International Journal of Applied Sciences: Current and Future Research Trends (IJASCFRT)
ISSN (Print) , ISSN (Online)

https://ijascfrtjournal.isrra.org/index.php/Applied_Sciences_Journal

Stacked Long Short-Term Memory for Vietnamese Stock Market Returns Prediction

Phuong Mai Pham[a], Quang Chieu Ta[b*]

[a,b]*Thuyloi University, 175 Tay Son, Dong Da, Hanoi 100000, Vietnam*
[a]*Email: maihoamieu@gmail.com,* [b]*Email: quangchieu.ta@tlu.edu.vn*

Abstract

Investment wealth management and macroeconomic research both use historical indices that describe the overall movements of the entire market over a period to make objective judgments and insightful decisions. Since their inception, exchange stock indices have provided a broad picture of the national economy, reflected various stages of economic development, and forecasted potential risks that could harm investments or lead to a large-scale crisis. Many articles have studied the volatility of stock prices and stock indexes in the world, but the separate research on the profitability of the whole stock market is relatively scarce. This paper focuses on studying the profitability of the Vietnamese stock market from 2010 to 2021 through the returns of the VN-Index. The aim of this paper is to propose a Stacked Long Short-Term Memory (LSTM) neural network to predict the volatility of VN-Index returns, in comparison with the autoregressive methods (ARIMA, SARIMA) often used in econometrics. Taking Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) as the objective cost functions, the research results show that the LSTM neural network model gives superior performance compared to the econometric methods and better predictive meaning in practice thanks to its flexible learning and memory capabilities.

Keywords: LSTM; Stock Market; Stock Returns.

1. Introduction

Financial economics has a long history of research into the predictability of stock markets, as they can have significant influence on the global economy. Some researchers compromise with the "efficient market hypothesis", which states that stock price changes are independent of the past and behave similarly to a random walk [1,2]. In fact, the financial time series data is noisy and often non-stationary in nature. They often possess rather turbulent structures and high volatility, which hardly follow reliable patterns to be captured.

---

* Corresponding author.

However, recently, many studies using modern technical analysis on historical financial data have been conducted and shown optimistic results. With the rapid advancement in data mining and computing technology, novel methods have been implemented to identify trends and patterns from a tremendous amount of available real-time stock data. The most frequently used approaches for stock market prediction are econometric or statistical methods, which can be subdivided into linear and nonlinear categories.

Exponential smoothing methods [3], regression methods [4], autoregressive integrated moving average methods (ARIMA) [5] can be considered a linear method, that is, methods that employ linear functional form for time series modeling; threshold methods [6], generalized autoregressive conditionally heteroskedastic methods (GARCH) [7, 8] are nonlinear methods. However, due to various statistical assumptions of the market frameworks, such as linearity and normality, and since the models' success depends mainly on the validity of these assumptions, these approaches have limitations when applied to the complex real-world financial data.

In recent years, artificial neural networks (ANNs) have acquired a lot of attention because of their excellent performance in picture classification and natural language processing (NLP) and various time series problems [9, 10]. Deep learning is the process of training several layers of neural networks with additional layers. It is a collection of novel structures and methods that may be experimented with and improved. Multilayer perceptron (MLP), Convolutional neural network (CNN), Recurrent neural network (RNN), Long short-term memory neural network (LSTM), Residual network (ResNet), Transformers neural network, and other new structures are examples. In the field of finance, there has been growing interest in whether deep learning can be effectively applied to forecast stock market movements with various target outputs. The most selected outputs are stock price [11, 12] market direction (up or down) [13], volatility [14], market trends (bull/bear-flag) [10], portfolio composition (cash: stock), stock index [15], stock returns [13], return ranks (divided by 25%) [16], etc. Applying the adaptive, self-organizing deep learning models to solve financial problems is challenging because it requires a combination of strong technical analysis, financial theory and economic analysis, time series analysis and basic analysis [17]. Besides, to obtain a good predictive ability with ANNs, input variables and network hyperparameters such as the learning rate, number of hidden layers, and number of nodes in each layer must be carefully selected [18].

A stock index, sometimes known as a stock market index, is a financial index that monitors a stock market, or a subset of the stock market, and allows investors to compare current price levels to previous price levels to determine market performance [19]. Due to the volatility and noise features, stock index return prediction has long been one of the most difficult tasks for people who operate in the financial field and other associated industries, as it is a type of financial time series with a poor signal-to-noise ratio and a fat-tailed distribution [20]. Most research on stock returns concentrates on the binary classification (up or down). Not much attention is paid to the distribution and predictability by regression of stock returns, especially stock market returns, which play a crucial role in the implication of the national economic situation associated with macroeconomic factors and in the comparison of regional or global market indices.

In this paper, we select the dataset of the Vietnam Stock Index (VN-Index), a major stock market index which tracks the performance of 303 equities listed on the Ho Chi Minh Stock Exchange in Vietnam. The feature

vectors in our financial time series are the closing price, the opening price, the lowest price, the highest price, and the transaction volume of historical data, like other current typical stock datasets. We developed a Stacked Long Short-Term Memory Neural Network (Stacked LSTM), whose hyperparameters are tuned both manually and with a grid-search to minimize the cost functions. To provide objective results, we compare the model with the Autoregressive Integrated Moving Average Method (ARIMA) and the Seasonal Autoregressive Integrated Moving Average Method (SARIMA), which are optimized by an auto grid-search Akaike Information Criteria (AIC) minimizing techniques as representatives for statistical prediction methods. Our evaluation metrics are Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). We also carefully split our time series into multiple datasets to carry out a nested cross-validation.

The paper is structured as follows: in Sec. 2 we provide a demonstration of the methods that we implement in predicting stock index returns; then, in Sec. 3, we discuss input data for algorithms, feature selection, preprocessing; provide the computational results related to the training processes, based on the VN-Index time series; and we provide a numerical interpretation of the results obtained exploiting the aforementioned approaches, and then commenting them; in Sec. 4, we make a brief conclusion and provide scopes for future research.

2. Model Description

This section is centered on the presentation of the models that are the subjects of our comparison. We examine some of the descriptive material on recurrent neural networks (RNNs), with a focus on a type of neural networks that is directly linked to our research: Long Short-Term Memory neural networks (LSTM).

Long Short-Term Memory (LSTM) networks are an artificial recurrent neural network (RNN) extension that is meant to learn sequence (temporal) data and their long-term relationships more precisely than ordinary RNNs. Traditional RNNs have problems with vanishing gradients and short-term memory, whereas these networks do not. It's widely utilized in deep learning applications including stock market forecasting, handwriting identification, speech recognition, natural language processing, and so forth [21]. Long short-term memory refers to networks that can recall short-term memories for an extended period.

The basic property of a Recurrent Neural Network (RNN) is that it has at least one feed-back link, allowing activations to loop back on themselves. This allows the networks to do temporal processing and learn sequences, such as sequence recognition/reproduction or temporal association/prediction, for example. Recurrent neural network designs come in a variety of shapes and sizes. A conventional Multi-Layer Perceptron (MLP) with additional loops is one popular form. These can take use of the MLP's strong non-linear mapping capabilities while also having some sort of memory. The RNN contains weight matrices: weight matrix U, which links input-to-hidden weight matrix W, which connects hidden-to-hidden, and weight matrix V, which connects hidden-to-output.

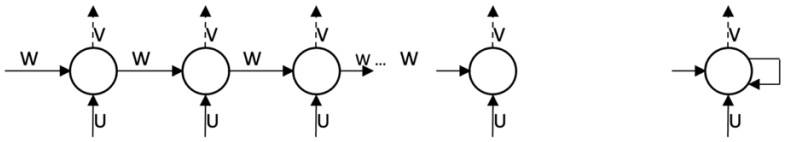

Figure 1 : RNN Structure.

The model has n inputs and n output, which are given in a sequence (e.g., time series). Each time step is fed with x_t as input and h_{t-1} as the output of the previous state. With f and g activation functions, h_t is given as $h_{t1} = f(Ux_t + Wh_{t-1})$, the predicted value is given as: $\hat{y}_t = g(Vh_t)$.

Since RNN has difficulty in learning long time-dependencies that are more than a few time steps in length with the vanishing gradient problem [22], LSTM is proposed to fulfill the drawbacks of RNN. The LSTM controls neurons and stores information using many functionally distinct gates, allowing it to store essential information for longer periods of time. The information is obtained using the activation function's dot product. The gradient descent approach is used to train a set of parameters for each gate state.

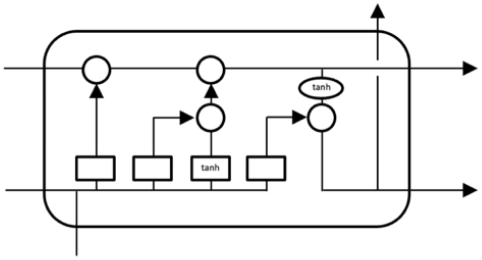

Figure 2: Structure of the LSTM Network**.**

The construction of the LSTM network is depicted in detail in Figure 3. Each gate in the LSTM network serves a distinct purpose. The forget gate f_t determines which information from the previous state h_{t-1} should be discarded. The revised forget gate f_t, along with the input x_t and prior state h_{t-1} following the update gate i_t operation, determine how much weight the candidate state C_{t-1} should use to update state C_t. A nonlinear function $tanh$ is used to filter the current state of output, and it is then returned following the output gate o_t operation. The next input h_{t-1} is obtained from the returning partial state h_t. Each gate is based on the previous output h_{t-1} and the current external input x_t. The updated process of the LSTM network is given by equations:

Forget gate: $f_t = \sigma\left(U_f x_t + W_f h_{t-1} + b_f\right)$

Candidate state: $\tilde{C}_t = \tanh\left(U_f x_t + W_f h_{t-1} + b_f\right)$

Update state: $i_t = \sigma(U_i x_t + W_i h_{t-1} + b_i)$

Cell state: $h_t = i \times \tilde{C}_t + f_t \times h_{t-1}$

Output gate: $o_t = \sigma(U_o x_t + W_o h_{t-1} + b_o)$

Output: $h_t = o_t \times tanh(C_t)$

Where x_t is the input vector at t time, U_f, U_i, and U_o are the weight matrices associated with the input units; W_f, W_i, and W_o are the weight matrices connecting hidden layers; b_f, b_i, and b_o are bias vectors, σ is the sigmoid function, $tanh$ is the hyperbolic tangent function.

The Stacked LSTM model consists of n LSTM layers and one fully connected layer. To reduce the model dimensions, the nth LSTM layer outputs a one-dimensional vector.

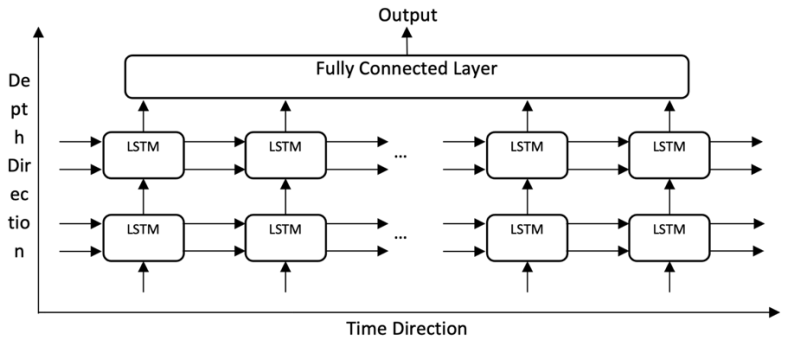

Figure 3: Architect of the Stacked Long Short-Term Memory Network.

Forward computation and back propagation are the two operations that make up the Stacked LSTM network. Like other ANN, the forward calculation is given by:

$$\hat{y}_t = Vf\left(UX_t + Wf\left(UX_{t-1} + Wf(UX_{t-2} + \ldots)\right)\right)$$

where U is the weight matrix of input X, W is the weight matrix of this input from the previous state h_{h-1}. f is the activation function, and V is the weight matrix of the output layer. The back propagation over time technique is also used in the Stacked LSTM model. The basic concept behind this technique is to use a back propagation algorithm to train the network after it has been unfolded. The difference between the actual and predicted outputs is computed, and the weight matrix is then modified to account for the smallest mistake. The weight gradient is finally calculated and repeatedly updated.

3. Experiments

3.1. Data Preprocessing and Environment Setting

Dataset

Research data in this study comes from the Vietnam Stock Index (VN-Index), a capitalization-weighted index of

all the companies listed on the Ho Chi Minh City Stock Exchange. The index was created with a base index value of 100 as of July 28, 2000. Prior to March 1, 2002, the market only traded on alternate days. The observed dataset contains 3042 trading days from January 2010 to August 2021, and historical data are obtained from Cafef; each sample contains daily index information, including the low price, high price, opening price, closing price, and trading volume [23]. For persistent time series, we focus on volatility modelling, so we eliminate the effects of non-trading days (or holidays), which lead to irregular gaps in daily financial data [24] and have been shown to be generally consistent [25]. Accordingly, we fill in the missing business date data with the previous figures.

Stock Returns Calculation and Transformation

We assume that prices are distributed log-normally [26]. Accordingly, the stock returns are calculated by taking the natural logarithm of the changes of daily prices. $r_t = ln\left(\frac{S_t}{S_{t-1}}\right)$, where r_t is stock return at time t; S_t is stock price at time t; S_{t-1} is the stock price at time t-1.

Logarithmic transformations are also a classical method of transforming a highly skewed variable into one that is more approximately normal in econometrics. The resulting distribution before and after transformations are plotted in Figure 4.

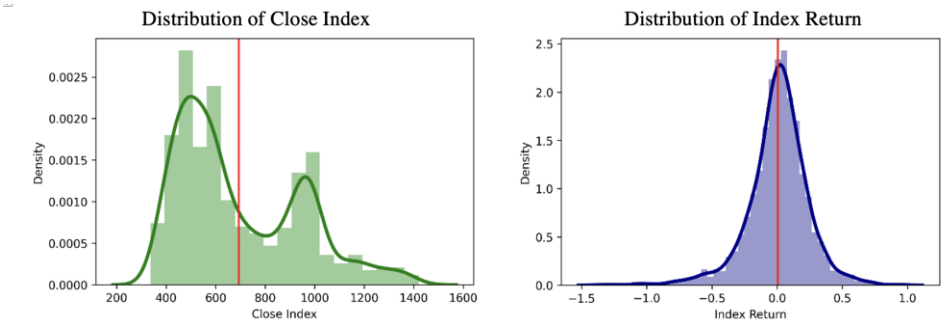

Figure 4: Distributions of Data.

Nested Cross-Validation Data Splits

A nested cross-validation consists of an inner loop for hyperparameter tuning and an outer loop for error estimation. Each training set is also divided into a training subset and a validation set. To avoid over-fitting, the model is trained on the training subset using the hyperparameters that minimize error (RMSE, MAE) on the validation set (the last 20% of the training sets). The entire dataset is divided into five pairs of training and testing sets included in the outer loop and illustrated in Figure 6, the error on each split averaged is a reliable estimate of model error. The true error is therefore estimated almost unbiasedly [27].

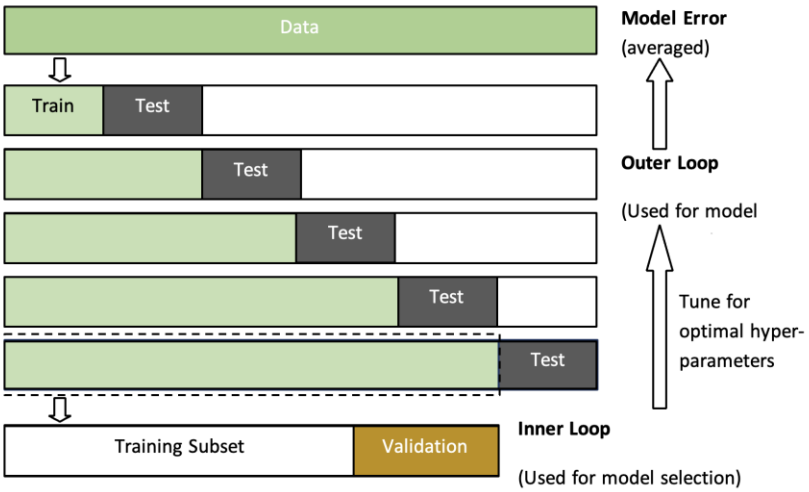

Figure 5: Nested Cross-Validation**.**

3.2. Performance Metrics

To evaluate and compare the performance of the models, this paper has selected two popular measure metrics in regression: Root Mean Square Error (RMSE) with quadratic squaring rule that measures the average magnitude of error and accordingly is sensitive to overlarge and over small values; and Mean Absolute Error (MAE), an unweighted linear score that treats the values equally, which seems not to penalize large errors and therefore underestimate the results.

The RMSE is defined as:

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

where n is the number of predicted values or the number of days ahead predicted, $\widehat{y_i}$ and y_i are predicted value and actual value at the same time step i. With the same denotations, the MAE is defined as:

$$MAE = \sqrt{\sum_{i=1}^{n} \frac{|\widehat{y_i} - y_i|}{n}}$$

3.3. Data Examination

Examining Moving Average (MA) and Density Probability Distribution

To soften the return action by filtering noise out of random short-term price movements, a MA plot is used. Within this time series dataset, the examined monthly resamples coincide with simple moving average results plotted in Figure 6. The yield series plot shows great volatility in data, with near-zero mean and constant

variance, suitable for a hypothetical white noise, which provides the basis for almost all applied statistical models, including the integrated autoregression models used to forecast index returns.

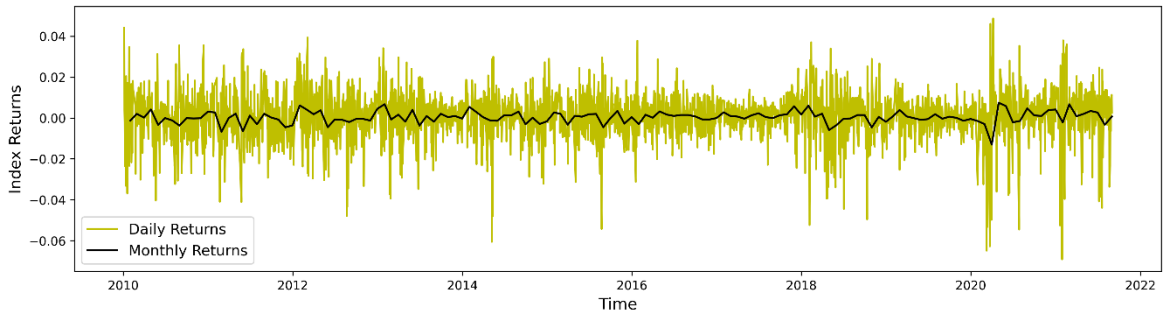

Figure 6: Simple Moving Average of Index Returns**.**

In Figure 6, the density probability distribution of the whole dataset is very close to normal, from calculating statistical parameters in table 1.

Table 1: Statistical Data Summary.

	Size	**Mean**	**Std.**	**Median**	**Skewness**	**Kurtosis**
Train 1	507	-0.000125	0.013359	0.000225	-0.128923	0.467460
Train 2	1014	0.000409	0.012794	0.000357	-0.209550	0.883043
Train 3	1521	0.000306	0.012279	0.000559	-0.375370	1.502480
Train 4	2028	0.000391	0.011277	0.000694	-0.377355	1.993994
Train 5	2535	0.000454	0.011399	0.000805	-0.442135	2.051325

It was found that the distribution has a sample mean that is close to the median and is zero. The skewness is close to 0. Excess kurtosis greater than 0 indicates a leptokurtic distribution. This distribution has heavier tails, or its outliers have a higher probability. To test the properties of the distribution more accurately, the Quantile-Quantile hierarchical histograms are shown in Figure 7, where the vertical axis represents sample quantiles, the horizontal axis indicates theoretical quantiles. These histograms are used to evaluate whether the data fits a pre-selected distribution, which shows that the two series have a similar distribution. If the points on the graph lie on a line, it can be considered as a normal distribution.

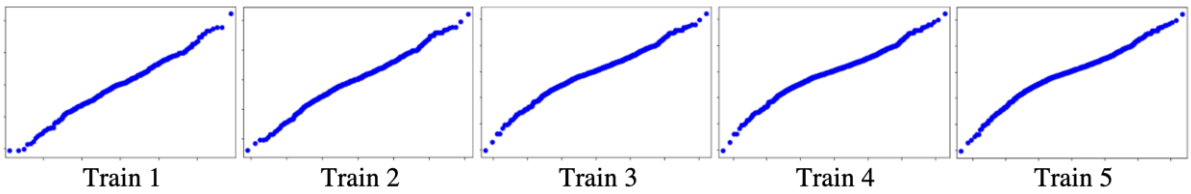

Figure 7 : Quantile-Quantile hierarchical histogram.

The distributions of index returns are almost like normal distributions because the points in the Quantile-Quantile series lie on straight lines.

Examining For Stationarity

A time series is stationary when the mean, variance, and covariance (at different lags) remain constant no matter what time the series is determined [28]. Applying the Augmented Dickey–Fuller (ADF) test on split sets we obtain results in table 2.

Table 2: Augmented Dickey-Fuller Statistics.

	Train 1	**Train 2**	**Train 3**	**Train 4**	**Train 5**
ADF Statistic	-17.824058	-10.556752	-11.334690	-13.106086	-14.628547
P-value	0.000000	0.000000	0.000000	0.000000	0.000000
Critical Value 1%	-3.443340	-3.436860	-3.434691	-3.433598	-3.432945
Critical Value 5%	-2.867269	-2.864414	-2.863457	-2.862975	-2.862686
Critical Value 10%	-2.569821	-2.568300	-2.567791	-2.567534	-2.567380

P-values are less than 0.05, so we reject the null hypothesis, the characteristic equations have no unit root; hence, the series of index returns are stationary.

Examining For Autocorrelation

Autocorrelation is a mathematical representation of the degree of similarity between a given time series and a lagged version of itself over successive time intervals. It's conceptually like the correlation between two different time series, but autocorrelation uses the same time series twice: once in its original form and once lagged one or more time periods. In Figure 8, we examine this characteristic via Autocorrelation Function (ACF) plot and Partial Autocorrelation Function (PACF) plot, where the horizontal axis is the lag, the vertical axis is the value of the autocorrelation coefficient corresponding to the lag.

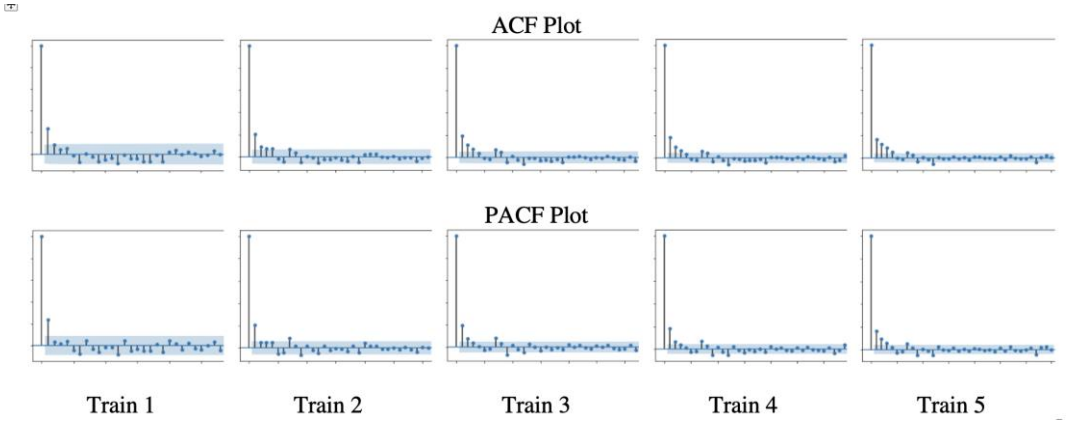

Figure 8 : Autocorrelation And Partial Autocorrelation Plots.

The blue bands are the 95% confidence intervals for the autocorrelation values to be zero. If at the smallest delay, the line segment (perpendicular to the horizontal axis) represents the value of the autocorrelation coefficient. If the correlation is outside the confidence interval, that is the largest fitting delay that we should choose for the moving average process. In general, the steps should not be too large. For this problem, the first autocorrelation coefficient has values outside the 95% confidence interval.

3.4. Model Specifications and Parameters

This study applies a manual and automatic grid search to investigate the optimal architectural factors of the LSTM network, including the number of inputs to be fed (also called the sequence length, n_inputs), the number of LSTM units composing hidden layers (n_nodes), the number of epochs (n_epochs) and the batch size (the number of training examples utilized in one iteration, denoted as n_batch). For short, the LSTM model configures can be written as: LSTM (n_inputs, n_nodes, n_epochs, n_batch). The best hyperparameters found manually are then fed to an automatic grid search, in which they take turns to combine to form new models, operating on the train sets and predict on the validation sets using a walk-forward validation approach. The cost function results of these models are compared with each other to finally determine the optimal models predicted on the test sets. Due to the volatility of artificial neural network estimated outputs, each model is operated 10 times and the results are averaged.

This study uses a stacked LSTM model, which can capture longer temporal dependencies, and thus capture even more complex patterns than vanilla LSTM (or baseline LSTM) [29]. Each LSTM layer has the same number of nodes, and the hyperbolic tangent function is utilized as an activation function, which returns values into a range between -1 and 1. A gradient-based “Adam” optimizer is used to adjust the network weight, which is noted for its simplicity, straightforwardness, and computational efficiency. The approach shows good performance with gradients that are noisy and sparse [12]. Table 4 shows the optimal hyperparameters after 480 grid search results, each of which is repeated in a ten-time loop.

Table 4: Optimal Stack LSTM Model Configurations.

Training Sets	Optimal LSTM Model For RMSE	Optimal LSTM Model For MAE
Train 1	LSTM (3, 128, 120, 256)	LSTM (4, 128, 120, 256)
Train 2	LSTM (4, 128, 120, 128)	LSTM (4, 128, 120, 128)
Train 3	LSTM (2, 128, 60, 128)	LSTM (2, 32, 120, 128)
Train 4	LSTM (4, 128, 60, 128)	LSTM (5, 128, 60, 128)
Train 5	LSTM (5, 32, 60, 256)	LSTM (5, 32, 60, 256)

3.5. Results And Discussion

As discussed earlier, this study applies the artificial neural network method Stacked LSTM, compared with the traditional statistical methods ARIMA and SARIMA to predict the return rate of the VN-Index according to stock index. Dates from 2010 to 2021. Figure 9 demonstrates the prediction of the output returns of VN-Index, comparing the actual and the predicted value of the last 507 days in the dataset, using the configuration fitted on the previous 2535 days.

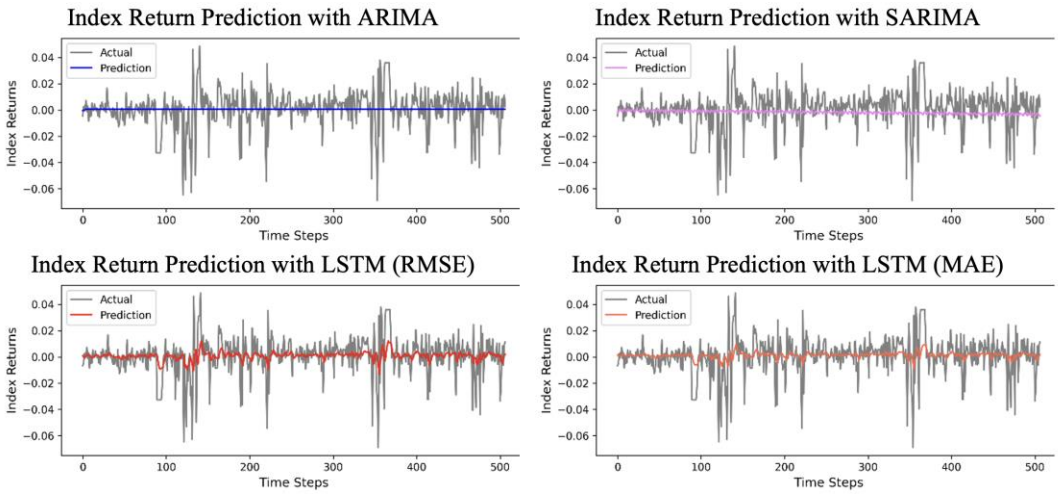

Figure 9 : Index Return Prediction.

The ARIMA model provides a linear forecast of the index returns (blue), eliminating the effects of trends and seasonality. It is likely that the linear fitted line overlaps the mean and close to zero because of the nearly normal distribution of the index returns. Although the ARIMA has been known for being statistically meaningful, these neutral predictions of the ARIMA model does not provide the analysts and investors with any stock market direction (up and down) to consider, and thus unhelpful in practice. The SARIMA model applies a fixed seasonal pattern learned from the previous time steps (violet), however, this can be technically disadvantageous in prediction compared to the ultra-stableness or the flexibility of the LSTM network. In fact, the stock market returns are highly volatile, unstable, and even random with sharp impulsive fluctuations at many points depending on complex factors of the stock market. The fixed effects of trading days on index returns are little and therefore the SARIMA model can hardly predict the erratic peaks, troughs, and longer-term outliers in the dataset.

The LSTM network prediction (red, orange) works flexibly and appears to have an optimistic performance during the fluctuant periods. When the market is highly irregularly unstable and follow a completely different pattern from the rest of the series, for example, during economic crisis in Vietnam starting in April 2020 with the plummet of stock returns, which can be seen at the third quarter area of Figure 8, the LSTM neural network can rapidly learn the trend of the outlying month and when index returns become more regular, or when the economy is improved and gradually obtains a more balance state, the LSTM also updates the new trends. It is also worth noticing that LSTM optimizing RMSE cost function (red) results in larger fluctuations at some outlying points compared to the one minimizing the MAE.

To optimize the parameters of the statistical models ARIMA and SARIMA, this study used the way of minimizing the intermediate objective function AIC by grid-search method as described in the previous section. The RMSE and MAE results are calculated based on the optimal configurations of the model and summarized in Table 10.

The LSTM neural network model is built with hyperparameters and optimized using several techniques as

described in the previous section and repeated testing many times. This model uses an optimizer that directly minimizes the RMSE and MAE objective functions. These results are also summarized in Table 10.

Any single metric provides only one projection of the model errors, and therefore only emphasizes a certain aspect of the error characteristics. In this paper, we use RMSE and MAE as objective cost functions. The underlying assumption when presenting the RMSE is that the errors are unbiased and follow a normal distribution. The RMSE is sensitive to the examples with the largest difference, because the error is squared before the average is reduced with the square root. RMSE is also sensitive to outliers, so the example with the largest error would skew the RMSE. To get the big picture of the results, in Figure 10, the RMSE of the three methods are compared.

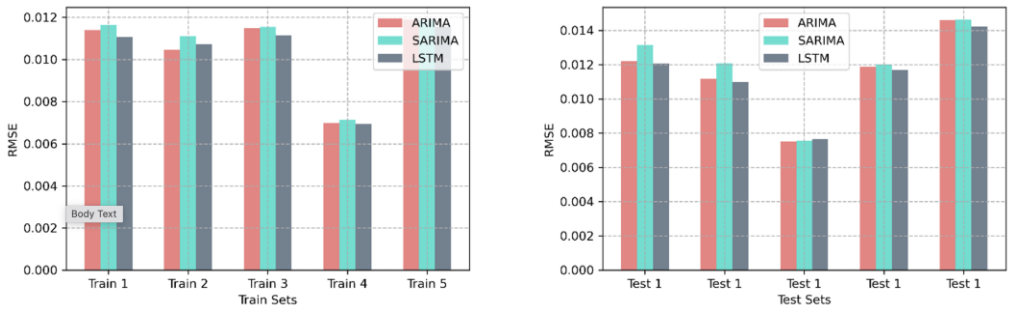

Figure 10 : RMSE Comparison of 3 Methods.

Unlike the RMSE, the MAE is suitable to describe uniformly distributed errors. MAE outperforms the RMSE in case we want the model not to be skewed by outliers. In Figure 11, we also plot the difference among 3 models.

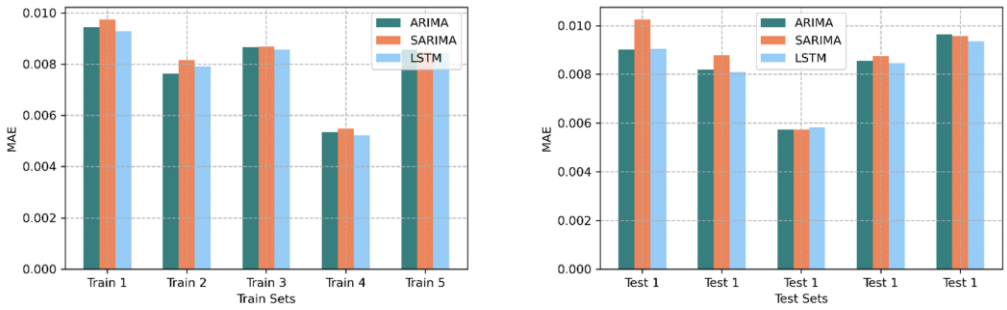

Figure 11 : MAE Comparison of 3 methods**.**

In table 5, we calculate the average RMSE and MAE of the train sets and test sets separately. The overall results show that the SARIMA method underperforms the others. The LSTM model yields the best result (the lowest RMSE and MAE) in almost all datasets provided.

The results show that the performance of Stacked LSTM is the best among the three methods. In terms of forecasting accuracy, RMSE of the train sets and test sets are 0.010285 and 0.011322 respectively, MAE of the train sets and test sets are 0.007883 and 0.008158 respectively, which are the smallest among the three forecasting methods and have high forecasting accuracy, in terms of forecasting performance, therefore, the

Stacked LSTM proposed in this paper is superior to the other three comparative models in terms of fitting degree and error value. It can well predict the returns of the stock market indices and provide a reference for analysts and investors' decisions.

Table 5: Result Comparison of Different Models.

	RMSE			**MAE**		
Models	**ARIMA**	**SARIMA**	**LSTM**	**ARIMA**	**SARIMA**	**LSTM**
Train 1	0.011400	0.011625	0.011075	0.009448	0.009735	0.009282
Test 1	0.012214	0.013165	0.012074	0.009029	0.010246	0.009040
Train 2	0.010475	0.011103	0.010712	0.007619	0.008149	0.007905
Test 2	0.011173	0.012092	0.010990	0.008188	0.008792	0.008098
Train 3	0.011494	0.011541	0.011147	0.008658	0.008674	0.008566
Test 3	0.007531	0.007564	0.007640	0.005737	0.005731	0.005821
Train 4	0.006967	0.007123	0.006931	0.005335	0.005480	0.005220
Test 4	0.011890	0.012040	0.011683	0.008566	0.008752	0.008470
Train 5	0.011888	0.011859	0.011560	0.008561	0.008476	0.008441
Test 5	0.014620	0.014633	0.014224	0.009636	0.009571	0.009358
Mean Train	**0.010445**	**0.010650**	**0.010285**	**0.007924**	**0.008103**	**0.007883**
Mean Test	**0.011485**	**0.011899**	**0.011322**	**0.008231**	**0.008619**	**0.008158**

4. Conclusions

As stock market index predictions can result in a real financial loss or gain, it is crucial to improve the predictability of models. As a result, several researchers have attempted to model and forecast financial time series utilizing statistical or soft computational abilities capable of analyzing the complex and chaotic financial market. Deep learning approaches have been widely used in recent years, owing to their good results in a variety of categorization tasks.

In this study, we construct an index return prediction model based on a recurrent neural network model LSTM, which is one of the most used deep learning techniques, as well as traditional statistical models ARIMA and SARIMA. We also utilize a nested cross-validation approach with a view to getting a fair model comparison.

To assess the efficiency of this method, we perform an experiment using 11 years of VN-Index data, predicting the log returns of closing prices, and comparing the results to a basic model. Our suggested LSTM method has the lowest RMSE and MAE, according to the empirical results. These results indicate that a stacked LSTM technique might be an effective tool for stock market index forecasting that considers temporal patterns.

The statistical models are optimized using an automatic grid search that minimizes AIC value, after that, they are fit to predict results on the new datasets. However, it is unlikely to offer a reference to the investors and economic analysts due to its neutral linear trend, in that case, a classifier or a neural network regressor would yield more valuable results. The SARIMA model is non-linear and more flexible than ARIMA, even so, since it has an unchanged pattern, the SARIMA poorly performs in the prediction of such a highly volatile and turbulent dataset. Both statistical models in this study are not practically useful to analysts and thus the LSTM network is more promising to develop since it provides a flexible and updatable learning technique. The results show that the performance of Stacked LSTM is the best among the three methods. The LSTM models' configuration of

hyperparameters are selected through a process that combines grid search and manual search. All the evaluations of models are repeated many times to avoid the volatility and improve the reliability of results before the conclusion.

This study suggests an implication for building regressive models to predict the stock market index returns. Besides, the distributions and characteristics of stock index returns are also examined, analyzed, and assessed before the regression.

Although the comparison of three models results in a prominent predictive performance of the LSTM model, it still has limitations. First, we did not take many variables of macroeconomics and financial markets, which might have a strong influence on stock indices into the regression models. In the real world, there are also qualitative factors that affect the movement of stock indices, and they might raise the accuracy of the models significantly. Secondly, this study is conducted only on the Vietnam Stock Index data in a truncated period between 2010 and 2021.

Future research can include data from various stock markets, longer time series or using a different resampling technique. Then, the LSTM network can be improved using different integrated models, hybrid models or even replaced by other updated networks to predict the stock index returns.

Acknowledgements

The authors would like to thank to Thuyloi University (TLU) and Vietnam Center of Research in Economics, Management and Environment (VCREME).

Reference

[1] Y. S. Abu-Mostafa and A. A. F, "Introduction to financial forecasting," *Applied Intelligence,* vol. 6, no. 3, pp. 205-213, 1996.

[2] F. Campanella, M. Mustilli and E. D¡ Angelo, "Efficient Market Hypothesis and Fundamental Analysis: An Empirical Test in the European Securities Market," *Review of Economics and Finance,* vol. 6, pp. 27-42, 2016.

[3] S. K. Shahid and A. Rahaman, "Exponential Smoothing Methods for Detection of the Movement of Stock Prices," *International Journal of Recent Technology and Engineering,* vol. 8, no. 5, 2020.

[4] P. K. Sahoo and K. Charlapally, "Stock Price Prediction Using Regression Analysis," *International Journal of Scientific & Engineering Research,* vol. 6, no. 3, pp. 1655-1659, 2015.

[5] A. A. Adebiyi, A. O. Adewumi and C. K. Ayo, "Comparison of ARIMA and Artificial Neural Networks Models for Stock Price Prediction," *Journal of Applied Mathematics,* pp. 1-7, 2014.

[6] D. G. McMillan, "Nonlinear predictability of stock market returns: Evidence from nonparametric and

threshold models," *International Review of Economics & Finance,* vol. 10, no. 4, pp. 353-368, 2001.

[7] V. Akgiray, "Conditional Heteroscedasticity in Time Series of Stock Returns: Evidence and Forecasts," *The Journal of Business,* vol. 62, no. 1, p. 55, 1989.

[8] R. Adhikari and R. K. Agrawal, "An Introductory Study on Time series Modeling and Forecasting," *LAP Lambert Academic Publishing,* 2013.

[9] H. Lee, R. Grosse, R. Ranganath and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning - ICML,* 2009.

[10] R. Cervelló-Royo, F. Guijarro and K. Michniuk, "Stock market trading rule based on pattern recognition and technical analysis: Forecasting the DJIA index with intraday data," *Expert Systems with Applications,* vol. 42, no. 14, p. 5963–5975, 2015.

[11] D. Enke and N. Mehdiyev, "Stock Market Prediction Using a Combination of Stepwise Regression Analysis, Differential Evolution-based Fuzzy Clustering, and a Fuzzy Inference Neural Network," *Intelligent Automation & Soft Computing,* vol. 19, no. 4, pp. 636-648, 2013.

[12] H. Chung and K. Shin, "Genetic Algorithm-Optimized Long Short-Term Memory Network for Stock Market Prediction," *Sustainability,* vol. 10, no. 10, p. 3765, 2018.

[13] E. Chong, C. Han and F. C. Park, "Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies," *Expert Systems with Applications,* vol. 83, pp. 187-205, 2017.

[14] B. Lei, B. Zhang and Y. Song, "Volatility Forecasting for High-Frequency Financial Data Based on Web Search Index and Deep Learning Model," *Mathematics,* vol. 9, no. 4, p. 320, 2021.

[15] R. K. Nayak, D. Mishra and A. K. Rath, "A Naïve SVM-KNN based stock market trend reversal analysis for Indian benchmark indices," *Applied Soft Computing,* vol. 35, pp. 670-680, 2015.

[16] H. Yu, R. Chen and G. Zhang, "A SVM Stock Selection Model within PCA," *Procedia Computer Science,* vol. 31, pp. 406-412, 2014.

[17] F. A. De Oliveira, C. N. Nobre and L. E. Zárate, "Applying Artificial Neural Networks to prediction of stock price and improvement of the directional prediction index – Case study of PETR4, Petrobras, Brazil.," *Expert Systems with Applications,* vol. 40, no. 18, p. 7596–7606, 2013.

[18] A. J. Hussain, A. Knowles, P. J. G. Lisboa and W. El-Deredy, "Financial time series prediction using polynomial pipelined neural networks," *Expert Systems with Applications,* vol. 35, no. 3, pp. 1186-1199, 2008.

[19] D. Caplinger, "What Is a Stock Market Index? The Motley Fool.," 2020. [Online]. Available: www.fool.com/investing/stock-market/indexes . [Accessed 29 September 2021].

[20] P. Gao, R. Zhang and X. Yang, "The Application of Stock Index Price Prediction with Neural Network," *Mathematical and Computational Applications,* vol. 25, no. 3, p. 53, 2020.

[21] F. A. Gers, "Learning to forget: continual prediction with LSTM," in *9th International Conference on*

Artificial Neural Networks: ICANN '99, 1999.

[22] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997.

[23] "cafef," [Online]. Available: s.cafef.vn/du-lieu/download.chn. [Accessed 6 September 2021].

[24] Š. &. M. P. Lyócsa, "The effect of non-trading days on volatility forecasts in equity markets," *Finance Research Letters*, vol. 23, pp. 39-49, 2017.

[25] M. Asai and M. McAleer, "Non-trading day effects in asymmetric conditional and stochastic volatility models," *The Econometrics Journal*, vol. 10, no. 1, pp. 113-123, 2007.

[26] J. Wooldridge, Introductory econometrics: A modern approach (5thed.), Mason, OH: South-Western, Cengage Learning, 2013.

[27] S. Varma and R. Simon, "Bias in error estimation when using cross-validation for model selection," *BMC Bioinformatics*, vol. 7, no. 1, 2006.

[28] D. Gujarati, Econometrics, 2015.

[29] L. Chen and M. Xu, "Piecewise Time Series Prediction Based on Stacked Long Short-Term Memory and Genetic Algorithm," *2020 Chinese Automation Congress (CAC)*, 2020.