Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Stock Price Prediction through STL Decomposition using Multivariate Two-way Long Short-term Memory

Junsuke Senoguchi
Tokyo University of Technology, School of Computer Science, Department of Computer Science, Japan
Corresponding Author: Junsuke Senoguchi, **E-mail**: senoguchijs@stf.teu.ac.jp

| ABSTRACT
With advancements in machine-learning techniques, stock-price movements can ostensibly be forecasted using time-series data. In this study, several different types of long short-term memory (LSTM) are used to predict the closing prices of Japanese stocks five days into the future. Also, in this study, four different features [i.e., simple moving average (SMA), linear weighted moving average (WMA), exponential WMA (EMA), and the Savitzky–Golay (SG) metric] are generated from daily stock-price data and split into two components (i.e., trend and seasonal) by applying seasonal–trend decomposition using Loess (STL) decomposition. The prediction results are evaluated in terms of return, root-mean-square error (RMSE), mean absolute error (MAE), and other relevant measures of accuracy and relevancy. As a result, the multivariate two-way LSTM model yielded the highest overall performance. With respect to the RMSE and MAE of the training data, the multivariate two-way LSTM was not superior to the other models. However, with respect to RMSE and MAE on the validation data, it was the best. Also, the multivariate two-way LSTM model yielded the highest overall performance in terms of the accuracy of the direction of stock prices.

| KEYWORDS

Machine learning; multivariate bidirectional LSTM; STL decomposition; stock-price prediction

| ARTICLE INFORMATION

ACCEPTED: 08 October 2022 **PUBLISHED:** 08 October 2022 **DOI:** 10.32996/jcsts.2022.4.2.11

1. Introduction

A stock price reflects the contract value associated with trading a share of a company's net worth at the moment in time, as reflected in one or more exchanges. Notably, stock prices change over time, and because all available information about a company's value is presumably factored into the price, predicting these changes promptly is extremely difficult. Traditional stock-market predictions are made using statistical models. However, with advancements in machine-learning techniques, stock-price movements can ostensibly be forecasted using time-series data. Notably, several analytic comparisons have been made between traditional statistical methods and machine learning models. Furthermore, combined models have been proposed, as well as Bayesian optimization versions.

In this study, daily stock-price data comprising high, low, opening, and closing values were used to predict future closing prices five days into the future. Three types of multivariate bidirectional long short-term memory (LSTM) were applied as candidate methods (i.e., standard, two-way, and multivariate). These were evaluated in terms of return, root-mean-square error (RMSE), mean absolute error (MAE), and other relevant measures of accuracy and relevancy.

Four additional features [i.e., simple moving average (SMA), linear weighted moving average (WMA), exponential WMA (EMA), and the Savitzky–Golay (SG) metric] were generated from daily stock-price data and split into two components (i.e., trend and seasonal) by applying seasonal–trend decomposition using Loess (STL) decomposition. A multivariate two-way LSTM model was constructed to forecast each component. Standard and multivariate LSTMs were used as comparison targets. The training and forecasting

results were evaluated using three indices: accuracy of stock price direction, RMSE, and MAE.

2. Background

Many prior studies have applied machine-learning techniques to stock-price prediction tasks (Nelson et al., 2017, Chen et al., 2015). Miyazaki and Matsuo 2017 proposed a model that predicted Nikkei 225 price changes using 30-min data increments and compared the results to those of LSTM, logistical regression, random forest (RF), convolutional neural network (CNN), multilayer perceptron (MLP), and other methods. The training data included 30 historical Nikkei and Topix Core component stocks. The classification problem was designed to predict two values corresponding to the rise and fall of the overall Nikkei Index. The temporal data interval was set to 30 min using 100 steps each to forecast stock prices 30 min into the future. Exchange opening data (9:00 a.m.) were discarded owing to the large shifts that occurred after the starting bell. However, data from 9:30 a.m. to 12:30 p.m. were divided by the square root of the period to account for shifts caused by the period durations. The overall forecasting range encompassed six months, from January to June 2016, with each month's data used for testing. The training period included 12 months of historical data. The nine models used for their evaluation included the following:

- Rand-1 (predicted fluctuations in test data by the rate of return of the Nikkei Index over one year's training data)
- Rand-2: (predicted rise and fall with a 50% probability)
- Logistic regression
- RF
- MLP (dense-4 layers)
- LSTM (two layers + dense-2 layers)
- CNN [conv + maxpooling (two layers) + dense-2 layers]
- CNN with principal component analysis preprocessing [conv + max-pooling (two layers)]
- CNN-LSTM: [Conv1D + Maxpooling1D (one layer) + LSTM (two layers) + Dense-1 layer]

The optimal model was determined to be the LSTM, and the rate of correct responses to stock fluctuations was used as an evaluation metric. A summary of CNN's results is presented in Figure 1.

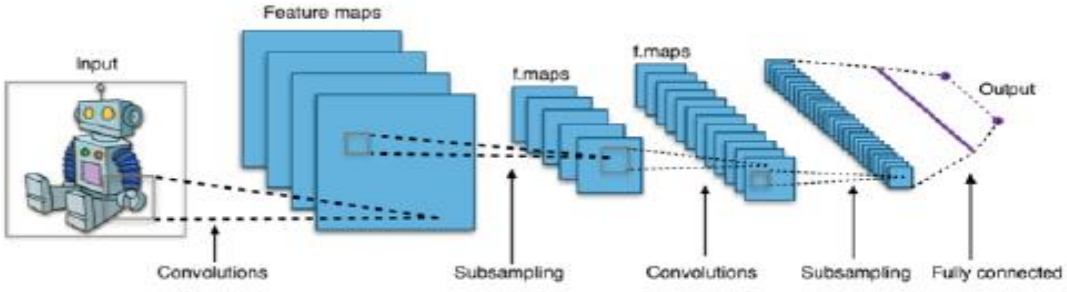

Figure 1. Outline of a convolutional neural network [1].

Several past forecasting studies have been conducted using daily data. Komatsu, 2018 evaluated stock-price prediction using an LSTM under various simulated training periods, forecast thresholds, training data, data-series length, and numbers of LSTM units. The results were evaluated and compared using MAE. Specifically, in a neural-network model consisting of input, intermediate, and output layers, Komatsu provided one that used an LSTM block in the intermediate layer, where s = {52,102,152,202}. The Nikkei 225 stock price was used as the forecasting target from March 13, 2000, to March 19, 2018.

Instead of forecasting the stock price itself directly, Angelo et al., 2017 decomposed the S&P 500 Index over a five-year period into trend and seasonal factors. After using STL decomposition to split the stock-price data into trend and seasonal components, models were constructed with several combinations of CNN and LSTM. Figure 5 provides an overview of the model, so called "MegazordNet," in this paper and the decomposed time-series data.

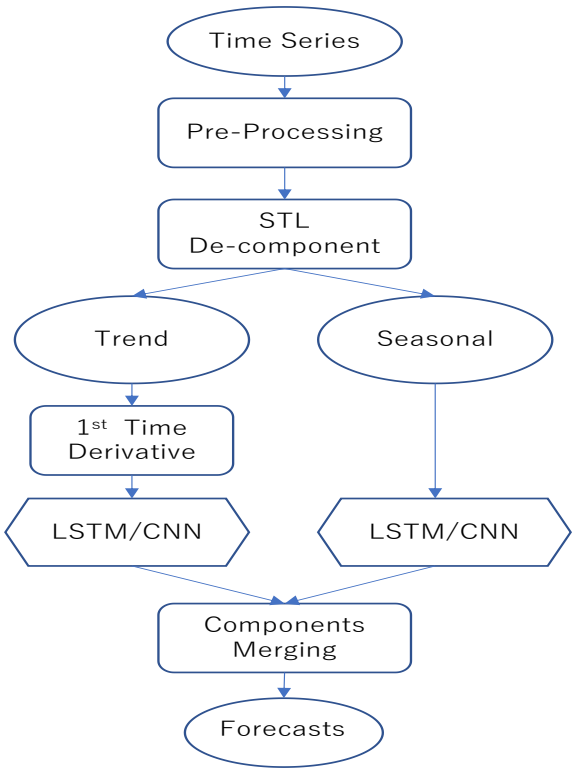

Figure 5. Overview of MegazordNet (Angelo et al., 2017)

As mentioned above, forecasts based on 30-min data have shown excellent forecasting accuracy. However, because the accuracy of past studies was verified over relatively limited periods, it is questionable whether the same results can be obtained when the market environment changes. When forecasts use models with relatively short back-testing periods, forward testing frequently does not produce the desired results. Therefore, it is necessary to construct a model with longer training and back-testing periods to ensure that accuracy remains stable, even during different financial market phases. Prior studies by Takayuki (2018) and Angelo et al. (2017) conducted multi-year back tests using daily data, resulting in excellent accuracy. Other studies (Ding et al., 2015, Jia, 2016, Persio & Honchar, 2016, Akita et al., 2016) used time-series data analyses with LSTM and STL decomposition, for which it was more appropriate to forecast stock prices a few business days ahead, rather than the next business day.

3. Methodology

As mentioned in the previous section, the purpose of this study was to predict future stock prices for five days ahead. Furthermore, as discussed in the introduction, to avoid the influence of short-term fluctuations with respect to time-series data trends, SMA, WMA, EMA, and SG metrics were created from daily closing data as explanatory variables. These features, alongside closing prices, were decomposed into trends and seasonal components using STL decomposition. The forecast target was also decomposed into trend and seasonal components, with forecasts made for each using the multivariate interactive LSTM. Standard and multivariate LSTMs were used for comparison targets, and stock price direction accuracy, RMSE, and MAE were used as evaluation indices.

3.1 Feature Engineering

We considered the following four features in this study:

- *SMA*: the average value from the present to the past n periods:

  $$SMA_n = \frac{\sum_{i=t-n}^{t} y_i}{n}, \quad (1)$$

  where n is the number of periods, SMA_n is the simple moving average of past periods n, t is the current time, and y_i is the stock price in period i.
- *WMA*: an average value that linearly decreases in weight as it moves away from the current time toward the past n periods, including the value at the current time:

  $$WMA_n = \frac{\sum_{i=1}^{n}\{(n-i+1)y_{t-i+1}\}}{\sum_{i=1}^{n} n-i+1}, \quad (2)$$

  where n is the number of periods, t is the current time, y_i is the stock price in period i, and WMA_n is the WMA over the past period n.
- EMA: a WMA value that exponentially decreases in weight as it moves away from the current time toward the past n periods, including the value at the current time:

$$WMA_n = \frac{\sum_{i=1}^{n}\{(n-i+1)y_{t-i+1}\}}{\sum_{i=1}^{n} n-i+1}, \quad (3)$$

where n is the number of periods, t is the current time, y_i is the stock price in period i,α is the smoothing coefficient, and EMA_n is the EMA average of the past period, n.

- SG: data of the past n periods, including the current value, are approximated using a polynomial equation. When differentiation is performed, the derivative coefficient of the polynomial equation is used as the value afterward.

3.2 STL decomposition

Next, we describe the STL decomposition used to split each feature into trend and seasonal components. Time-series data can generally be expressed using the additive model of Equation (4):

$$y_t = S_t + T_t + R_t, \quad (4)$$

where y_t represents the data at time t, S_t is the seasonal component at time t, T_t is the trend component at time t, and R_t is the residual component at time t.

Conventional methods used to decompose time-series data into additive models include the *X*11, which applies symmetric moving averages to estimate trend, seasonal, and irregular components, and the seasonal extraction method, which uses the autoregressive integrated moving average (SEATS). The STL decomposition method has three major advantages. First, it can be applied to any type of seasonal data, whereas SEATS can decompose only monthly or quarterly data. Second, seasonality can vary over time, and its rate of change can be specified. Third, STL is robust against outliers, which allows for more accurate estimations.

3.3 Machine learning models used in this study

The LSTM replaces the middle-layer units of a recurrent neural network (RNN) with LSTM blocks so that it may learn long-term dependencies, which is otherwise impossible for an RNN. This advantage allows Hochreiter's gradient vanishing problem – explained next – to be solved. The two most common methods for training RNNs are back-propagation-through-time and real-time recurrent-learning methods, which employ algorithms that propagate the gradient in opposite directions, causing it to diverge or disappear during learning. As shown by Hochreiter, 2001, this causes the long-term dependence series to not be learned. An overview of the vanishing gradient problem is provided below.

Consider the propagation of errors from units u to v. If the error generated in u at step t is propagated q steps ahead of v, then the propagating error is scaled by the following coefficients:

$$\frac{\partial v_v(t-q)}{\partial v_u(t)} = \begin{cases} f_v'\big(net_v(t-1)\big)w_{uv} & q = 1 \\ f_v'\big(net_v(t-q)\big)\sum_{l=1}^{n}\frac{\partial v_v(t-q+1)}{\partial v_u(t)} & q > 1' \end{cases} \quad (5)$$

where $l_q = v$ and $l_0 = u$. We then obtain Equation. (6):

$$\frac{\partial v_v(t-q)}{\partial v_u(t)} = \sum_{l_1=1}^{n}\cdots\sum_{l_{q-1}=1}^{n}\prod_{m=1}^{q} f_{l_m}'\left(net_{l_m}(t-m)\right)w_{l_m l_{m-1}} \quad (6)$$

Next, when $\left|f_{l_m}'\left(net_{l_m}(t-m)\right)w_{l_m l_{m-1}}\right| > 1.0$ for all m, the scale factor diverges. Furthermore, when $\left|f_{l_m}'\left(net_{l_m}(t-m)\right)w_{l_m l_{m-1}}\right| < 1.0$ for all m, the scale factor vanishes. Extending this notion with respect to all units generates the gradient vanishing problem.

Here, we describe the basic LSTM structure. The first step is to decide whether to discard the information from a cell. This decision is made using a sigmoidal "forgetting" layer. Specifically, a numeric value, f_t in [0,1], is the output reflecting information in each cell, C_{t-1}, based on the output value in the previous step, h_{t-1}, and the value to be input, x_t:

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big), \quad (7)$$

where f_t is the value output by the forgetting layer, σ is the sigmoid function, W_f indicates the weights of f, and b_f is the intercept of f. Values closer to zero results in more information being removed, and vice versa.

Next, the LSTM determines the new information to be stored in the cell. This step employs two layers: a sigmoid input layer that determines the values to be updated and a tanh layer that creates new information to be added to the cell in the form of a vector.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \quad (8)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \quad (9)$$

where i_t is the input layer, and $\tilde{C}_t$ represents the candidate values of the cell state.

Next, the cell state is updated accordingly, and the output value, h_t, is determined. The first step determines the part of the cell to be output by the sigmoidal layer:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t, \quad (10)$$

Next, tanh is used to compress the output value from −1 to 1 and pass it to the sigmoid gate:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_0), \quad (11)$$

where$h_t = o_t * tanh(C_t)$, and o_t is the output layer.

Conventional LSTM learns to predict subsequent data chronologically in a time series using a forward LSTM learner. However, the bidirectional LSTM employs an additional backward LSTM to learn using past and future data. This study extended the conventional LSTM model to a multivariate one that accommodated multiple explanatory variable inputs and further extended it to a multivariate two-way LSTM that learned from both directions to forecast stock-price trends and seasonal components.

4. Experiment

The target stock data chosen for this study was Nikkei 225 because it was the most suitable representative of the Japanese stock market. The data period was from January 4, 2001, to December 3, 2021. In our case, the first 60% of the data were allocated chronologically to the training set, and the remaining 40% was used for validation.

As shown in Figure 6, a multivariate two-way LSTM model was constructed for each component after decomposing stock prices into trend and seasonal components. The standard and multivariate LSTM models were then compared using a package implemented with TensorFlow. It had a three-layer, all-coupled structure with the squared error as the loss function using the Adam optimizer function. Learning was conducted using 100 epochs in each model.

Input	Input	[(None, 5, 5)]
	Output	[(None, 100, 100)]

Hidden Layer	Input	[(None, 10, 5)]
	Output	[(None, 100, 10)]

Dense	Input	[(None, 100, 10)]
	Output	[(None, 1)]

Figure 6. Architecture of the proposed model.

5. Results and Discussion

The training and prediction accuracies of all three models were evaluated using the following seven indices:

- *acc*: percentage of actual time-series data in the forecasting period that are positive or negative:

  $$acc = \frac{1}{n}\sum_{i=1}^{n} f(y_i, \hat{y}_i), \quad (12)$$

  where $f(y_i, \hat{y}_i) = \begin{cases} 0 & (when\ y_i \times \hat{y}_i < 0) \\ 1 & (when\ y_i \times \hat{y}_i > 0) \end{cases}$, n is the number of forecast periods, y_i is the actual data for the forecast period, and $\hat{y}_i$ is the value predicted by the model.
- RMSE: square root of the mean of squared errors between actual and predicted time-series data. This is easy to interpret as the units are consistent with the original data:

  $$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \quad (15)$$
- MAE: average absolute error value between actual and predicted time-series data:

  $$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \bar{y}_i|, \quad (16)$$

Tables 3 present the evaluations of all three LSTM models using the three evaluation indices defined above by using test data.

Table 3. Evaluation results for test data of each model.

	acc	trend RMSE	trend MAE	seasonal RMSE	seasonal MAE	trend RMSE	trend MAE	seasonal RMSE	seasonal MAE
LSTM (close)	59.74%	0.3557	0.1356	0.6619	0.4851	2.6704	1.9512	1.3350	0.8420
LSTM (SMA)	56.62%	0.6945	0.4005	0.6716	0.4759	1.3414	1.0118	1.1559	0.8146
LSTM (WMA)	51.06%	0.6614	0.3552	0.6984	0.4940	1.4113	1.0721	1.2076	0.8296
LSTM (EMA)	61.48%	0.6726	0.3890	0.6589	0.4743	2.5869	1.7755	1.2465	0.8285
LSTM (SG method)	64.55%	0.8987	0.5626	0.7397	0.5017	1.6219	1.0084	1.1359	0.7864
Multivariate LSTM	60.78%	0.6961	0.5434	0.7017	0.5521	1.1885	0.8990	1.1340	0.8593
Multivariate bidirectional LSTM	66.25%	0.5724	0.3635	0.8929	0.5979	1.2379	0.9022	1.0247	0.7267

Note: acc = Accuracy; EMA = Exponential weighted moving average; LSTM = Long short-term memory; SG = The Savitzky–Golay metric; SMA = Simple moving average; WMA = Linear weighted moving average.

6. Conclusions and Future Directions

In this study, stock prices were separated into trend and seasonal components using STL decomposition, and a multivariate two-way LSTM model was constructed using each component to make forecasts. We compared this model to standard and multivariate LSTM models for each component and evaluated the results using seven indices.

According to the stock direction accuracy measure, the multivariate two-way LSTM model yielded the highest overall performance. With respect to the RMSE and MAE of the training data, the multivariate two-way LSTM was not superior to the other models. However, with respect to RMSE and MAE on the validation data, it was the best. Therefore, it can be concluded that the multivariate two-way LSTM generates the most accurate predictions for new data and offers the best generalization performance.

However, three issues must be addressed in future studies. First, the model must be validated through back-testing. In this study, training and forecasting were conducted using chronologically-ordered data, with training data preceding the validation data. However, it is difficult to ascertain the practicality of the model using only this verification method. Based on this constraint, we conducted three back tests: walk-forward, cross-validation, and parsing cross-validation.

The walk-forward back test is a general time-series validation method. The model is trained using data prior to the time of forecasting, and it generates new forecasts over time. The cross-validation back-testing method uses future data to verify past data, regardless of the time series. This method allows validation in all periods, making it comparable to the walk-forward back test. Finally, the combination parsing cross-validation method allows for the creation of various validation periods by generating different combinations. The model can be evaluated more accurately by applying validation methods to the model proposed in this study.

The second issue that must be addressed is feature engineering. In this study, four feature sets were created: SMA, WMA, EMA, and SG. However, many other indicators exist for technical analysis, and incorporating them may improve model accuracy.

The third issue is the same inherited by other models. Several recent studies have applied machine- and deep-learning methods to analyze financial data. In this study, we used a model that extended those. However, many attempts like these have been applied to time-series data. Therefore, as a future challenge, more accurate results may be obtained by applying a model using transformer technology. This idea has been explored in the fields of natural language processing, music generation, and object detection.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations or those of the publisher, the editors, and the reviewers.

References

[1] Akita, R., Yoshihara, A., Matsubara, T., & Uehara, K. (2016). Deep learning for stock prediction using numerical and textual information. *In Proceedings of ACIS 15th International Conference on Computer and Information Science, Okayama,* 1-6. https://doi.org/10.1109/ICIS.2016.7550882

[2] Chen, K., Zhou, Y., & Dai, F. (2015). A LSTM-based method for stock returns prediction: A case study of China stock market. *In Proceedings of 2015 IEEE International Conference on Big Data,* 2823-2824. https://doi.org/10.1109/BigData.2015.7364089

[3] Ding, X., Zhang, Y., Liu, T., & Duan, J. (2015). Deep learning for event-driven stock prediction. *In Proceedings of 24th International Joint Conference on Artificial Intelligence*. https://dl.acm.org/doi/10.5555/2832415.2832572

[4] Hochreiter, S., Bengio, Y., Frasconi, P., & Schmidhuber, J. (2001). Gradient flow in recurrent nets: The difficulty of learning long-term dependencies. *Faculty of Computer Science Technical University of Munich 80290. München*. DOI: 10.1109/9780470544037.ch14

[5] Jia, H. (2016). Investigation into the effectiveness of long short-term memory networks for stock price prediction. https://arxiv.org/abs/1603.07893

[6] Kunihiro, M., & Yutaka, M. (2017). Stock prediction analysis using deep learning technique. *In Proceedings of the 31st Annual Conference of the Japanese Society for Artificial Intelligence, Nagoya City, Aichi, Japan*.

[7] Menezes, A. G., & Mastelini, S. M. (2017). MegazordNet: Combining statistical and machine learning standpoints for time-series forecasting. *arXiv:2017.01017v1[q-fin.ST].* https://doi.org/10.48550/arXiv.2107.01017

[8] Nelson, D. M. Q., Pereira, A. C. M., & Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. *In Proceedings of 2017 International Joint Conference on Neural Networks*, 1419-1426. https://doi.org/10.1109/IJCNN.2017.7966019

[9] Persio, L.D., & Honchar, O. (2016). Artificial neural networks architectures for stock price prediction: Comparisons and applications. *Int. J. Circuit. Sys. Signal Proc*, 10, 403-413.

[10] Takayuki, K. (2018). Experimental study on stock price prediction using deep learning. *Hokkaido Univ. Sci*. 46.