Predicting Vietnamese airlines' stock prices utilizing statistical, machine learning, and deep learning algorithms (2019 - 2024)

TRAN HOANG NHAT[1], THAI NGOC DUNG[2], AND NGUYEN HOANG VU[3]

[1]Faculty of Information Systems, University of Information Technology, (e-mail: 21522420@gm.uit.edu.vn
[2]Faculty of Information Systems, University of Information Technology, (e-mail: 21521982@gm.uit.edu.vn)
[3]Faculty of Information Systems, University of Information Technology, (e-mail: 21522799@gm.uit.edu.vn)

ABSTRACT

Stock prediction plays a crucial role in economic planning and investment decision-making. This study focuses on leveraging advanced statistical, machine learning and deep learning algorithms to predict stock prices in the Vietnamese market, with a particular emphasis on HVN, SCS, VJC. Eight time-series models, including Linear Regression, ARIMA, RNN, GRU, LSTM, SEMOS, Stacking and FCN, are utilized for prediction. Evaluation of these models is conducted using metrics such as MAPE, RMSE, and MSLE. Furthermore, related research on stock price prediction utilizing 8 algorithms has been examined and analyzed to demonstrate their effectivenes. The dataset spans from 01/03/2019 to 29/02/2024, and only the "Close" prices (VND) are considered for analysis. This study aims to provide stakeholders with actionable insights for informed decision-making in the Vietnamese stock market.

INDEX TERMS Vietnam , Stock prices , LR , ARIMA , RNN , GRU , LSTM , SEMOS , Stacking , FCN

I. INTRODUCTION

Stocks represent a vital component of Vietnam's economic landscape, serving as a cornerstone for economic development and fostering investor participation. The stock market not only facilitates capital infusion into enterprises, enabling their expansion and innovation but also serves as a lucrative avenue for investors to seek returns on their investments.

Recognizing the pivotal role of stocks in driving economic progress, our endeavor focuses on leveraging advanced statistical, machine learning, and deep learning algorithms to predict stock prices in the Vietnamese market, with a particular emphasis on Vietnam Airlines JSC (HVN), SCSC Cargo Service Corporation (SCS), and Vietjet Aviation Joint Stock Company (VJC).

There are numerous amounts of time-series models, this project uses eight time-series models to predict stock price in Vietnam: Linear regression, ARIMA, RNN, GRU, LSTM, SEMOS, Stacking, FCN.

We evaluate predictive models through a comprehensive analysis based on multiple criteria such as MAPE, RMSE, MSLE, and the results of data division methods. Through this process, we determine whether this model is good or not, which model should be used, which model should not be used to estimate stock prices, providing stakeholders with clear insights and decision-making support tools.

II. RELATED WORKS

Due to the profitability and importance of stocks, many methods have been developed to predict stock prices. Here, we will utilize 8 algorithms: Stacking, FCN, SEMOS, Linear regression, ARIMA, RNN, GRU and LSTM. Below are some related research papers on stock price prediction using these 8 algorithms.

In the study by Philip Ngare, Dennis Ikpe, and Samuel Asante Gyamerah, three models were employed: AdaBoost, KNN, and Stacking (AdaBoost and KNN serve as base-level classifiers, and GBM acts as the meta-level classifier). They utilized a dataset obtained from the Nairobi Stock Exchange, and the results showed that the Stacking model outperformed the two individual models, AdaBoost and KNN. This demonstrates that combining different models using the stacking ensemble learning method can lead to better results[1].

In the research by Shima Nabiee and Nader Bagherzadeh, they compared the performance of six different models, including FCN, SegNet, U-Net, DeepLab V3+, and two proposed models with 20-day and 40-day input frames, respectively. The results showed that when using price frames as input, FCN performed extremely well and ranked second among the six models. Notably, when the input was changed to trends, the FCN model achieved the best results among all six models, demonstrating that FCN is very suitable for stock prediction. Additionally, it was observed that when the input

frame is 1 (40 days), it provides better short-term prediction results, and when the number of input frames increases, the long-term results are improved [2].

The research of David Jobst, Annette Möller, and Jürgen Groß has shown that SEMOS is used to enhance the forecasting performance of numerical weather prediction (NWP) models by employing finite Fourier series, making it a clearly seasonal and trend-aware time series model. The results show that SEMOS provides more accurate weather forecasts based on evaluation metrics such as CRPS, LogS, and RMSE. Additionally, SEMOS maintains good forecasting performance across different time horizons. These advantages indicate that SEMOS could be a good choice for applying to stock market forecasting[3].

In the study by Xiwen Jin and Chaoran Yi, they conducted a comparison of the effectiveness among 6 models: LSTM, GRU, RandomForest, XGBoost, LightGBM, and Linear Regression, and found that GRU yielded the best results, followed by LSTM and Linear Regression. The R2 scores were as follows: LSTM 0.84, GRU 0.86, Linear Regression 0.73, and the MSE scores were: LSTM 7.06, GRU 6.26, Linear Regression 6.64 [4]. In another research by Dias Satria, four models were employed: ARIMA, RNN, LSTM, and GRU. The results showed that GRU exhibited the best performance in predicting stock prices, followed by LSTM and RNN, while ARIMA was deemed unsuitable due to the nonlinear characteristics of the data, violating the assumption of white noise in the estimation of ARIMA Box-Jenkins parameters [5].

III. MATERIALS

A. DATASET

The historical stock price of Vietnam Airlines JSC (HVN), SCSC Cargo Service Corporation (SCS) and Vietjet Aviation Joint Stock Company (VJC) from 01/03/2019 to 29/02/2024 will be applied. The data contains column such as Date, Open, High, Low, Close, Volume. As the goal is to forecast close prices, only data relating to column “close" (VND) will be processed.

B. DESCRIPTIVE STATISTICS

TABLE 1. HVN, SCS, VJC's Descriptive Statistics

	HVN	SCS	VJC
Count	1,245	1,253	1,252
Mean	20,266	63,678	117,882
Std	6,462	8,429	14,232
Min	8,610	37,320	93,800
25%	13,500	59,500	105,500
50%	21,011	64,629	117,400
75%	25,000	67,350	129,000
Max	34,753	85,420	149,000

FIGURE 1. HVN stock price's boxplot

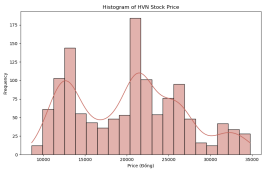

FIGURE 2. HVN stock price's histogram

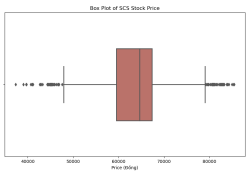

FIGURE 3. SCS stock price's boxplot

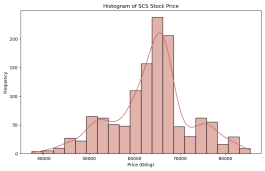

FIGURE 4. SCS stock price's histogram

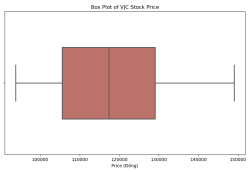

FIGURE 5. VJC stock price's boxplot

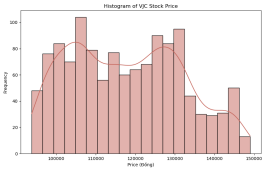

FIGURE 6. VJC stock price's histogram

IV. METHODOLOGY

A. LINEAR REGRESSION

Linear Regression is a method of statistical analysis used to determine the relationship between a dependent variable and one or many independent variables. It is used to predict the value of the dependent variable based on the value of the independent variables. A multiple linear regression model has the form:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \varepsilon \quad [12]$$

Where:

- Y is the dependent variable.
- $X_1, X_2, \ldots, X_k$ are the independent variables.
- β_0 is the intercept term.
- $\beta_1, ..., \beta_k$ are the regression coefficients for the independent variables.
- ε is the error term.

B. ARIMA

ARIMA(Autoregressive Integrated Moving Average) is a statistical forecasting method widely used in time series analysis. This model incorporates autoregressive (AR), moving average (MA) and Integrated (I) components to capture the relationship between current and past values of a time series. Auto Regression (AR):

$$y_t = \mu + \sum_{i=1}^{p} \gamma_i y_{t-i} + \epsilon_t \quad [6]$$

In there : y_t is the current value; μ is the constant term; p is the number of autoregressive terms; γ_i is the autocorrelation coefficient and ε_t is error.
Moving Average (MA):

$$y_t = \mu + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} + \epsilon_t \quad [6]$$

In there : y_t is the current value; μ is the constant term; q is the number of terms in the moving average; θ_i is the moving average coefficient and ε_t is error.
Integrated (I):

$$\mathrm{I}(d=0) : \Delta y_t = y_t$$

$$\mathrm{I}(d=1) : \Delta y_t = y_t - y_{t-1}$$

$$\mathrm{I}(d=2) : \Delta\left(\Delta y_t\right) = \left(y_t - y_{t-1}\right) - \left(y_{t-1} - y_{t-2}\right)$$

In there : d is the number of differences required to make it a stationary sequence

After combining them, we will have the ARIMA (p, d, q) express as follow:

$$y_t = \mu + \sum_{i=1}^{p} \gamma_i y_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} + \epsilon_t \quad [6]$$

C. SEMOS

SEMOS (Smooth EMOS) is a statistical approach used for post-processing ensemble forecasts, particularly to handle seasonal variations in the predictive distribution parameters. SEMOS integrates seasonal effects directly into the estimation of location and scale parameters, enhancing the accuracy of ensemble forecasts, especially for quantities exhibiting seasonal variability.

The first parameter is location, in general, represents the central tendency or the "location" of the predictive distribution.In SEMOS, the location parameter is modeled as a function of the ensemble mean forecast and seasonal effects in order to capture systematic biases or trends in the ensemble forecasts:

$$\mu_S(t) = a_0 + f_0(t) + (a_1 + f_1(t)).x(t) \quad [3]$$

Where:

- $\mu_S(t)$ is the location parameter at time t.
- a_0, a_1 are coefficients representing the baseline intercept and slope.
- $f_0(t), f_1(t)$ are seasonal effects modeled using cyclic regression splines.
- $x(t)$ is the ensemble mean forecast at time t

The scale parameter represents the spread or "scale" of the predictive distribution.It accounts for variations in forecast uncertainty over time, including factors such as model errors and ensemble spread, modeled as a function of the empirical ensemble standard deviation and seasonal effects.

$$log(\sigma_S(t)) = b_0 + g_0(t) + (b_1 + g_1(t)).s(t) \quad [3]$$

Where:

- $\sigma_S(t)$ is the scale parameter at time t.
- b_0, b_1 are coefficients representing the baseline intercept and slope.
- $g_0(t), g_1(t)$ are seasonal effects modeled using cyclic regression splines.
- $s(t)$ is the empirical ensemble standard deviation at time step t.

D. STACKING

Stacking, short for Stacked Generalization, is a machine learning algorithm belonging to the Ensemble Learning category. A basic Stacking model is usually divided into two levels: level-0 models and the level-1 model. Level-0 models (base-models) are the foundational models that learn directly from the dataset and produce predictions for the level-1 model (meta-model). The meta-model is trained based on the predicted outputs of the base models. These outputs, combined with the labels, form the input-output pairs during the training process of the meta-model. In this work, ARIMAX and RNN are used as base-models and Linear Regression is selected as meta-model.

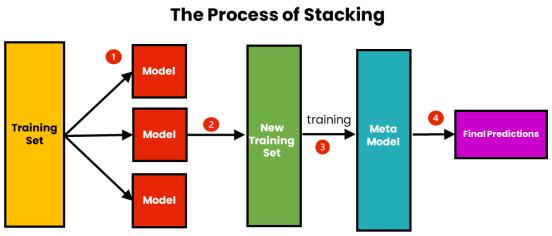

FIGURE 7. Stacking Architecture [7]

E. RNN

RNN (Recurrent Neural Network) is a type of artificial neural network with feedback connections closed by loop. The looping structure allows the network to store past information in the hidden state and use that past information to improve the performance of the network. In figure 8, the inputs x_t will be combined with the hidden layer h_{t-1} using an activation function to compute the current hidden layer h_t.

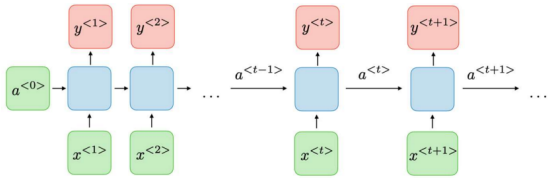

FIGURE 8. RNN Architecture [8]

Forward propagation :

$$a^{<t>} = g_1\left(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a\right) \quad [9]$$

$$y^{<t>} = g_2\left(W_{ya}a^{<t>} + b_y\right) \quad [9]$$

Where:
- $x^{<t>}$ is the input value at time step t
- $a^{<t>}$ is the state at time step t
- $y^{<t>}$ is the output value at time step t
- $\mathrm{W_{aa}}, \mathrm{W_{ax}}, \mathrm{W_{ya}}$ are weights
- b_a và b_y are bias
- $\mathrm{g_1}$ is activation function(e.g., tanh, ReLU)
- $\mathrm{g_2}$ is activation function(e.g., softmax)

Backward propagation :

$$L = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad [9]$$

$$W_{ay}^{(t+1)} = W_{ay}^{(t)} - \eta\frac{\partial \mathcal{L}}{\partial W_{ay}} \quad [9]$$

$$W_{aa}^{(t+1)} = W_{aa}^{(t)} - \eta\frac{\partial \mathcal{L}}{\partial W_{aa}} \quad [9]$$

$$W_{xa}^{(t+1)} = W_{xa}^{(t)} - \eta\frac{\partial \mathcal{L}}{\partial W_{xa}} \quad [9]$$

$$b_{y}^{(t+1)} = b_{y}^{(t)} - \eta\frac{\partial \mathcal{L}}{\partial b_{y}} \quad [9]$$

$$b_{h}^{(t+1)} = b_{h}^{(t)} - \eta\frac{\partial \mathcal{L}}{\partial b_{h}} \quad [9]$$

Where:
- y_i is actual value of the i-th element
- $\hat{y}_i$ is predicted value of the i-th element
- η is learning rate

F. LSTM

Long Short-Term Memory (LSTM) is an advanced variant of recurrent neural network (RNN) architecture used in the field of deep learning. Its goal is to give RNN a "long short-term memory" — a short-term memory that can endure thousands of timesteps.

A vanilla LSTM unit is composed of a cell, an input gate, an output gate and a forget gate.

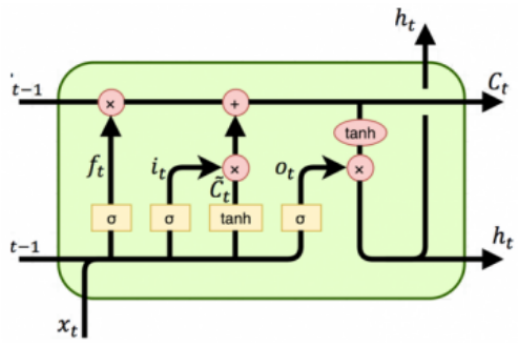

FIGURE 9. LSTM Model At Time Step t [10]

Where:
- Input gate: conditionally decides which values from the input to update the memory state.
- Output Gate: conditionally decides what to output based on input and the memory of the block.
- Forget Gate: conditionally decides what information to throw away from the block.

The update equations for the LSTM unit are expressed by equation below:

$$h^{(t)} = g_0^{(t)} f_h(s^{(t)}) \quad [10]$$

$$s^{(t-1)} = g_f^{(t)} s^{(t-1)} + g_i^{(t)} f_s(wh^{(t-1)}) + uX^{(t)} + b \quad [10]$$

$$g_i^{(t)} = sigmold(w_i h^{(t-1)} + u_i X^{(t)} + b_i \quad [10]$$

$$g_f^{(t)} = sigmold(w_f h^{(t-1)} + u_f X^{(t)} + b_f \quad [10]$$

$$g_f^{(t)} = sigmold(w_o h^{(t-1)} + u_o X^{(t)} + b_o \quad [10]$$

where f_h and f_s represent the activation functions of the system state and internal state, typically utilizing the hyperbolic tangent function.

G. GRU

GRU stands for Gated Recurrent Unit, which is a type of recurrent neural network (RNN) architecture that is similar to LSTM (Long Short-Term Memory). This means that GRU also have the input gate, output gate and forget gate. The differences are that GRU combines the input and forget gate into a single update gate, resulting in a more streamlined design and separate cell state is not included in GRU.

A GRU unit consists of three main components: an update gate, a reset gate and the current memory content.

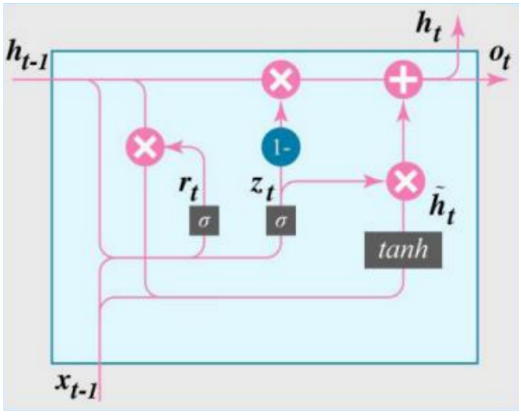

FIGURE 10. GRU Model At Time Step t [10]

The update gate determines how much of the past information should be retained and combined with the current input at a specific time step.

$$z_t = \sigma(W_z[h_{t-1}, x_t] + b_z) \quad [10]$$

The reset gate decides how much of the past information should be forgotten.

$$r_t = \sigma(W_r[h_{t-1}, x_t] + b_r) \quad [10]$$

The current memory content is computed based on the reset gate and the concatenation of the transformed previous hidden state and the current input.

$$\tilde{h}_t = \tanh(W_h[r_t h_t, x_t]) \quad [10]$$

The final memory state h_t is determined by a combination of the previous hidden state and the candidate activation.

$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t \quad [10]$$

Finally, the output gate is computed using the current memory state h_t and is typically followed by an activation function, such as the sigmold function.

$$o_t = \sigma_o(W_o h_t + b_o) \quad [10]$$

H. FCN

Fully Convolutional Neural Networks (FCNs) were first proposed in Wang et al. (2017b) for classifying univariate time series and validated on 44 datasets from the UCR/UEA archive. FCNs are mainly convolutional networks that do not contain any local pooling layers which means that the length of a time series is kept unchanged throughout the convolutions. In addition, one of the main characteristics of this architecture is the replacement of the traditional final FC layer with a Global Average Pooling (GAP) layer which reduces drastically the number of parameters in a neural network while enabling the use of the CAM (Zhou et al., 2016) that highlights which parts of the input time series contributed the most to a certain classification. [11]

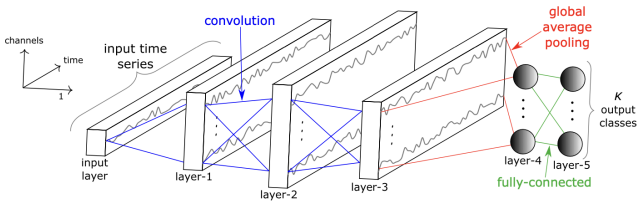

FIGURE 11. Fully Convolutional Neural Network architecture [11]

V. RESULT

A. EVALUATION METHODS

Root Mean Squared Error (RMSE): is the square root of average value of squared error in a set of predicted values.

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

Mean Absolute Percentage Error (MAPE): is the average percentage error in a set of predicted values.

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

Mean Squared Logarithmic Error (MSLE):is the relative difference between the log-transformed actual and predicted values.

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

Where:

- n is the number of observations in the dataset.
- y_i is the true value.
- $\hat{y}_i$ is the predicted value.

B. HVN DATASET

HVN Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MSLE
LR	**7:3**	**3993.14**	24.3741	**0.07304**
	8:2	4163.78	**13.5631**	0.07678
	9:1	5940.92	22.6138	0.15792
ARIMA	7:3	3581.65	16.3043	0.06087
	8:2	**3341.08**	**10.9181**	**0.04085**
	9:1	5375.16	19.9011	0.11742
SEMOS	**7:3**	**610.171**	**2.65241**	**0.00141**
	8:2	1616.817	1.267	0.00035
	9:1	1699.655	1.052	0.00032
Stacking	**7:3**	**197.562**	**0.92907**	**0.00015**
	8:2	240.401	1.01158	0.00021
	9:1	434.257	1.59473	0.00038
RNN	**7:3**	**610.171**	**2.65241**	**0.00141**
	8:2	776.437	2.90091	0.00179
	9:1	1254.82	4.54483	0.00329
FCN	7:3	2523.85	17.0891	0.04537
	8:2	**990.824**	5.53287	0.00654
	9:1	1226.98	**5.30376**	**0.00492**
GRU	**7:3**	**424.191**	**1.74601**	**0.00069**
	8:2	440.368	2.00419	0.00072
	9:1	880.814	3.36061	0.00171
LSTM	**7:3**	**517.393**	**2.26375**	**0.00108**
	8:2	603.747	2.45307	0.00126
	9:1	851.073	3.27205	0.00162

TABLE 2. HVN Dataset's Evaluation

FIGURE 12. Linear model's result with 7:3 splitting proportion

FIGURE 13. ARIMA model's result with 8:2 splitting proportion

FIGURE 14. SEMOS model's result with 9:1 splitting proportion

FIGURE 15. Stacking model's result with 7:3 splitting proportion

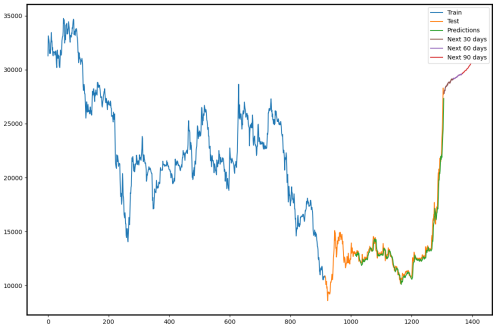

FIGURE 16. RNN model's result with 7:3 splitting proportion

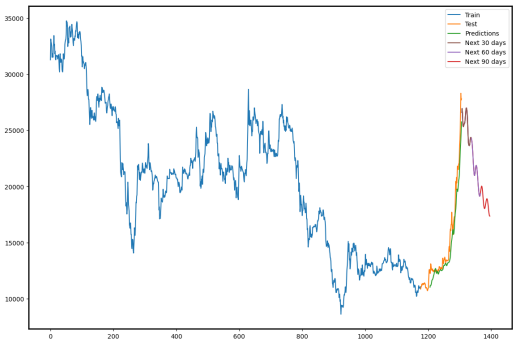

FIGURE 17. FCN model's result with 9:1 splitting proportion

FIGURE 18. GRU model's result with 7:3 splitting proportion

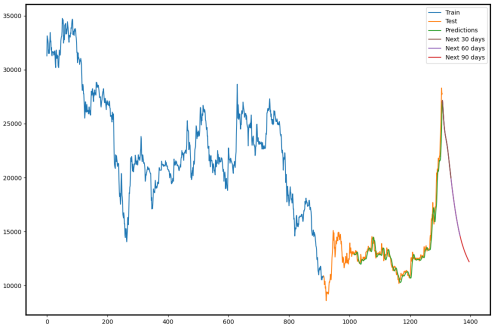

FIGURE 19. LSTM model's result with 7:3 splitting proportion

C. SCS DATASET

SCS Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MSLE
LR	7:3	9905.85	13.9523	0.02007
	8:2	**7788.89**	10.3625	**0.01199**
	9:1	8985.64	**9.69887**	0.01395
ARIMA	**7:3**	**7633.15**	**6.23802**	**0.01103**
	8:2	11243.2	10.7119	0.02545
	9:1	13349.1	12.7524	0.03356
SEMOS	**7:3**	**1545.676**	**1.262**	**0.00033**
	8:2	1616.817	1.267	0.00035
	9:1	1699.655	1.052	0.00032
Stacking	**7:3**	**457.389**	**0.48271**	**0.00004**
	8:2	532.678	0.53753	0.00005
	9:1	678.649	0.64756	0.00006
RNN	**7:3**	**1441.73**	**1.41166**	**0.00038**
	8:2	1576.35	1.60094	0.00046
	9:1	1715.44	1.60622	0.00041
FCN	7:3	**1755.93**	2.12337	0.00064
	8:2	1765.97	**1.91911**	**0.00058**
	9:1	2607.35	2.81101	0.00112
GRU	**7:3**	**946.301**	**0.88267**	**0.00017**
	8:2	1257.21	1.14076	0.00028
	9:1	1475.51	1.31043	0.00031
LSTM	7:3	1299.93	1.34468	0.00032
	8:2	**1260.76**	**1.16337**	**0.00028**
	9:1	2390.65	2.37200	0.00081

TABLE 3. SCS Dataset's Evaluation

FIGURE 20. Linear model's result with 8:2 splitting proportion

FIGURE 21. ARIMA model's result with 7:3 splitting proportion

FIGURE 22. SEMOS model's result with 9:1 splitting proportion

FIGURE 23. Stacking model's result with 7:3 splitting proportion

FIGURE 24. RNN model's result with 7:3 splitting proportion

FIGURE 25. FCN model's result with 8:2 splitting proportion

FIGURE 26. GRU model's result with 7:3 splitting proportion

FIGURE 27. LSTM model's result with 8:2 splitting proportion

D. VJC DATASET

VJC Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MSLE
LR	7:3	21720.7	20.7604	0.03762
	8:2	12454.9	11.3989	0.01373
	9:1	**4220.91**	**2.76031**	**0.00149**
ARIMA	7:3	7056.66	5.94446	0.00465
	8:2	7807.39	5.87333	0.00579
	9:1	**4857.61**	**4.07812**	**0.00203**
SEMOS	**7:3**	**1545.676**	**1.262**	**0.00033**
	8:2	1616.817	1.267	0.00035
	9:1	1699.655	1.052	0.00032
Stacking	**7:3**	**791.931**	**0.56862**	**0.00005**
	8:2	859.683	0.57597	0.00006
	9:1	985.981	0.66053	0.00007
RNN	**7:3**	**2153.31**	**1.49846**	**0.00042**
	8:2	2332.43	1.40122	0.00047
	9:1	3634.93	2.16684	0.00107
FCN	7:3	5254.80	4.36589	0.00263
	8:2	2682.99	2.05971	0.00065
	9:1	**2646.81**	**1.53967**	**0.00057**
GRU	**7:3**	**1502.23**	**0.95296**	**0.00021**
	8:2	1697.52	0.99987	0.00025
	9:1	2567.96	1.47789	0.00054
LSTM	7:3	1793.41	1.24209	0.00029
	8:2	**1726.18**	**1.03356**	**0.00026**
	9:1	2684.56	1.71757	0.00058

TABLE 4. VJC Dataset's Evaluation

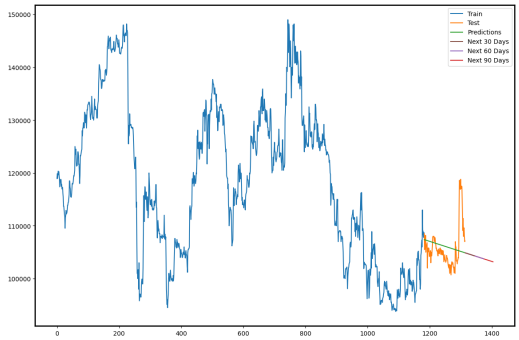

FIGURE 28. Linear model's result with 9:1 splitting proportion

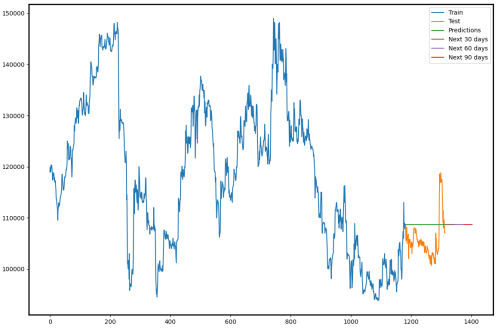

FIGURE 29. ARIMA model's result with 9:1 splitting proportion

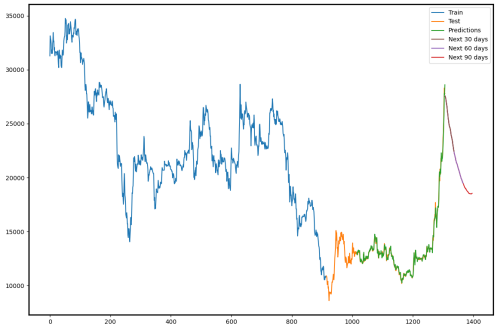

FIGURE 30. SEMOS model's result with 9:1 splitting proportion

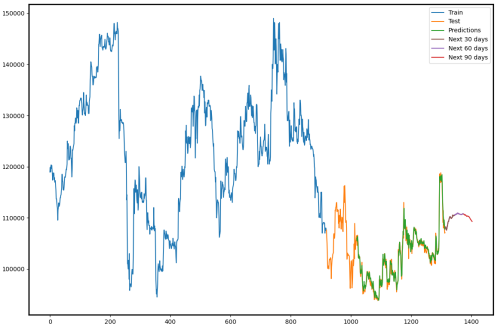

FIGURE 31. Stacking model's result with 7:3 splitting proportion

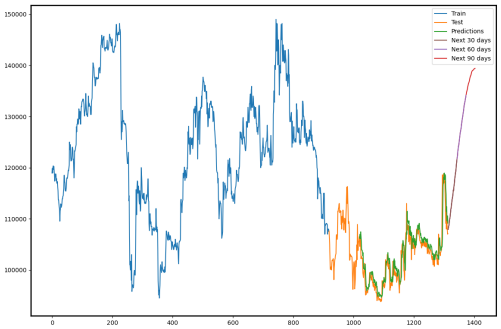

FIGURE 32. RNN model's result with 7:3 splitting proportion

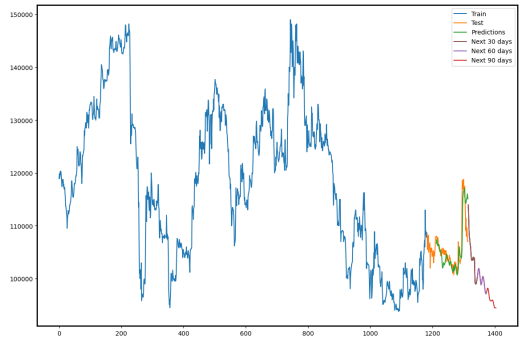

FIGURE 33. FCN model's result with 9:1 splitting proportion

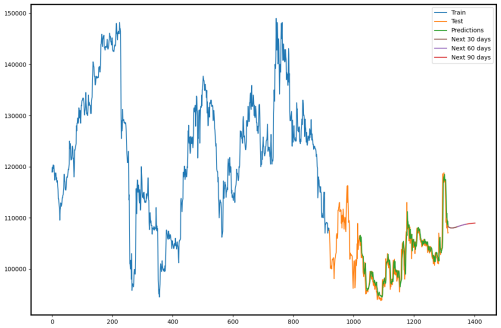

FIGURE 34. GRU model's result with 7:3 splitting proportion

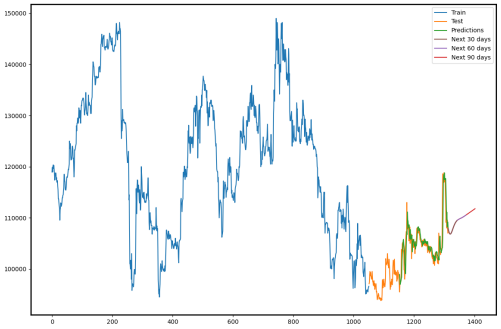

FIGURE 35. LSTM model's result with 8:2 splitting proportion

VI. CONCLUSION

A. SUMMARY

In the pursuit of forecasting stock prices, a variety of methodologies have been explored, ranging from traditional statistical models to advanced machine learning algorithms. Among the models applied, Linear Regression (LR), Auto Regressive Integrated Moving Average (ARIMA), Recurrent Neural Networks (RNN), Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), Seasonal Exponential Smoothing (SEMOS), Stacking, and Fully Convolutional Networks (FCN), it is evident that GRU, LSTM, and Stacking emerge as the most promising and effective models for predicting stock prices.

The complexities of stock price forecasting, rooted in the intricacies and unpredictability of financial markets, demand models capable of capturing nuanced patterns and relationships within the data. GRU models exhibit notable performance in forecasting stock prices by capturing sequential dependencies. LSTM models further enhance this capability by effectively handling long-term dependencies, providing robust predictions. The introduction of ensemble learning through Stacking refines the predictive capabilities even further, offering collective insights that surpass individual models.

As evidenced by the evaluation metrics, including RMSE, MAPE, and MSLE, the GRU, LSTM, and Stacking models consistently demonstrate superior performance across various aspects of forecasting accuracy. Their adaptability to handle the inherent uncertainties of stock markets positions them as formidable tools for investors and analysts seeking reliable predictions.

B. FUTURE CONSIDERATIONS

In our future research, it is crucial to prioritize further optimization of the previously mentioned models. This optimization effort should specifically focus on: • Enhancing the accuracy of the model. While the above algorithms have demonstrated promising results in predicting stock prices, there is a need to further improve the model's accuracy to ensure more precise forecasting outcomes. • Exploring alternative machine learning algorithms or ensemble techniques. Ensemble techniques, such as combining multiple models or using various ensemble learning methods, can also improve the robustness and accuracy of the forecasts. • Researching new forecasting models. The field of forecasting continuously evolves, with new algorithms and models being researched and developed. It is crucial to stay updated with these approaches and explore new forecasting models that offer improved accuracy and performance.

By continuously exploring and incorporating new features, data sources, and modeling techniques, we can strive for ongoing optimization of the forecasting models and enhance their ability to predict stock prices with greater precision and reliability.

ACKNOWLEDGMENT

First and foremost, we would like to express our sincere gratitude to **Assoc. Prof. Dr. Nguyen Dinh Thuan** and **Mr. Nguyen Minh Nhut** for their exceptional guidance, expertise, and invaluable feedback throughout the research process. Their mentorship and unwavering support have been instrumental in shaping the direction and quality of this study. Their profound knowledge, critical insights, and attention to detail have significantly contributed to the success of this research.

This research would not have been possible without the support and contributions of our mentors. We would like to extend our heartfelt thanks to everyone involved for their invaluable assistance, encouragement, and belief in our research. Thank you all for your invaluable assistance and encouragement.

REFERENCES

[1] S. A. Gyamerah, P. Ngare and D. Ikpe, "On Stock Market Movement Prediction Via Stacking Ensemble Learning Method," 2019 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr), Shenzhen, China, 2019, pp. 1-8, doi: 10.1109/CIFEr.2019.8759062.

[2] S. Nabiee và N. Bagherzadeh, "Stock Trend Prediction: A Semantic Segmentation Approach," Mar 2023, doi: 10.48550/arXiv.2303.09323.

[3] D. Jobst, A. Möller, and J. Groß, "Time Series based Ensemble Model Output Statistics for Temperature Forecasts Postprocessing", Feb 2024, doi: 10.48550/arXiv.2402.00555.

[4] Xiwen Jin and Chaoran Yi, "The Comparison of Stock Price Prediction Based on Linear Regression Model and Machine Learning Scenarios," in Proceedings of the 2022 International Conference on Bigdata Blockchain and Economy Management (ICBBEM 2022), December 2022, pp. 837-842, doi: 10.2991/978-94-6463-030-5_82.

[5] D. Satria, "Predicting Banking Stock Prices Using RNN, LSTM, and GRU Approach," Applied Computer Science, vol. 19, no. 1, pp. 82-94, March 2023, doi: 10.35784/acs-2023-06

[6] S. Kumar, A. Gupta, K. Arora, and K. Vatta, "Effect of Rainfall in Predicting Tomato Prices in India: An Application of SARIMAX and NARX Model" December 2022, vol. 32, no. 2, pp. 159-164

[7] B. Soni, "Stacking to Improve Model Performance: A Comprehensive Guide on Ensemble Learning in Python", Medium. Accessed: May 12, 2024. [Online]. Available: https://medium.com/@brijesh_soni/stacking-to-improve-model-performance-a-comprehensive-guide-on-ensemble-learning-in-python-9ed53c93ce28

[8] A. Amidi and S. Amidi, "Recurrent Neural Networks cheatsheet" Stanford University. [Accessed: May 25, 2024]. [Online]. Available: https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks

[9] Nguyen Minh Nhut, “Mo Hinh Mang Hoi Quy (RNN) trong chuoi thoi gian”

[10] Farhad Mortezapour Shiri, Thinagaran Perumal, Norwati Mustapha and Raihani Mohamed, "A Comprehensive Overview and Comparative Analysis on Deep Learning Models: CNN, RNN, LSTM, GRU", May 2023, doi: 10.48550/arXiv.2305.17473

[11] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," Data Mining and Knowledge Discovery, vol. 33, no. 4, pp. 917-963, Jul. 2019, doi: 10.1007/s10618-019-00619-1.

[12] D. C. Montgomery, E. A. Peck, and G. G. Vining, "Introduction to Linear Regression Analysis," 5th ed., Hoboken, NJ, USA: Wiley, 2012.