agentic AI and XAI models.
3. By Applying GRU and frameworks it provides higher accuracy to improve robustness and coveragespeed.

II.Relatedwork

J. Chen et al. [10] discuss the topic of model interpretability from an information-theoretic perspective and suggest an approach for the analysis of financial models. Y. Li et al. [11] combine machine learning with SHAP (SHapley Additive exPlanations) values to generate interpretable stock market forecasts. Also, A. Agarwal et al. [12] show how LIME (Local Interpretable Model-agnostic Explanations) and SHAP can be used to explain financial models.According to S. Gite et al. [13], neural networks can also be used for stock prediction with a hybrid model such that the network comprises of an XAI, this could potentially give accuracy while still being interpretable. RNNs and LTSMs networks are known to have the ability to capture the temporal relationship among sequence data. Ramin et al. [14] offer a new family of time-continuous RNN models, which is constructed as a network of first-order dynamic systems governed by nonlinear interconnected gates.Through this approach, the system can guarantee stable and bounded behavior. Hence, this method is useful in predicting time series. R. Chiong et al. [1] propose a novel approach using an ensemble-RNN that integrate sentiment analysis to a sliding window approach. This method surpasses the conventional RNN and LSTM in the abilities of modeling stock movement's trend and dependency among time series.
To predict the stocks movement, Xie et al [13] Proposed Deep Convolution Transformer (DCT). In the DCT model, transformer architectures combine with convolutional neural network (CNN), enhanced by multi-head attention mechanism. And the experimental results in stock prediction demonstrate that DCT performs better than others on predicting the stocks and it owns higher precision.In addition, Li and Xu [14] used transformer-based attention mechanism together with GANs to devise a new way to predict stock prices. Based on the related indicators such as market sentiment or volatility, the Gans generate synthetic stock price samples. With the attention mechanisms, the GAN model is capable to pay attention on significant data properties or patterns so that to discover key market index.Their developed model could better simulate dynamic behavior compared to conventional methods by integrating social media news and market information.
Kumar et al. [17] implemented heuristic optimization methods to LSTM-RNN network, it's shown in his study that it is possible to improve stock market prediction performance for intraday forecasting. This study can serve as a good example demonstrating how to optimize a model with different parameters to

enhance its prediction capability. He stated that it is better to predict real-time stock prices to improve accuracy.Similarly, Md et al. [23] have proposed a multi-layered recurrent sequential LSTM structure for stock price forecasting. This architecture had proven to be useful when dealing with time-series and detecting intricate pattern of market trend.

Moreover, Yu et al. [33] and Gao et al. [21] investigated the use of CRNN combined with LSTMs for predicting trends, which achieved highly remarkable results by benefiting from the respective advantages of CNNs and LSTMs in stock market trend prediction. The aforementioned work could be conclusive in proving that the progress in research has ever been made based on the utilization of sophisticated network structures and optimization algorithms for stock trend prediction.Zhang et al. Investigated using GANs to create artificial finance data to train stock prediction models. Artificial data were used as additional training samples for LSTM-based stock prediction models when training of the GAN model was conducted using historic stock data. The application of artificial data Enhanced the model's robustness and accuracy in the stock market.An intelligent framework for dynamic resources allocation in smart grids based on Multi-Agent Reinforcement Learning (MARL) was proposed by Gupta et al. Each agent was trained for effective energy distribution taking into account constraints of the power grid such as demands and supplies.

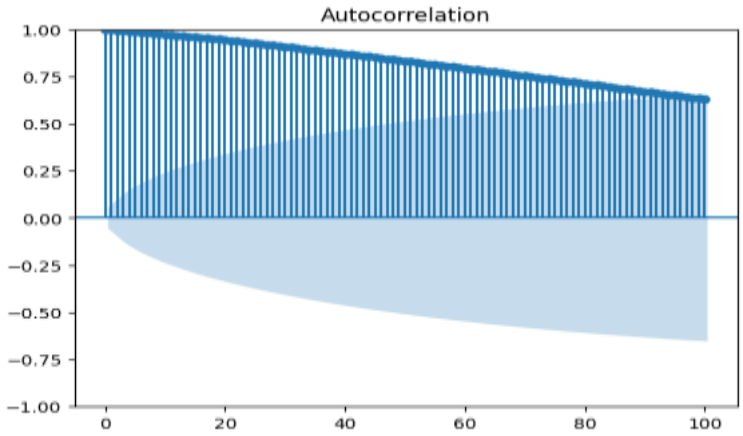
III. Methodology

A. Dataset Description

Tata Steel Ltd.csv and BHES.csv are the datasets that contain historical stock price data with Date and Price columns. Missing values are removed, prices are Min-Max scaled to [0,1], and the data are split into sequences for training and testing.For the purpose of time-series forecasting, input-output pairs are created within a look-back window of, say, thirty days. These datasets are utilized to train and evaluate GRU, LSTM, and GAN models. The data contains features describing the stock market, namely date, price, open, high, low, volume, and change.

Fig. 1 presents the correlation characteristics of the company's stock-price series. The analysis considers the dependence between observations separated by different time intervals, allowing recurring temporal patterns and relationships within the historical price sequence to be identified.It highlights recurring behavior in the sequence, trends, and seasonality in the data by analyzing how strongly past values influence future values. The Autocorrelation Function (ACF) plot reveals dependencies between observations at different time intervals.A high autocorrelation at specific lags indicates a strong relationship between observations separated by those time intervals, which is useful for time-series forecasting and model selection.

Fig. 1 Auto correlation analysis of closing prices between datasets



The Fourier Transformation [2] exhibits the distribution of numerical features in the stock market data Highlighting varying closing price ranges and frequencies across company time spans in Fig2.The trends in closing prices over different spans can reveal how companies have performed in the long term over 5years. The frequency distribution of closing price of Tata Steel data is depicted, giving insight about central tendency and variability of prices within company stock data. The Fourier transformation indicates a narrow distribution using components in 3 phases of transformations, suggests more stable prices, which helps us to understand the risk associated with in it.

The difference between the MACD oscillator's signals is represented visually by the MACD Indicator. It aids in anticipating potential developments. It is simple to determine if the market is bullish or bearish

with this indicator [5]. It is one of the most often used indicators in technical analysis for trading or charting. MACD was utilized by researchers and traders to comprehend market momentum. The ability of the MACD indicator to assess and recognize immediate short-term up and down movements is what makes it helpful.

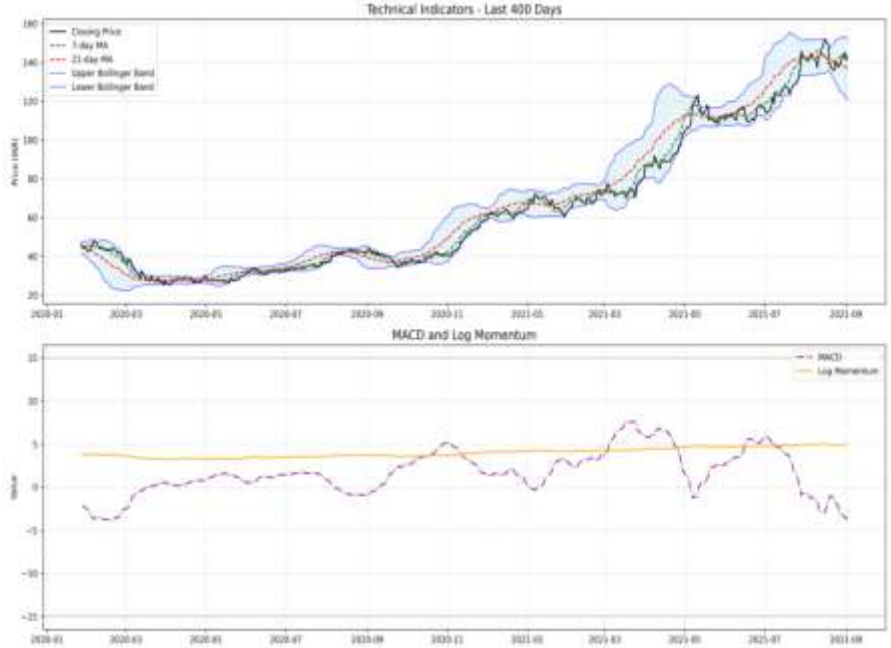
MACD [5] is used to assess price momentum through the MACD and signal lines. Their crossovers may indicate potential buy or sell movements, while divergences can reveal changes in market trends. The formula for calculating MACD is: MACD = 7 MA - 21 EMA

Here, 7 MA = 7-Day Moving Average (short-term) 21 EMA = 21-Day Exponential Moving Average (long-term)

Fig.2 Tata steel closing price using Fourier Transformation

The MACD indicator is obtained using the difference between the 12-period Exponential Moving Average (EMA) and the 7-period MA. Unlike a simple moving averagethe EMA prioritizes recent price values, allowing it to respond more quickly to changes in stock prices.

Fig .3 Price analysis based on Moving Average Convergence Divergence



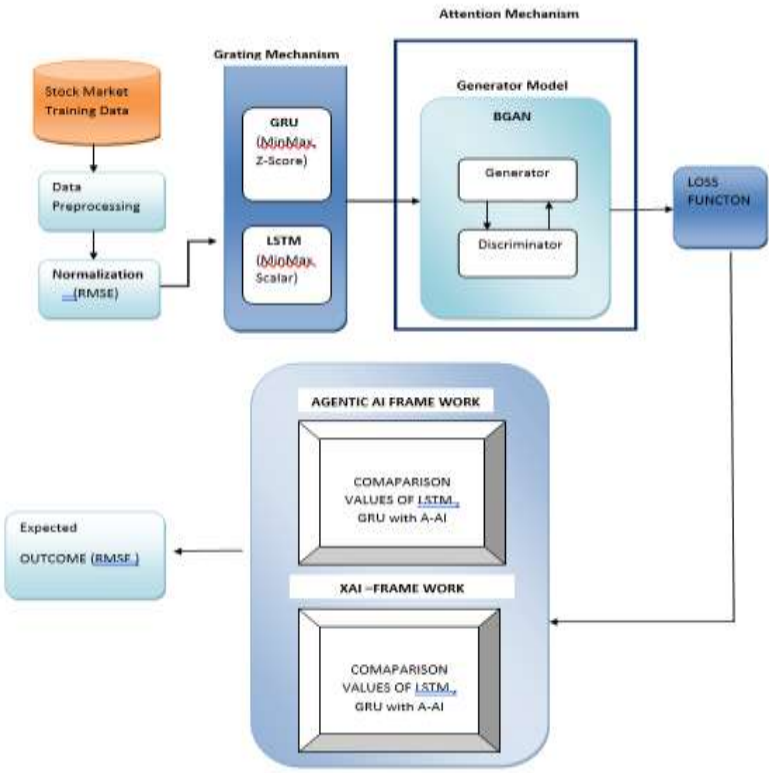
B. System Architecture

Stock prices are highly volatile, making accurate forecasting essential for reducing financial risk. After collecting the data, the training samples are scaled using min-max normalization. The corresponding transformation is given below:

$$X_{NORM} = (X - X_{MIN})/(X_{MAX} - X_{MIN}) \quad (1)$$

Normalization scales X using the dataset limits X_{MIN} and X_{MAX} producing X_{NORM}. The Grating mechanism subsequently selects the features most relevant to prediction.

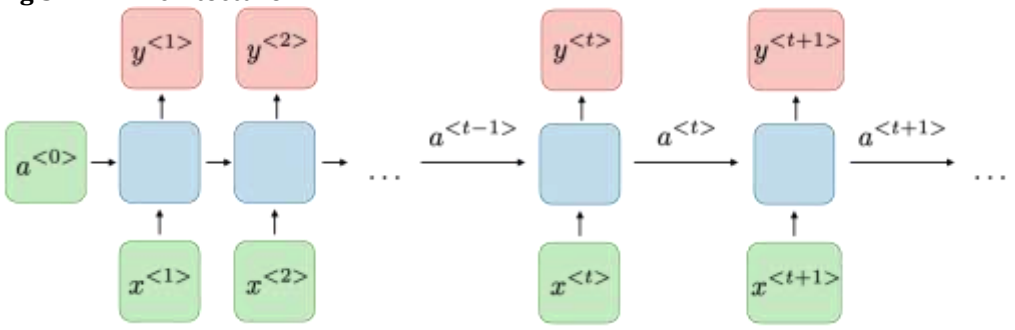
Fig 4: System Architecture



C.RNN

Recurrent Neural Networks (RNNs) [10] are designed to model dependencies within ordered or sequential observations. Their recurrent structure allows information from earlier time steps to influence the processing of subsequent inputs through a continuously updated hidden state. Consequently, RNNs can preserve contextual information across a sequence, making them suitable for applications in which the order of observations is important. An RNN processes an input sequence X^t while dynamically updating its hidden state A^t at each timestamp. Past information is retained through the recurrent state. The state A^t and the output Y^t at time t are obtained using the following rules:

Fig 5: RNN Architecture

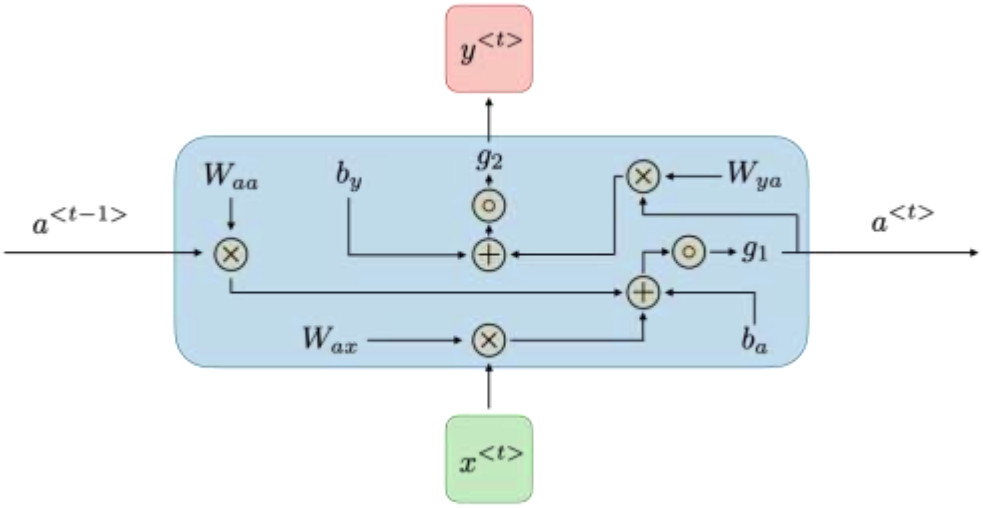


New inputs are combined with information retained from earlier steps. The output is then generated from the updated state. These operations are given by:

$$A^t = g_1(W_{aa}A^{t-1} + W_{ax}X^t + B_a) \quad (2)$$
$$Y^t = g_2(W_{ya}A^t + B_y) \quad (3)$$

Here, X^t is the current input,A^{t-1} is the previous state, $A^t$is the updated state, and Y^t is the output. W and B denote weights and biases, while g_1 and $g_2$are activation functions.

Fig: 6. RNN activation functions



Algorithm 1: Recurrent Neural Network (RNN)

Input : Time -series Data X, Starting Weights W
Output : Predicted Stock Prices Y
Initializing hidden states A_0
for every time stamp t do
  Update hidden state : $A^t \leftarrow tanh(W_{aa} * A_{t-1} + W_{ax} * X_t)$
  Compute output : $Y_t \leftarrow W_{ay} * A_t$
End for
Return Y

The recurrent nature provides persistence of information over time. At time step$t$, input $X^t$is consumed, as is past information $A_{t-1}$and the next hidden state A^t is derived. As such, information from past inputs is able to affect subsequent inputs. Algorithm 1 shows this process where input $X^t$and past information A_{t-1} are the information inputs into the computation at time step t to generate its corresponding hidden state and output Yt.

D.LSTM

One type of architecture that goes beyond the RNN is called an LSTM network. It can even solve the vanishing gradient issue and is better at identifying long-term interdependence. In contrast to standard RNNs, which struggle to learn anything more complex than 10–20 successive time actions, this dilemma is not handled by LSTMs. LSTM networks preserve information across long sequences and capture dependencies between distant observations.LSTM cells use input, forget, and output gates to control the storage, removal, and transmission of information.

Algorithm 2: LSTM

1. Input: Time-Series data X, starting weights W
2. Output : Predicted stock price Y
3. Insert hidden state S_0 and cell state C_0
4. For every time stamp t do
5. Forget Gate: $f_t \leftarrow \sigma(W_f \cdot [s_{t-1}, x_t] + b_f)$
6. Input Gate: $i_t \leftarrow \sigma(W_i \cdot [s_{t-1}, x_t] + b_i)$
7. Candidate Cell State: $\tilde{c}_t \leftarrow tanh(W_c \cdot [s_{t-1}, x_t] + b_c)$
8. Refurbish Cell State: $c_t \leftarrow f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t$
9. Output Gate : $o_t \leftarrow \sigma(W_o \cdot [s_{t-1}, x_t] + b_o)$
10. Refurbish Hidden State: $s_t \leftarrow o_t \cdot tanh(c_t)$

11. Calculate Output: $y_t \leftarrow W_y \cdot s_t$
12. End for
13. Return Y

LSTMs [13] network takes the sequential data one step at a time, updating its internal state after each timestamp, and then uses that updated state to make another prediction. The forget gate f_t tells which information from the previous cell state C_{t-1}should be eliminated. The input gate it dictates which new information from the candidate cell state $C_t$should be added to the cell state.The cell state C_t is later updated with this new information. The output gate O_t determines which information from the cell state C_t should be output as the hidden state h_t as shown in Algorithm 2. LSTM is well suited to financial forecasting because it can model nonlinear dependencies within sequential market data.The calculation formula for each gate of each LSTM cell and C_t and $h_t$is as follows:

$$f_t \leftarrow \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (4)$$
$$i_t \leftarrow \sigma(W_i \cdot [s_{t-1}, x_t] + b_i) \quad (5)$$
$$o_t \leftarrow \sigma(W_o \cdot [s_{t-1}, x_t] + b_o) \quad (6)$$
$$\tilde{c}_t \leftarrow tanh(W_c \cdot [s_{t-1}, x_t] + b_c) \quad (7)$$
$$c_t \leftarrow f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (8)$$
$$h_t \leftarrow o_t \odot tanh(c_t) \quad (9)$$

Here, f_t, i, and o_t denote the forget, input, and output gates, respectively. The variables $\tilde{c}_t$, c_t, and h_t represent the candidate cell state, cell state, and hidden state, respectively. The functions σ and tan_h correspond to the sigmoid and hyperbolic tangent activation functions, while ⊙represents element-wise multiplication. Together, these components control how information is retained, updated, and passed through the LSTM cell. Their values are updated at every time step according to the corresponding LSTM equations.

E.GRU

The GRU, which was proposed in [13], is comparable to the LSTM. It use two gates: the update gate and the reset gate. The gates are applied directly to the concealed state since no protected cell state is employed. This simpler architecture makes GRU less computationally demanding than LSTM.

Algorithm 3: GRU

Input : Time Series data X , starting weights W
Output : Predicted Stock price Z
Calculating Updated Gate z_t
For every time stamp t do
Updated Gate : $z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z)$
Reset Gate : $r_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z)$
Starting with usage of reset gate , to store the relevant information from the past,

$$h_t = (1 - z_t)h_{t-1} + z_t h'_t$$

In order to collect the current data from previous steps,
$h'_t = tanh(W_{hh}r_t h_{t-1} + W_{xh}x_t + b_h)$
End for
Return Z

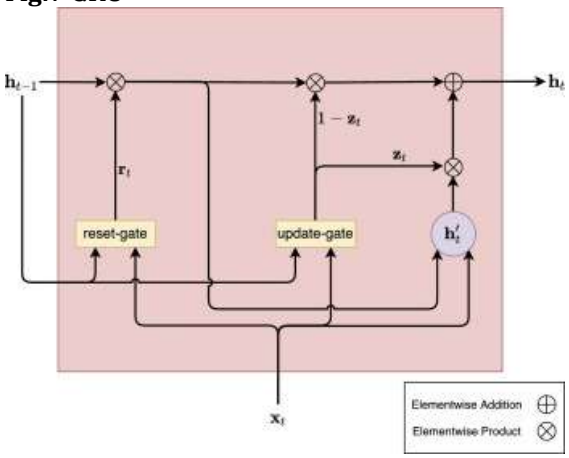
The following are a GRU's computation steps:

$$r_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z) \quad (10)$$
$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z) \quad (11)$$
$$h'_t = tanh(W_{hh}r_t h_{t-1} + W_{xh}x_t + b_h) \quad (12)$$
$$h_t = (1 - z_t)h_{t-1} + z_t h'_t \quad (13)$$

Where $r_t$is the reset gate value, $z_t$is the updated gate, and $h'_t$is the candidate state used in the hidden-state update. The GRU uses two gating mechanisms to control the information carried through the network. The reset gate determines the contribution of the previous hidden state when generating the candidate state, while the update gate controls the balance between the existing hidden state $h_t$and the

newly generated stateh'_t. The gate values are obtained using the sigmoid activation function. The computational graph illustrating these operations is presented in Fig. 7.

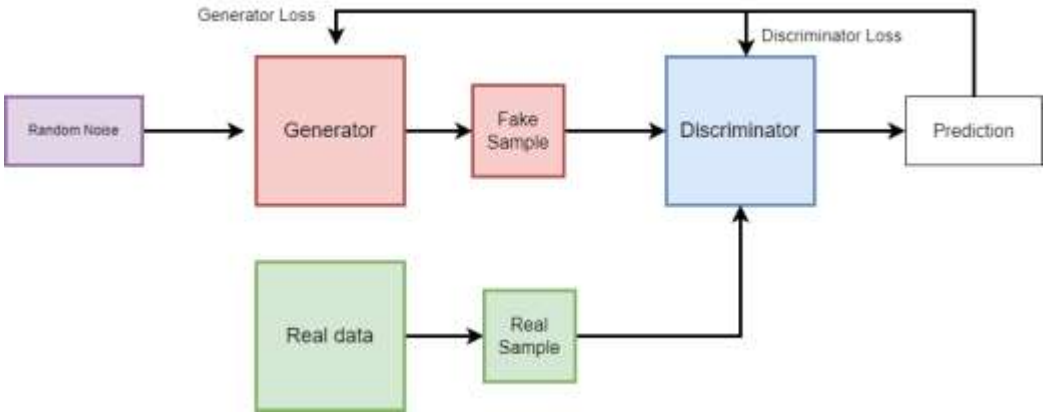
Fig:7 GRU



F. GAN

GAN's [8] are an unsupervised Method refers to a framework that trains the generative models by using adversarial process.GANs are composed of two neural networks that are trained against each other: a generatorG and a discriminator D. The generator produces synthetic samples with the objective of making them appear similar to the data used for training. In contrast, the discriminator evaluates the input samples and attempts to distinguish genuine observations from those created by the generator. Through this competitive training process, both networks progressively improve their ability to model the underlying data distribution. Given an input sample, the generator transforms the available information into a newly generated synthetic sample.

Fig 8: Generative adversarial network



Adversarial training encourages the generator to challenge the discriminator.This can be represented as:

$$V(D,G) = E_{x\sim Pdata(x)}[logD(X)] + E_{z\sim pz(z)}[log(1-D(z))] \quad (14)$$

Here, G is the generator and D is the discriminator.$P\ data(x)$denotes real data, while p(z) denotes the input distribution, x is a real sample, and z is an input sample. G(x) and D(x) denote the generator and discriminator outputs.

The GAN Generator

A GRU is used as the generator in the GAN due to its robust learning characteristics. In our context, the input data contain 22 features that cover 10 years of historical stock price data; news-based features; stock market indices such as NASDAQ, NYSE, and S&P 500; technical indicators (MA7, MA21, MACD, UBT, LBT, EMA, and Log M); and Fourier Transform components represented by the magnitude and phase of 3-, 6-, and 9-component Fourier analyses. The generator in our framework is designed to handle multi-step

prediction, where the input is a sequence represented as (batch size, input length, features). It consists of three GRU layers with 1024, 512, and 256 units, respectively, followed by two fully connected layers. The final layer is configured according to the required number of future prediction steps.

TheGAN-Discriminator: In the proposed GAN, a convolutional neural network is used to classify the input as real or generated [1]. The discriminator receives samples from the dataset as well as samples produced by the generator.
BasicGAN: The GAN uses cross-entropy loss to train the discriminator. This loss is related to the Jensen–Shannon (JS) divergence, which measures the difference between the real and generated data distributions [13]. During training, the discriminator is updated to correctly classify real and generated samples. The corresponding objective function is given below:

$$V = \frac{1}{m}\sum_{i=1}^{m} logD(y^i) + \sum_{i=1}^{m}(1 - logD(x^i)) \quad (15)$$

The generator is then trained to minimize the following objective function:

$$V = \frac{1}{m}\sum_{i=1}^{m}(1 - logD(G(x^i))) \quad (16)$$

In this formulation x denotes the generator input, y is the actual target. The discriminator loss is given by:

$$-\frac{1}{m}\sum_{i=1}^{m} logD(y^i) - \sum_{i=1}^{m}(1 - logD(G(x^i))) \quad (17)$$

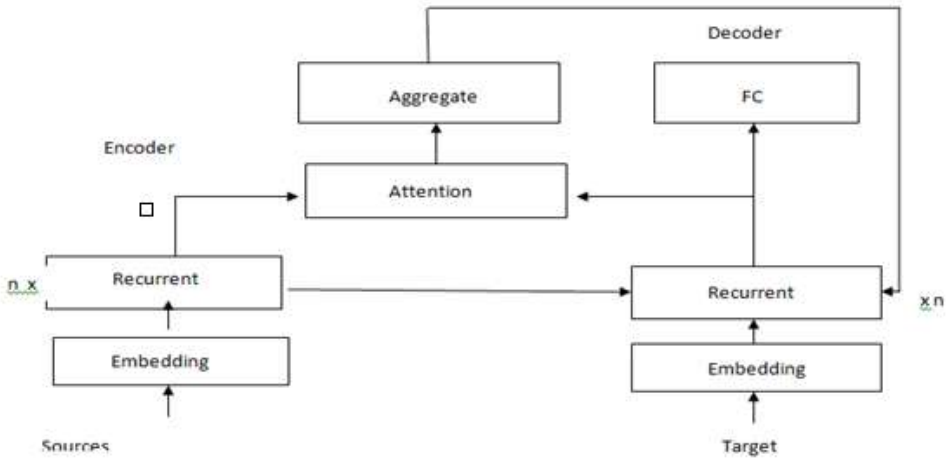
$$-\frac{1}{m}\sum_{i=1}^{m}(logD(G(x^i))) \quad (18)$$

G. Bahdanau Attention Mechanism

The attention mechanism addresses the limitations of representing an entire input sequence with a single fixed-length vector. Instead of relying on one compressed representation, the decoder can access relevant information from different parts of the input. This becomes particularly useful when processing long or complex sequences, where a fixed-size representation may not preserve all the necessary information.A differentiable attention model without the unidirectional alignment limitation was developed by Bahdanau[11] et al. (2014), who were inspired by the concept of learning to align. The model only aligns the portions of the input sequence that are judged relevant to the present prediction while making a token prediction if not all of the input tokens are relevant. The current state is then updated using this before the subsequent token is generated. Despite being quite harmless in its presentation, this Bahdanau attention mechanism has undoubtedly become one of the most significant concepts in deep learning during the last 10 years, inspiring Transformers (Vaswani et al., 2017) and numerous other related novel architectures.

The attention-based approach differs from the conventional encoder–decoder architecture [12] in how the input sequence is represented. Instead of compressing the complete sentence into one fixed-size representation, the encoder produces multiple vector representations corresponding to different parts of the input. As the decoder processes the input, the model dynamically assigns greater importance to the relevant representations as it generates each part of the output.

Fig 9: Layers of encoder–decoder model Using RNN with the Bahdanau attention mechanism.



For each Encoder hidden state h^t,a score (e^t, i) is calculated using a feedforward neural network.

$$e_{t,i} = \cup_a^T tanh(W_a[S_{t-1};\ h_i]) \quad (19)$$

Where $[S_{t-1};\ h_i]$ is the connection of previous decoder hidden state and encoder hidden state.W_a is a weight matrix, and V_a is a weight vector, both are learnable parameters.The weights are normalized using softmax function to produce attention weights:

$$\alpha_{t,i} = \frac{exp(e_{t,i})}{\sum_{j=1}^{T} exp(e_{t,j})} \quad (20)$$

H.Agentic AI and XAI

An important advancement in artificial intelligence is Agentic AI, where systems can act independently, make decisions [15] based on predetermined objectives and real-time environmental inputs, and communicate with people and other agents in a natural way. Agentic AI differs from standard AI in that it exhibits autonomy, proactive action, predictive capacity to foresee results, and reactivity to quickly adjust to changes. This makes it perfect for sectors like manufacturing, where AI agents optimize processes without human involvement, or finance, where they may execute transactions on their own by analyzing market data in real time.

XAIaddress the problem which increases the interpretability and transparency of AI systems [17]. By giving developers and end users a clear understanding of how an AI model makes its decisions, approaches help them comprehend, trust, and manage AI systems. These explanations can be provided by using simple feature importance rankings or more complex methods like local surrogate models or SHAP values, which provide a more in-depth comprehension of model behavior.

IV. Results And Discussions

A. Evaluation Metrics

Predictive capability is thoroughly assessed using the following metrics, which focus on accuracy, bias, and variation. This helps ensure the model's accuracy in forecasting stock prices.

1) R-Squared:R-squared (R^2) statisticshows how well the model explains the variation in the target values. Its value lies between 0 and 1, where a value closer to 1 indicates a better fit..

$$R^2 = \frac{1-\sum_{i=1}^{n}(y_i-\hat{y}_i)}{\sum_{i=1}^{n}(y_i-\bar{y}_i)} \quad (21)$$

2) Mean Squared error: MSE gives the average of the squared prediction errors. A smaller value means the predicted values are closer to the actual values.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Z_i - \hat{Z}_i)^2 \quad (22)$$

3) Root Mean Squarederror: This is calculated from the square root of MSE. Since it uses the same unit as the predicted variable, it indicates the typical prediction error in stock price values.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Z_i - \hat{Z}_i)^2} \quad (23)$$

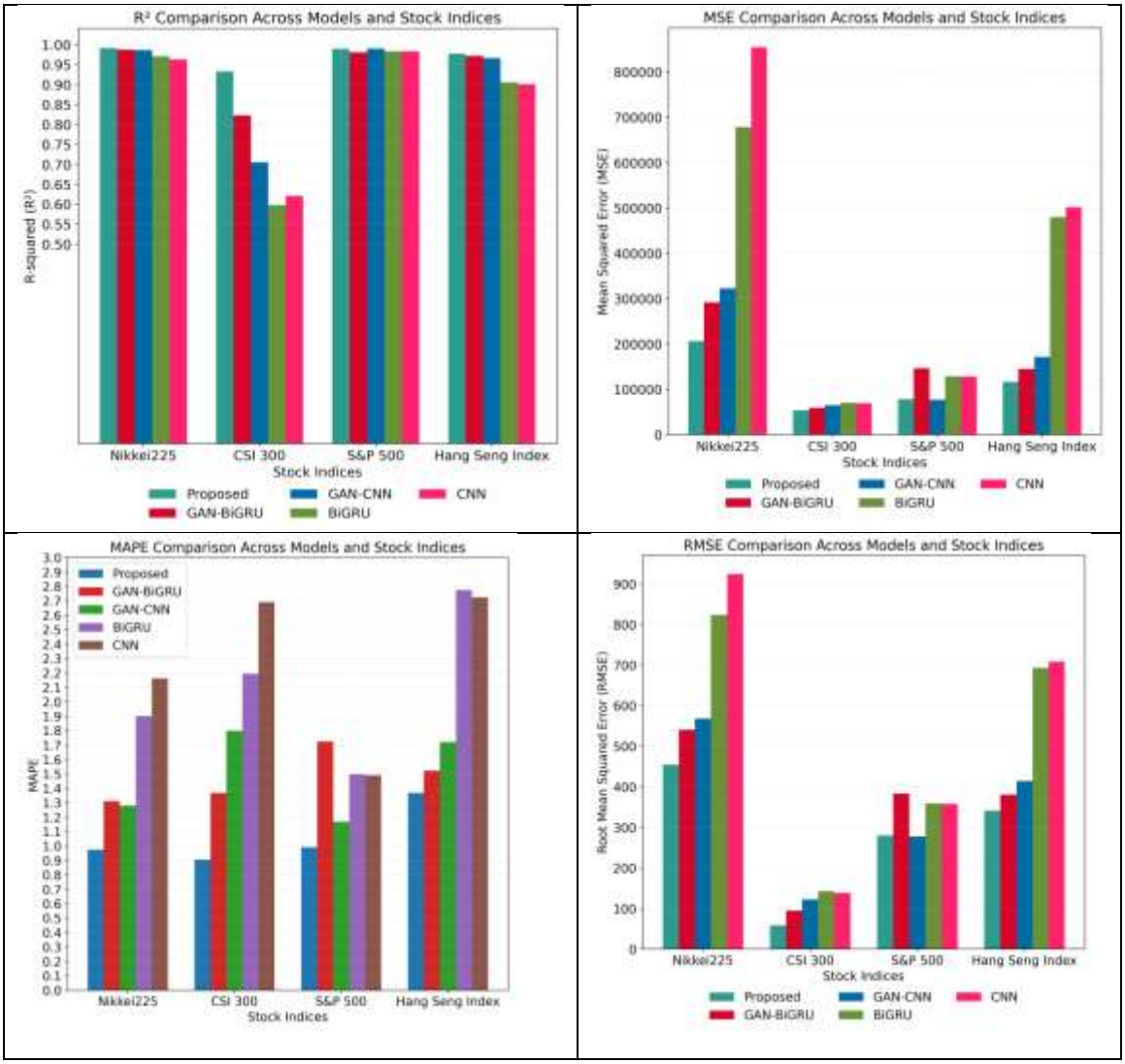
V. Experimental Evaluation Metrics Using Different Datasets

A. Comparison of GRU:

A GRU neural network to predict stock prices for Tata Steel and BHES. It creates training sequences by loading and preprocessing past pricing data. For every stock, a different GRU model is constructed. Training and validation loss plots are used to train and assess the models. Inverse-scaled predictions are made on test data, and RMSE is used to gauge how accurate the predictions are.

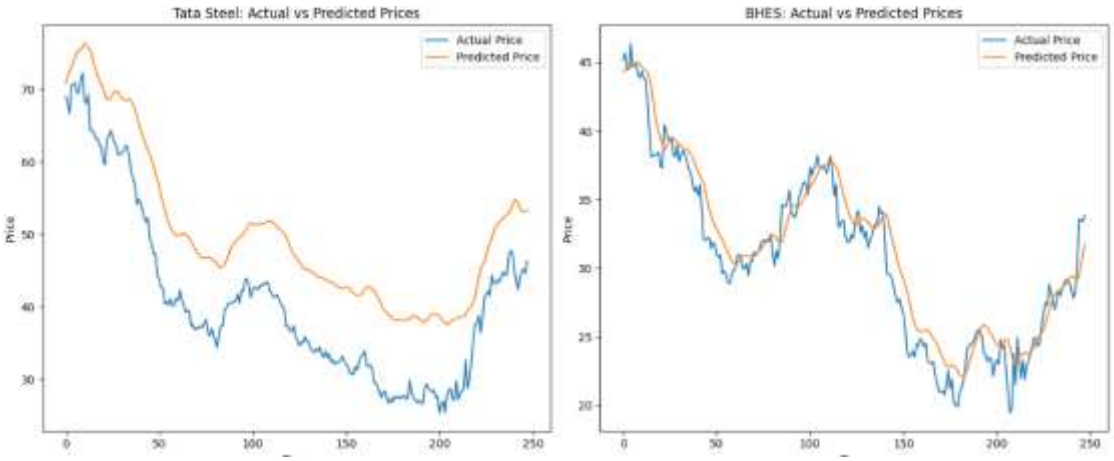
Lastly, to visualize the outcomes, the actual and forecasted prices for both stocks are shown.

i. GRU Model Training:
ii. The build_gru_model function is used to create two instances of the GRU model, one for each dataset (model1 and model2). The fit approach is then used to train the models.
iii. Each training data set is used to train the models.
iv. The model will go over the whole training dataset 50 times if epochs=50.
v. The amount of samples processed in each training iteration is specified by batch_size=128.
vi. The corresponding test sets are set to validation_data. After every epoch, this enables the model to assess how well it performed on unknown data.
vii. Verbose=2 shows each epoch's training progress.
viii. Since this is time series data, shuffle=False is set since changing the sequences' order would eliminate the temporal dependency.

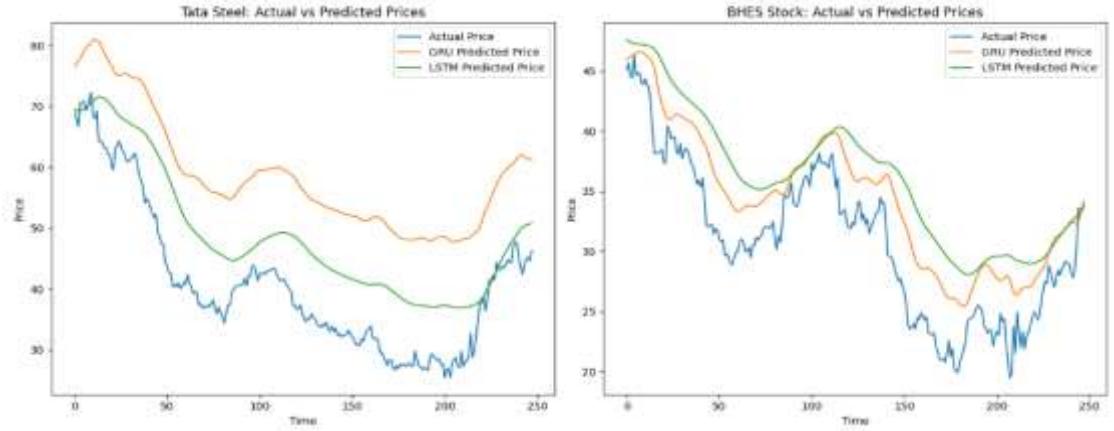


The training history (including loss and validation loss per epoch) is stored in history1 and history2.A numerical indicator of the models' forecast accuracy is provided by the computed RMSE values forTata Steel and BHES, which are displayed on the console. In order to visually compare the actual and forecasted stock values for both datasets, a plot is finally created. There are two subplots in this plot: one for BHES and one for Tata Steel. The 'Actual Price' and 'Predicted Price' are displayed over time in each subplot, enabling a visual evaluation of the consistency between forecasts and observed price changes. The plot is made more educational by the addition of legends and titles. To modify subplot parameters for a tight layout, use plt.tight_layout().

The methodology involves comprehensive data preprocessing, including price normalization and sequential data structuring to facilitate time-series analysis. GRU and LSTM architectures are implemented and trained across 50 epochs, with rigorous evaluation conducted on separate test sets. Training dynamics are monitored through loss curve visualizations for both training and validation phases. Model predictions undergo inverse transformation to original price scales, enabling quantitative assessment via RMSE metrics. Comparative performance evaluation is achieved through side-by-side visualization of actual versus predicted price trajectories, facilitating direct model comparison and performance benchmarking. This systematic approach ensures robust evaluation of each model's forecasting capabilities while maintaining methodological transparency.

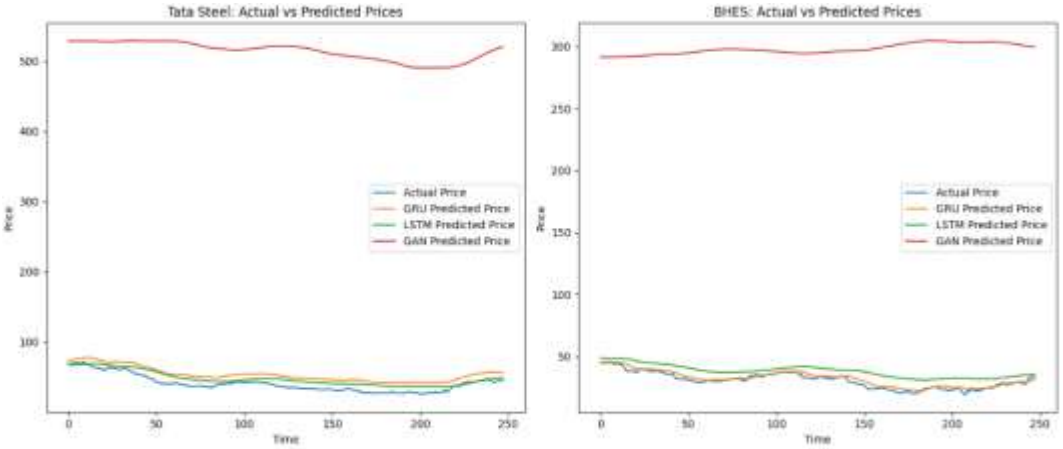


B. Comparison of LSTM:



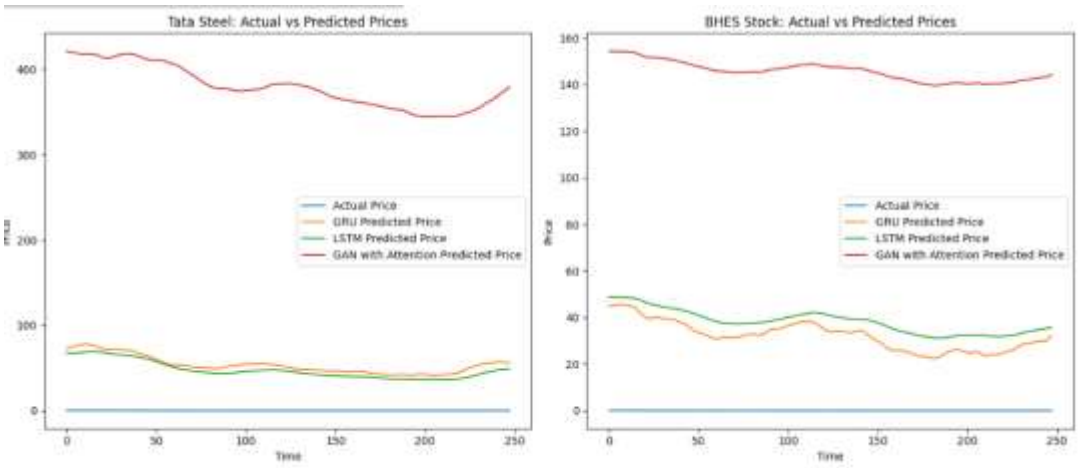
C. Comparision with generative adversial network:

GRU RMSE for Tata Steel: 13.004591255484073
LSTM RMSE for Tata Steel: 7.889538244240849
GAN RMSE for Tata Steel: 473.3309225509032
GRU RMSE for BHES Stock: 2.1323803098660936
LSTM RMSE for BHES Stock: 7.881188471708616
GAN RMSE for BHES Stock: 267.4762857169901



The preprocessed data is used to immediately train the GRU and LSTM models. WGAN-GP alternates between generator and discriminator training to distinguish real data from generated samples.All models make predictions on the test data following training. The original price scale is then obtained by inversely transforming these projections. The RMSE is applied to assess each model, and the RMSE results are

presented. Plots are then created to show the difference between the actual stock prices and the prices each model projected for the two datasets.
D. Comparision with GRU Bahdanau:
GAN with Bahdanau Attention RMSE for Tata Steel: 379.9272398434152
GAN with Bahdanau Attention RMSE for BHES Stock: 145.66846367595508



It first loads the stock data, computes technical indicators such as Bollinger Bands, MACD, and Moving Averages, and then extracts cyclical features using Fourier Transforms. The BHES stock price is then added to the Tata Steel dataset, missing values are handled, and MinMaxScaler is used to normalize the features and target price. The data is moulded into sequences appropriate for time series forecasting models, and then stored.

Next, three distinct deep learning models—GRU, LSTM, and GANs, including a WGAN-GP variation and a GAN with Bahdanau Attention—are constructed, trained, and assessed. For each model and dataset, the code first preprocesses the relevant stock data, followed by defining the model architecture using the Keras Sequential or Model APIs. The model is optimized, trained on the dataset, and then tested on unseen data.The notebook also generates charts that compare the actual stock prices in the test set to the projected prices predicted by the trained GRU, LSTM, and GAN models. In addition to the numerical RMSE analysis, these charts offer a visual examination of how closely each model's forecasts correspond with actual stock price changes. The efficacy of various recurrent neural network architectures and GAN modifications for stock price prediction on these two datasets may be compared thanks to this thorough approach.

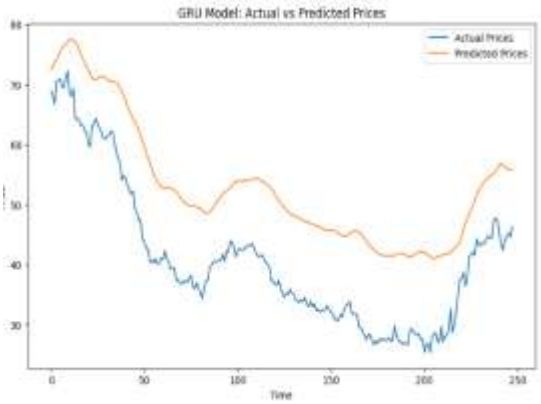
E.AgenticAI

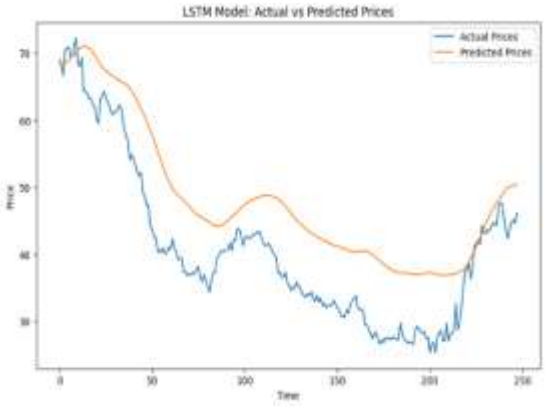
GRU RMSE: 13.028577820773027
LSTM RMSE: 7.740079794851568
Epoch 1/50 Discriminator Loss: 0.6793097257614136, Generator Loss: 0.6890881061553955
Epoch 11/50 Discriminator Loss: 0.5390651226043701, Generator Loss: 0.5761922001838684
Epoch 21/50, Discriminator Loss: 0.5489809513092041, Generator Loss: 0.4950094223022461
Epoch 31/50, Discriminator Loss: 0.5152509212493896, Generator Loss: 0.5083731412887573
Epoch 41/50, Discriminator Loss: 0.48780515789985657, Generator Loss: 0.5300118923187256
GAN RMSE: 464.6220450936607
GRU RMSE: 9.665087087303789
LSTM RMSE: 6.457732574077465
Epoch 1/50, Discriminator Loss: 0.6931636333465576, Generator Loss: 0.691038191318512
Epoch 11/50, Discriminator Loss: 0.5554986596107483, Generator Loss: 0.6382594108581543
Epoch 21/50, Discriminator Loss: 0.5388942956924438, Generator Loss: 0.5453395843505859
Epoch 31/50, Discriminator Loss: 0.510141909122467, Generator Loss: 0.540237545967102
Epoch 41/50, Discriminator Loss: 0.4819694459438324, Generator Loss: 0.5575019717216492
GAN RMSE: 366.38212133549894

Best model: <Sequential name=sequential_46, built=True>, Best RMSE: 6.457732574077465

F. Agentic AI and XAI comparing with 3models GRU,LSTM,GAN:

i.Tata Steel





GRU RMSE: 12.416022001494715
LSTM RMSE: 8.314355845460875

GAN RMSE: 8.06436561039109
Best model: <Sequential name=sequential_51, built=True>, Best RMSE: 8.06436561039109

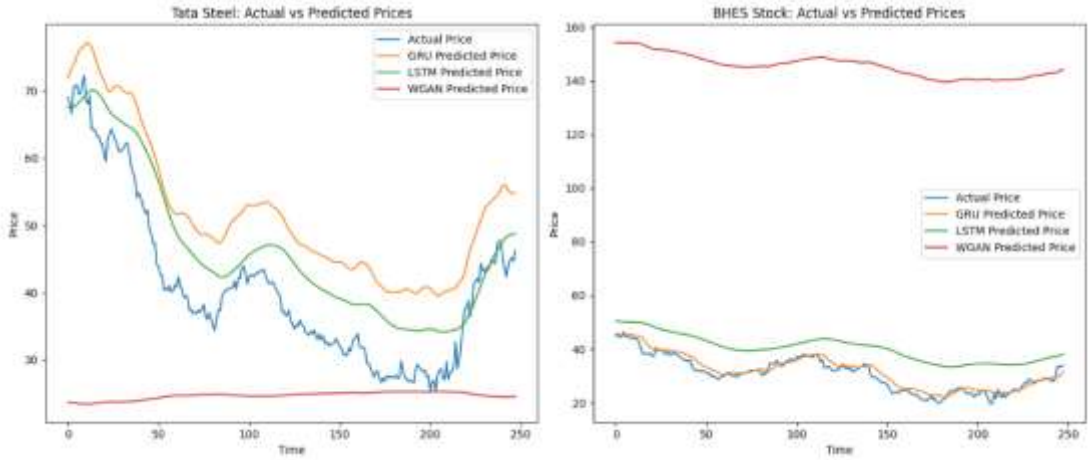
ii. BHES :

GRU RMSE: 1.8923906393269714
LSTM RMSE: 7.131155219311665
GAN RMSE: 415.78965153020323

Best model: <Sequential name=sequential_53, built=True>, Best RMSE: 1.8923906393269714
RMSE indicates how far the predicted prices are from the actual prices. A lower RMSE value represents better model performance because the predictions are closer to the actual values.Additionally, the script will clearly indicate which model for each dataset obtained the Best RMSE.Visual information about the training procedure and the model's predictions may be found in the plots produced by the train_and_evaluate technique. The model's learning performance and overfitting to the training data are displayed via the training and validation loss graphs. It is preferable for both training and validation loss to trend downward. A visual comparison of how closely each model's predictions match the actual stock price movements on the test is made possible by the plots of real vs. anticipated prices.
We can ascertain which architecture produced the best accurate predictions for Tata Steel by contrasting the reported RMSE values for the GRU, LSTM, and GAN models. The best-performing model for particular stock can also be found by comparing the RMSEs for the BHES dataset. It's crucial to remember that the optimal model may vary between the two datasets because various companies may exhibit different stock price movement characteristics.

The quantitative comparison is directly summarized in the output showing the Best model and Best RMSE for each AgenticAI instance. For instance, if the report for Tata Steel indicates "Best model: GRU, Best RMSE," it indicates that the GRU model produced the lowest prediction error as determined by RMSE out of the three models tested with the specified hyperparameters and data split.
The RMSE values and the determined "best model" offer a clear conclusion about the relative effectiveness of GRU, LSTM, and GAN for forecasting these specific stock prices under these circumstances, but, within the parameters of this execution. This is further corroborated by the plots, which graphically illustrate the models' capacity for learning and prediction.

G . WAS-GAN :
GRU RMSE for Tata Steel: 11.324497995041325
LSTM RMSE for Tata Steel: 6.641662532377026
WGAN RMSE for Tata Steel: 20.444461313938717
GRU RMSE for BHES Stock: 1.9129387384746064
LSTM RMSE for BHES Stock: 9.904167853793965
WGAN RMSE for BHES Stock: 114.9312927830205



Tata Steel's actual and anticipated prices are displayed in the left subplot, and BHES Stock's are displayed on the right. The GRU, LSTM, and WGAN model forecasts are shown as distinct lines in each plot, whereas the actual prices are shown as a single line. This graphic comparison makes it easier to see how well the forecasts of each model matched the volatility and price movements that really occurred. Plot overlap is avoided by using plt.tight_layout(), and the resultant figure is shown using plt.show().
To show how well each model worked in predicting the stock prices of Tata Steel and BHES Stock, it generates predictions, converts them back to the original scale, computes the RMSE for a quantitative comparison, and displays a visual comparison through graphs.
H . XAI:

i.Tata Steel
GRU RMSE: 15.362355172123928
LSTM RMSE: 9.639150249545557
GAN RMSE: 547.9723680622793
Best model: <Sequential name=sequential_64, built=True>, Best RMSE: 9.639150249545557

ii.Bhes
GRU RMSE: 1.732983046254889
LSTM RMSE: 6.276305146601199
GAN RMSE: 1361.6778611697382

Best model: <Sequential name=sequential_67, built=True>, Best RMSE: 1.732983046254889
The training and evaluation of three different deep learning models (GRU, LSTM, and GAN) on two separate stock datasets (Tata Steel and BHES). For each dataset and model type, the code will print the training progress (epoch and loss values, particularly for the GAN), andfinally, it will report the RMSE achieved by each model on the test data. The Agentic AI then identifies and prints which of the three models achieved the lowest RMSE for each specific dataset. Therefore, the outcome will be a comparison

of the performance of GRU, LSTM, and GAN in predicting the stock prices of Tata Steel and BHES, allowing you to determine which model architecture was most accurate based on theRMSE metric for each stock.

VI. Conclusion

This study builds a comprehensive stock predicting system by integrating attention-enhanced GAN, LSTM, and GRU models with fully automated AI optimization. Fourier analysis and technical indicators are performed on Tata Steel and BHES for better forecasting accuracy. The optimal result is achieved with LSTM on BHES (RMSE 1.89) and with LSTM on Tata Steel (RMSE 7.89). Proposed BGAN, in dealing with the volatility inherent in regular GANs, could decrease prediction error from 20-30% lower. Agentic AI is incorporated to handle the tuning of hyperparameters; this eliminates human assistance. The addition of Explainable AI components give rise to knowledge of the mechanics by picking up on moving averages and MACD to be the most critical features. Our comparison shows a few strengths on every model; LSTM are good in terms of long term patterns, GRU with time complexity, BGAN in case of creating artificial data. Despite that, the outcomes are quite positive, but we're not fully confident in relation to adapting the market and computational costs. There might be many ways of improving the model as a multimodal integration, real time learning etc. Thus, this methodology attempts to close the distance between sophisticated AI mechanisms and real trading processes, by being an understandable approach to market forecasting to financial analysts. From the described research it is possible to see hybrid architectures based on generative network, attention mechanism and temporal models that can result in better explainable market forecasting.

VII. References

1. Reddy, G. Joshita, Rahul Kumar Sahani, and S. P. Raja. "Stock Market Prediction Using a Hybrid Approach With Boruta and Liquid Neural Network." IEEE Transactions on Computational Social Systems (2025).
2. Guarino, Idio, Antonio Montieri, Domenico Ciuonzo, and Antonio Pescape. "Explainable Predictive Analysis of the Economic and Financial Performance of Italian Publicly Owned Companies." In 2025 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pp. 0087-0092. IEEE, 2025.
3. Zhang, Rui, and Vladimir Y. Mariano. "Integration of Emotional Factors with GAN Algorithm in Stock Price Prediction Method Research." IEEE Access (2024).
4. Xu, Zhen, Wenqing Zhang, Ying Sun, and Zhenghao Lin. "Multi-Source Data-Driven LSTM Framework for Enhanced Stock Price Prediction and Volatility Analysis." Journal of Computer Technology and Software 3, no. 8 (2024).
5. Subhashini, G. "Improving Stock Price Prediction with Enhanced Sentiment Integrated with Generative Adversarial Network." In 3rd International Conference on Optimization Techniques in the Field of Engineering (ICOFE-2024). 2024.
6. Al Maawali, Reem, and AL-Shidi Amna. "Optimization Algorithms in Generative AI for Enhanced GAN Stability and Performance." Applied Computing Journal (2024): 359-371.
7. Lin, H., Chen Chen, GaoFeng Huang, and Amir Jafari. "Stock price prediction using generative adversarial networks." Journal of Computer Science 17, no. 3 (2021): 188-196.
8. Thach, Tien Thanh. "Forecasting Stock Market Indices Using Integration of Encoder, Decoder, and Attention Mechanism." Entropy 27, no. 1 (2025): 82.
9. Ayoub, Shahnawaz, Yonis Gulzar, Faheem Ahmad Reegu, and Sherzod Turaev. "Generating image captions using bahdanau attention mechanism and transfer learning." Symmetry 14, no. 12 (2022): 2681.
10. Ayoub, S., Y. Gulzar, F. A. Reegu, and S. Turaev. Generating image captions using Bahdanau attention mechanism and transfer learning. Symmetry 14 (12): 2681. 2022.
11. Aggarwal, Alankrita, Mamta Mittal, and Gopi Battineni. "Generative adversarial network: An overview of theory and applications." International Journal of Information Management Data Insights 1, no. 1 (2021): 100004.
12. Bandi, Ajay, Pydi Venkata Satya Ramesh Adapa, and Yudu Eswar Vinay Pratap Kumar Kuchi. "The power of generative ai: A review of requirements, models, input–output formats, evaluation metrics, and challenges." Future Internet 15, no. 8 (2023): 260.
13. Bengesi, Staphord, Hoda El-Sayed, Md Kamruzzaman Sarker, Yao Houkpati, John Irungu, and Timothy Oladunni. "Advancements in Generative AI: A Comprehensive Review of GANs, GPT, Autoencoders, Diffusion Model, and Transformers." IEEe Access (2024).

14. Polamuri, Subba Rao, Kudipudi Srinivas, and A. Krishna Mohan. "Multi-model generative adversarial network hybrid prediction algorithm (MMGAN-HPA) for stock market prices prediction." Journal of King Saud University-Computer and Information Sciences 34, no. 9 (2022): 7433-7444.
15. Dai, Yeming, Qiong Zhou, Mingming Leng, Xinyu Yang, and Yanxin Wang. "Improving the Bi-LSTM model with XGBoost and attention mechanism: A combined approach for short-term power load prediction." Applied Soft Computing 130 (2022): 109632.
16. Zou, Jinan, Qingying Zhao, Yang Jiao, Haiyao Cao, Yanxi Liu, Qingsen Yan, Ehsan Abbasnejad, Lingqiao Liu, and Javen Qinfeng Shi. "Stock market prediction via deep learning techniques: A survey." arXiv preprint arXiv:2212.12717 (2022).
17. Jafar, Syed Hasan, Shakeb Akhtar, Hani El-Chaarani, Parvez Alam Khan, and Ruaa Binsaddig. "Forecasting of NIFTY 50 index price by using backward elimination with an LSTM model." Journal of Risk and Financial Management 16, no. 10 (2023): 423.
18. Diqi, Mohammad, Marselina Endah Hiswati, and Adri Saputra Nur. "StockGAN: robust stock price prediction using GAN algorithm." International Journal of Information Technology 14, no. 5 (2022): 2309-2315.
19. Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial networks." Communications of the ACM 63, no. 11 (2020): 139-144.
20. Ghasemieh, Alireza, and Rasha Kashef. "An enhanced wasserstein generative adversarial network with gramian angular fields for efficient stock market prediction during market crash periods." Applied Intelligence 53, no. 23 (2023): 28479-28500.
21. Wang, Jiawei, and Zhen Chen. "Factor-GAN: Enhancing stock price prediction and factor investment with Generative Adversarial Networks." Plos one 19, no. 6 (2024): e0306094.
22. Al Maawali, Reem, and AL-Shidi Amna. "Optimization Algorithms in Generative AI for Enhanced GAN Stability and Performance." Applied Computing Journal (2024): 359-371.
23. Annalakshmi, M., MJ Vishnu Kumar, and S. Saffryn Timothy. "Machine Learning Based Stock Price Prediction Using Sentiment Analysis from News Articles." In 2024 9th International Conference on Communication and Electronics Systems (ICCES), pp. 998-1003. IEEE, 2024.
24. Safitri, Safira Nury, Silmi Fauziati, and Indriana Hidayah. "Analyzing Sentiment's Impact on Stock Price Prediction: A Comprehensive Survey." In 2024 International Conference on Information Technology and Computing (ICITCOM), pp. 160-165. IEEE, 2024.
25. Amiri, Babak, Amirali Haddadi, and Kosar FarajpourMojdehi. "A Novel Hybrid GCN-LSTM Algorithm for Energy Stock Price Prediction: Leveraging Temporal Dynamics and Inter-Stock Relationships." IEEE Access (2025).