Proceedings of the 3rd International Conference on Financial Technology and Business Analysis
DOI: 10.54254/2754-1169/2024.25234

Predicting Stock Price Volatility using a CNN-BiLSTM-AM Model

Qiyu Liu[1*], Haoyu Wang[2], Yunbo Wang[3], Zihe Fang[4]

[1]*Faculty of Statistics and Mathematics, Shandong University of Finance and Economics, Jinan, China*
[2]*Faculty of Science and Engineering, University of Nottingham, Nottingham, UK*
[3]*College of Arts and Sciences, New York University, New York, USA*
[4]*College of Arts and Sciences, The Ohio State University, Columbus, USA*
**Corresponding Author. Email: qdliuqiyu@126.com*

Abstract: This study presents a novel approach to forecasting stock price volatility by combining Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) networks with a self-attention mechanism to enhance model accuracy. The CNN-BiLSTM model leverages CNN's spatial feature extraction capabilities alongside BiLSTM's bidirectional memory for analyzing temporal dependencies. To further refine model performance, Optuna was employed for hyperparameter tuning. The study utilizes daily stock data from NVIDIA, spanning 2,520 trading days from May 2014 to May 2024, with key features including first-order differenced prices, change, turnover, and volume. The data is split into 75% for training, 10% for validation, and 15% for testing. The model processes features extracted from the past 10 days to predict future volatility. Experimental results demonstrate that the CNN-BiLSTM-AM model significantly improves forecasting accuracy, providing valuable insights into financial time series analysis, investment strategies, and risk management.

Keywords: Stock price volatility, Convolutional Neural Networks, Bidirectional LSTM, Time series analysis, Machine learning

1. Introduction

Deep learning models have become integrated into financial markets due to their ability to capture the complex, non-linear relationships that are likely to exist in financial data. Among the many applications, it can be used for predicting stock price volatility, which is an indispensable part of risk management, portfolio optimization, and algorithmic trading strategies. Accurate prediction of volatility does not only help in the reduction of financial risk but also enhances decisions by investors and financial institutions.

The financial market is dynamic and volatile, making accurate prediction of future movements challenging. The Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model, a traditional statistical method, is widely used for forecasting volatility. Similarly, linear models such as Autoregressive Integrated Moving Average (ARIMA) are known for their high accuracy in

DOI: 10.54254/2754-1169/2024.25234

predicting stock market trends. [1] Nevertheless, these models often fall short in capturing the complex, non-linear dynamics inherent in financial time series. This shortcoming has prompted researchers to investigate advanced machine learning methods, which are adept at detecting non-linear patterns across diverse data types, including images, text, and speech. These methods offer promising opportunities for enhancing investment strategies in financial markets. [2]

CNNs, LSTMs, and AM are very strong with time-series data. They thus work much better than traditional models where the past events do matter for forecasting future ones. This has thus far been used in prediction tasks like stock market index prediction.

In this paper, an innovative combined model, called the CNN-BiLSTM-AM model, is introduced. CNN structures are proposed for better feature extraction, BiLSTMs to capture temporal dependencies in the data, and finally AM to place more emphasis on the most important features in the data. Integration of the above elements shall ensure higher levels of accuracy and robustness in stock volatility prediction.

1.1. CNN-BiLSTM-AM model

The proposed CNN-BiLSTM-AM model consolidates CNN, BiLSTM, and the Attention Mechanism to a strong predictive framework with high accuracy. The first block extracts features from raw input data to capture local patterns and dependencies. Subsequently, these features are channeled into the BiLSTM network to model forward and backward temporal dependencies. Finally, the AM is applied over the output of the BiLSTM network to boost the most influential features and enhance the prediction power of the model.

1.2. Contributions

This study makes several significant contributions to the field of financial time series prediction:

1. Novel Hybrid Model: It introduces a new model, CNN-BiLSTM-AM, with the ability to leverage all its components to increase predictive accuracy.
2. Comprehensive Feature Extraction: A good feature extraction utilizing CNN to capture low- and high-level data patterns.
3. Enhanced Temporal Modeling: We implemented bidirectional BiLSTM networks to incorporate the bidirectional temporal dependencies of features in capturing more underwhelming elements, which help to understand the dynamics of the market.
4. Selective Attention: Integration of the Attention Mechanism to center on the most relevant features, enhancing the interpretability and performance of the model.
5. Empirical Validation: We provide extensive empirical validation of the proposed model with real-world financial data, showing its superiority to traditional models as well as other deep learning architectures.

2. Related work

In recent years, numerous approachs for forecasting inventory have emerged. These methods encompass predicting future stock prices, forecasting stock price movements, managing portfolios, assessing risks, and developing trading strategies. And some paper specifically addresses inventory forecasting methods utilizing techniques powered by machine learning. This is also a bold attempt of machine learning for prediction. [3]

DOI: 10.54254/2754-1169/2024.25234

The financial time series prediction is considered one of the most researched fields that improve the accuracy and reliability of predictive models. This section presents a strong review of recent literature, outlining the progress from previous statistical models towards the new advanced deep learning architectures. The concern has always been on predicting stock volatility; some additional major emphasis has been laid on those that incorporate CNNs, LSTM networks, as well as the self-attention mechanism.

2.1. Convolutional Neural Networks (CNNs)

CNNs are employed to automatically find the data features such as image and speech recognition [4]. A CNN is comprised of convolutional layers, pooling layers, and fully connected layers. Key features are identified by the convolutional layers, while the size of these feature maps is diminished by pooling layers. Nonlinearity is introduced into the model through the Rectified Linear Unit (ReLU). To prevent overfitting, dropout layers are utilized, randomly setting the output of certain neurons to zero during training. [5][6][7]

CNNs have also been applied to time series forecasting. In these applications, time-series data is transformed into two-dimensional images, which are then used as inputs for the CNN to recognize patterns. Multivariate time series data is treated as space-time images, allowing the CNN to analyze the data effectively [8][9][10].

2.2. Long Short-Term Memory (LSTM)

As a type of Recurrent Neural Network, LSTM networks are created to solve the disappearing gradient problem, making them capable of learning long-term dependencies in sequential data [11]. LSTMs are widely used for predicting financial time series because they can effectively model time-based relationships. The bidirectional version, BiLSTM, enhances this capability by processing data both forwards and backwards, capturing both past and future information simultaneously. This bidirectional processing is particularly useful in financial markets, where future trends can influence current market conditions.

2.3. Attention Mechanism (AM)

The inspiration for attention mechanism comes from the working principle of human vision [12]. Our eyes can quickly scan an area and center on important parts, ignoring unnecessary details. This allows us to efficiently find valuable information with limited resources [12]. Initially developed for language translation, the Attention Mechanism has proven useful in many other applications by helping models center on the most important parts of the input data. In financial time series prediction, the AM can give different weights to various features, highlighting the key parts of the data. This selective focus improves the model's interpretability and performance by emphasizing important features and reducing the impact of noise and irrelevant information [13].

3. Background

3.1. Dataset

The dataset employed consists of the daily closing prices of NVIDIA stock over a span of ten years, from May 2014 to May 2024, totaling 2520 trading days. This dataset was partitioned into three

DOI: 10.54254/2754-1169/2024.25234

segments: the training set consists of the 75% data, and 10% of the data for the validation set, and 15% for the testing set.

The logarithmic return (r_t) is calculated for each element in the closing price series p_t where t denotes the daily closing price observation.

$$r_t = \log\left(\frac{p_t}{p_{t-1}}\right)$$

The historical volatility is computed by taking the standard deviation of the logarithmic returns over a rolling window of 30 trading days, and scaling the standard deviation by the square root of 252 (the annual count of trading days):

$$HV = \sqrt{252} \cdot \sqrt{\frac{1}{30-1}\sum_{i=1}^{30}\left(r_i - \overline{r}\right)^2}$$

where r_i represents the logarithmic returns and $\overline{r}$ represents the average logarithmic return rate.

3.2. Convolutional Neural Network (CNN)

CNNs are sophisticated multilayer neural networks designed for deep supervised learning [14]. CNNs excel in processing both time series and image data, particularly two-dimensional images, though the same principles apply to one-dimensional data. Their main advantage is the ability to efficiently capture key features from input data using a small number of parameters [15]. These extracted features are then integrated to form more intricate representations, which are subsequently passed to a fully connected layer for classification tasks.

A standard CNN architecture includes various fundamental layers: an input layer, convolution layers, pooling layers, fully connected layers, and an output layer. In the convolutional layer, convolutions are applied to the input data using kernels, generating outputs for the subsequent layer. The pooling layer is essential for minimizing parameters and computational load, while preserving key information from the feature maps. By repeatedly performing convolution and pooling, CNNs efficiently extract data features. The convolutional filters can be adjusted with various window sizes and stride lengths to suit the input data dimensions and desired features.

In a one-dimensional CNN, a one-dimensional array acts as the convolutional kernel. Unlike conventional two-dimensional convolution operations, the dimensions are adjusted to align with the one-dimensional convolution kernel, ensuring proper feature extraction. In the process of forward propagation, the output generated by the current convolutional layer l is determined using the following method:

$$x_j^{(l)} = f\left(\sum_{i=1}^{N_{(l-1)}} k_{i,j}^{(l)} * x_i^{(l-1)} + b_j^{(l)}\right)$$

where $x_j^{(l)}$ is the output of the j -th feature map in layer l , $N_{(l-1)}$ is the number of feature maps in the previous layer $l-1$, $k_{i,j}^{(l)}$ represents the convolutional filter connecting the i -th

DOI: 10.54254/2754-1169/2024.25234

feature map of layer $l-1$ to the j -th feature map of layer l, $x_i^{(l-1)}$ is the i -th feature map in layer $l-1$, $*$ denotes the convolution operation, $b_j^{(l)}$ is the bias term for the j -th feature map in layer l and the activation function, f, was defined as:

$$ReLu(x) = f(x) = \max(0, x)$$

The pooling layer, often employing max-pooling, subsamples the data to maintain the invariance of the mapping.

3.3. Long Short-Term Memory (LSTM)

Recurrent Neural Networks (RNNs) are extensively utilized for time series analysis and prediction. [15]. LSTM is a particular type of RNN designed to address the long-term dependency issues faced by traditional RNNs when dealing with sequential data. The key feature of LSTM is its ability to maintain a cell state, which acts as a memory unit capable of storing information over long sequences. This capability enables LSTMs to selectively retain or discard information, making them appropriate for tasks requiring context and long-range dependencies. LSTMs have been widely adopted due to their gate control mechanism, which manages information flow and retains important data over extended periods.

LSTMs comprise three gates and memory cells: the input, output, and forget gates. These gates control the flow of information into and out of the memory cell. The figure illustrates the structure of an LSTM unit, where x_t and h_t denote the input and hidden states at time t. The forget, input, and output gates are represented by g_t, j_t, and o_t, respectively. The candidate cell state, representing new information for storage, is denoted by $\widetilde{C_t}$. The actual cell state C_t is updated based on the outputs of these gates.

In Figure 1, x signifies the input data; h indicates the hidden state, which provides memory capability to the network. The subscripts $t-1$ and t denote different time steps. The directed graph formed by the connections between nodes represents the sequence. The hidden state h_t is computed using the hidden state output from the previous layer and the current input. The operations of an LSTM unit are defined by the following equations:

Forget Gate: Determines which information to remove from the cell state.

$$g_t = \sigma(W_g \cdot [h_{t-1}, x_t] + b_g)$$

Input Gate: Incorporates new information into the cell state.

$$j_t = \sigma(W_j \cdot [h_{t-1}, x_t] + b_j)$$

$$\widetilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Cell State Update: Updates the cell state with both new and retained information.

$$C_t = g_t * C_{t-1} + j_t * \widetilde{C_t}$$

Output Gate: Produces the output based on the cell state.

DOI: 10.54254/2754-1169/2024.25234

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

In this context, the weight matrices for the forget, input, candidate, and output gates are denoted by W_g, W_j, W_C, and W_o, respectively. The corresponding bias terms are b_g, b_j, b_C, and b_o. The activation functions used are σ for sigmoid and $\tanh$ for hyperbolic tangent.

LSTMs have excelled in numerous tasks by effectively capturing long-term dependencies in sequential data. Their gate structures enable systematic maintenance and updating of information, making them ideal for applications requiring extended memory, including tasks like speech recognition, forecasting time series data, and processing natural language.

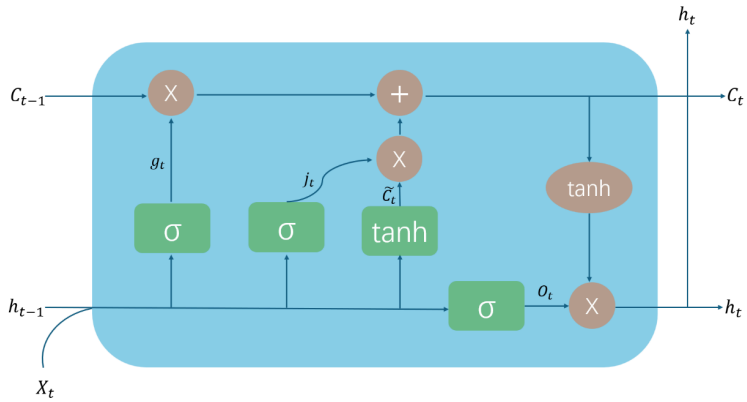

Figure 1. Operation of an LSTM unit

3.4. Bidirectional Long Short-Term Memory (BiLSTM)

Although the ability to capture long-term dependencies through sequential data processing is enabled, LSTMs can only consider the context of what has been seen before. This is because LSTMs solely use information up to the current time step and do not take into account future information. For instance, with time series prediction tasks, there could be information from the future that is itself additive to better capturing the underlying dynamics; hence, this unidirectional view is not sufficient.

The BiLSTM network improved upon it by having two LSTM layers: one that processes data from past to future and another in reverse. This double approach permits the network to access information in two directions for a better understanding of the sequence.

The network's ability to understand and predict time series data is improved by this bidirectional processing. For instance, in language processing tasks, understanding the context in which a word is used can depend on both the preceding and succeeding words. Similarly, considering both past market conditions and future expectations allows for better capture of trend and volatility patterns in financial time series prediction.

Let h_t^f represent the hidden state of the left-to-right LSTM at time step t, and h_t^b represent the hidden state of the right-to-left LSTM. In a BiLSTM, the output y_t at each time step can be expressed as:

DOI: 10.54254/2754-1169/2024.25234

$$h_t^f = \text{LSTM}_{\text{left-to-right}}\left(x_t, h_{t-1}^f\right)$$

$$h_t^b = \text{LSTM}_{\text{right-to-left}}\left(x_t, h_{t+1}^b\right)$$

$$y_t = h_t^f \oplus h_t^b$$

where $\oplus$ denotes the concatenation operation. This combined hidden state y_t integrates information from both directions, enhancing the network's ability to make more accurate predictions.

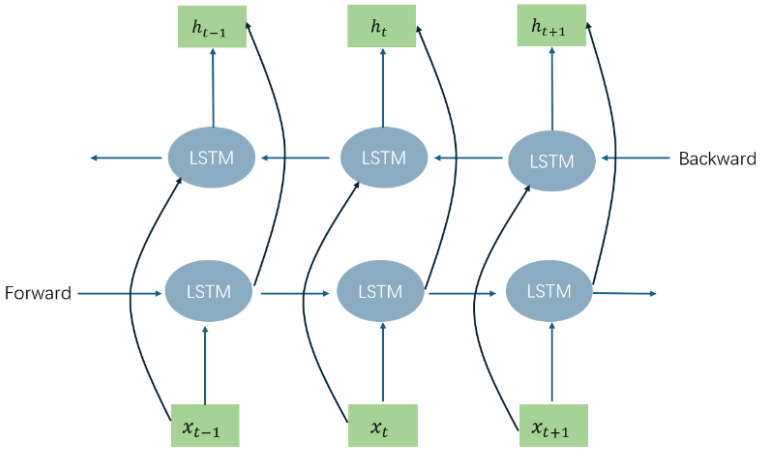

Figure 2. BiLSTM structure

3.5. Self-Attention mechanism

In the context of sequence modeling, the attention mechanism significantly enhances the capabilities of recurrent neural networks (RNNs) by enabling dynamic concentrate on various parts of the input sequence, particularly beneficial for long sequences. This mechanism operates through several key steps:

1.Linear Transformation: The hidden states of the bidirectional LSTM (BiLSTM) are transformed using a linear layer:

$$H^{'} = W_a H + b_a$$

where H is the BiLSTM output, W_a and b_a are learned parameters.

2.Non-linear Activation: The transformed states are then passed through a tanh activation function:

$$e_t = \tanh\left(H^{'}\right)$$

DOI: 10.54254/2754-1169/2024.25234

3.Attention Weights Calculation: These states are further transformed to compute the attention weights:

$$e_t = v_a^T e_t$$

where v_a is a learned context vector.

4.Normalization: The attention weights are normalized with the softmax function to yield probabilities.:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{t'} \exp(e_{t'})}$$

5.Context Vector Calculation: Finally, a context vector is the weighted sum of the hidden states:

$$c = \sum_{t=1}^{T} \alpha_t H_t$$

This self-attention method allows the model to capture long-range dependencies and concentrate on the most relevant sequence parts, improving the performance and interpretability of RNN-based models [14].

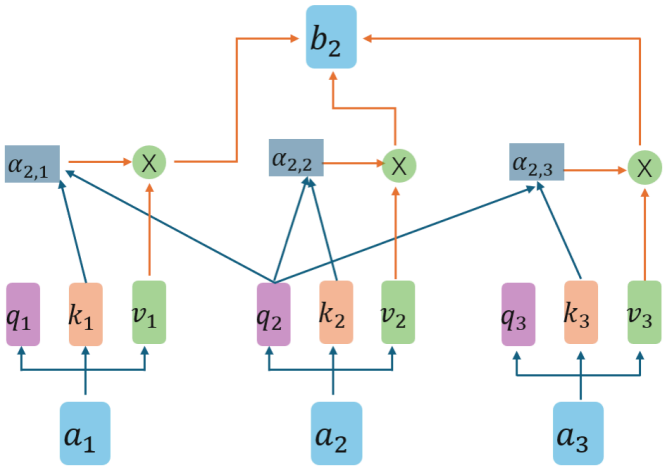

Figure 3. Structure of attention mechanism

4. The proposed network model structure

In order to efficiently extract features and enhance prediction accuracy, we combined CNN, BiLSTM, and a self-attention module into the CNN-BiLSTM-AM model, adding a residual block. This integrated framework autonomously learns and captures local and long-term features from time series data while minimizing complexity. The model's structure is depicted in Figure 4.

1.CNN Layer: Deep feature vectors are extracted from the input time series by the CNN model.

2.Residual Block: A residual block with three convolution layers (each with kernel size 3) is introduced between the CNN and BiLSTM layers to deepen the network without hindering training. The forward propagation in the residual block involves:

(a)Computing the residual connection.

DOI: 10.54254/2754-1169/2024.25234

(b)Processing input data through the first convolution layer followed by a ReLU activation function.

(c)Passing data through the second convolution layer.

(d)Adding the residual connection includes the output of the convolution layers.

(e)Applying the ReLU activation function again.

3.BiLSTM Layer: The BiLSTM model learns temporal features from the deep feature vectors constructed by the CNN.

4.Attention Mechanism (AM): The self-attention module extracts the most important features from the temporal data.

5.Dense Layer: The Dense model, with three fully connected layers, uses the fully connected layer to extract feature correlations through nonlinear mapping and then maps these to the output space. This setup addresses nonlinear problems effectively and achieves accurate predictions.

This comprehensive approach ensures that the model fully utilizes data information, enhancing prediction accuracy while maintaining a streamlined architecture.

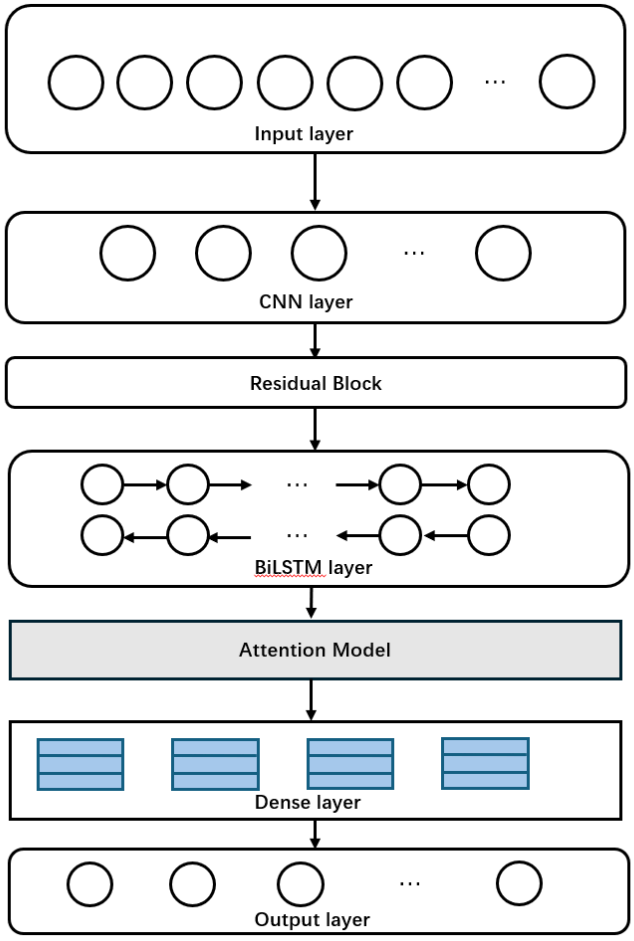

Figure 4. Proposed network model structure

DOI: 10.54254/2754-1169/2024.25234

5. Experiments

5.1. Experimental environment and data

The dataset utilized includes ten years of NVIDIA stock data, featuring daily closing prices, opening prices, highest and lowest prices, trading volume, turnover, price fluctuations and changes from May 2014 to May 2024, totaling 2520 trading days. This dataset was partitioned into three segments: 75% of the data for the training set, 10% for the validation set, and the remaining 15% for the testing set. First-order differencing was performed on the first four datasets to ensure the stationarity of their time series data.

5.2. Experimental process

Our model is used to predict historical volatility, with the experimental process described as follows:

1.Data Preprocessing: Eliminate non-essential elements from the experimental data. Convert time data to a serialized format. Normalize the data.

2.Model Training: The preprocessed time series data is fed into the CNN-BiLSTM-AM model to train it. The Optuna library was used to optimize hyperparameters such as convolution channels, kernel size, LSTM hidden dimensions, number of LSTM layers, and learning rate.

3.Prediction and Data Restoration: The testing data is fed into the trained model to obtain predictions, and then revert the predicted data to its original scale using the standardization formulas.

4.Result Evaluation: Create a comparison image showing the true versus predicted historical volatility. Assess the model's prediction accuracy by comparing the actual and predicted values using evaluation metrics such R-squared (R^2).

By following this comprehensive approach, the CNN-BiLSTM-AM model effectively leverages deep learning techniques to predict stock's volatility with high accuracy. The experimental process is illustrated in Figure 5.

DOI: 10.54254/2754-1169/2024.25234

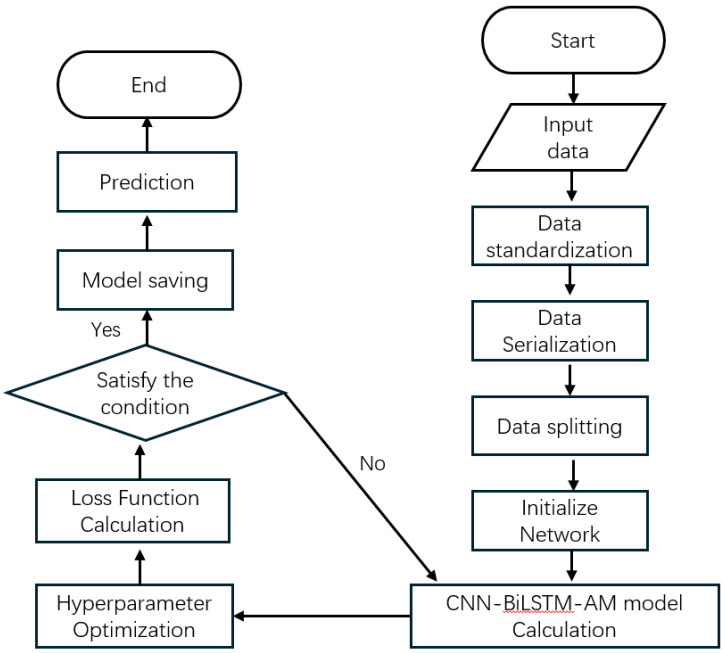

Figure 5. Diagram of model training and prediction

5.3. Experimental data preprocessing

The initial dataset is checked for missing values, which are either addressed by filling in or removing them to ensure smooth model training and testing. For data with intermittent gaps, these are addressed by using the average of the preceding and following trading days, given the minimal daily fluctuations in time series data. Additionally, data from non-trading days—such as weekends and major holidays—are excluded, leaving only the trading day data.

5.4. Experimental model and parameters

In this experiment, the CNN-BiLSTM-AM model is evaluated against several other models, including CNN-AM, RNN, LSTM-AM, CNN-LSTM-AM, CNN-BiLSTM, Support Vector Machine, and Random Forest. The parameters for the CNN-BiLSTM-AM model are outlined in Table 2, while the comparison models are configured with some of the same parameters used for our approach.

The CNN-BiLSTM-AM model includes specific parameter settings such as convolutional channels, kernel size, BiLSTM hidden dimensions, number of BiLSTM layers, and learning rate, optimized using the Optuna library. These settings ensure a fair comparison between different models in terms of performance and accuracy in predicting stock price volatility.

5.5. Experiment setting, implementation, and evaluation criterion

The developed CNN-BiLSTM-AM model architecture is composed of a CNN to feature identification and selection, followed by a BiLSTM layer with attention mechanisms to capture temporal dependencies and enhance prediction accuracy. The model was trained with a Mean Squared Error loss function and a learning rate of 0.005836147525921336 was determined by the Optuna optimizer. Training was conducted over 350 epochs, with the data divided into training

DOI: 10.54254/2754-1169/2024.25234

(75%), validation (10%), and test (15%) sets. Table 1 summarizes the parameter settings for the CNN-BiLSTM-AM model. To avoid overfitting, training and validation losses were tracked, and early stopping was implemented based on the validation loss.

Table 1. Parameter settings of CNN-BiLSTM-AM

Parameter	Value
Learning Rate	0.005836147525921336
Convolutional Layer Filters	128
Kernel Size	2
Conv1D padding	Same
Conv1D activation function	Sigmoid
Pooling layer activation function	Relu
BiLSTM Hidden Units	64
BiLSTM Layers	1
Attention Hidden Dimension	128
The number of layers in the Dense model	3
Epochs	350
Batch Size	64

The input layer takes in historical volatility sequences, which are processed through convolutional and residual blocks to extract features. These features are downsampled with max-pooling and then passed to a bidirectional LSTM layer. The attention mechanism refines the BiLSTM outputs, and a fully connected layer generates the final predictions. The detailed architecture is as follows:

Table 2. The model structure of CNN-BiLSTM-AM

Layer	Input Shape	Output Shape
InputLayer	(None, 10, 1)	(None, 10, 1)
Conv1D	(None, 10, 1)	(None, 10, 128)
ResidualBlock (Conv1D x2)	(None, 10, 128)	(None, 10, 128)
MaxPooling1D	(None, 10, 128)	(None, 5, 128)
Permute	(None, 5, 128)	(None, 128, 5)
Bidirectional (LSTM)	(None, 128, 5)	(None, 128, 128)
Attention (Linear)	(None, 128, 128)	(None, 128)
Dense	(None, 128)	(None, 1)
Flatten	(None, 128)	(None, 128)
Dense	(None, 128)	(None, 64)
Dense	(None, 64)	(None, 32)
Dense	(None, 32)	(None, 1)

To assess the forecasting performance of the CNN-BiLSTM-AM model, we used R-squared (R^2), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Mean Squared Error (MSE). These metrics are defined as follows:

DOI: 10.54254/2754-1169/2024.25234

Mean Absolute Error (MAE): Calculates the average size of errors in predictions, regardless of their direction. It is determined by:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|z_i - \widehat{z_i}\right|$$

Root Mean Square Error (RMSE): Calculates the square root of the average of squared differences between predictions and real observations. It is determined by:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(z_i - \widehat{z_i}\right)^2}$$

Mean Squared Error (MSE): Evaluates the mean of the squared errors. It is computed as follows:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(z_i - \widehat{z_i}\right)^2$$

R-squared (R^2): Reflects the ratio of variance in the dependent variable that is explained by the independent variables. It is computed as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(z_i - \widehat{z_i}\right)^2}{\sum_{i=1}^{n}\left(z_i - \bar{z}\right)^2}$$

By using these metrics, we were able to assess the accuracy and robustness of the CNN-BiLSTM-AM model in predicting stock price volatility. The results demonstrated that our model achieved high accuracy with low error rates, making it a reliable tool for financial forecasting.

6. Discussion

Table 3 ranks the models based on their MSE values, from highest to lowest. Figures 6-13 illustrate the errors between predicted and real values for each comparison model. Based on these errors, the CNN-BiLSTM-AM model demonstrates the most accurate prediction performance. The single LSTM-AM and CNN-AM models show the worst prediction capabilities. However, when CNN is combined with other models, the feature extraction by CNN allows the composite models to learn advanced features with stronger expressive capabilities. The prediction results have improved significantly and surpassed the traditional ANN and SVM models.

The BiLSTM model is more complex than the LSTM model, and this more complex structure does improve the model's prediction results for stock volatility: compared to CNN-LSTM-AM, CNN-BiLSTM-AM reduces MSE by about 20%, RMSE by about 10%, MAE by about 34%, and R^2 is closer to 1. Additionally, by comparing the CNN-BiLSTM model and the CNN-BiLSTM-AM model, we can see that the attention mechanism also significantly improves the model's prediction ability: MSE, RMSE, and MAE all decrease significantly. Thus, the CNN-BiLSTM-AM model is better suited for forecasting stock's volatility than the other models.

DOI: 10.54254/2754-1169/2024.25234

Table 3. Forecast errors of different network models

Model	MSE	RMSE	MAE	R^2
LSTM-AM	0.0030	0.0544	0.0290	0.8539
CNN-AM	0.0017	0.0414	0.0203	0.9154
Artificial Neural Network	0.0016	0.0401	0.0208	0.9207
Support Vector Machine	0.0016	0.0403	0.0304	0.9060
CNN-LSTM-AM	0.0015	0.0387	0.0211	0.9259
CNN-BiLSTM	0.0015	0.0385	0.0179	0.9270
Random Forest Regressor	0.0013	0.0354	0.0175	0.9274
CNN-BiLSTM-AM	0.0012	0.0348	0.0140	0.9403

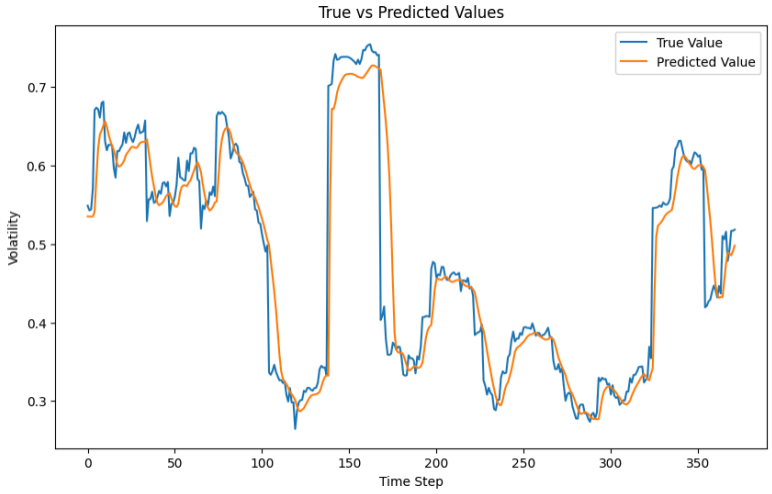

Figure 6. Comparison of LSTM-AM's predicted values and actual values

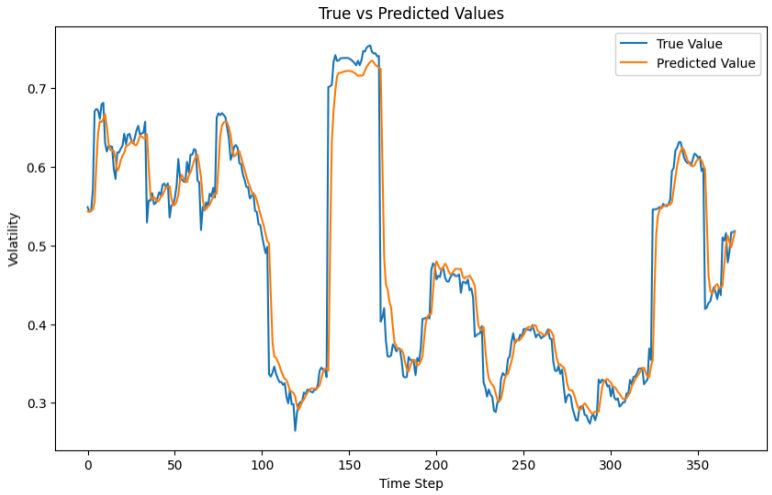

Figure 7. Comparison of CNN-AM's predicted values and actual values

DOI: 10.54254/2754-1169/2024.25234

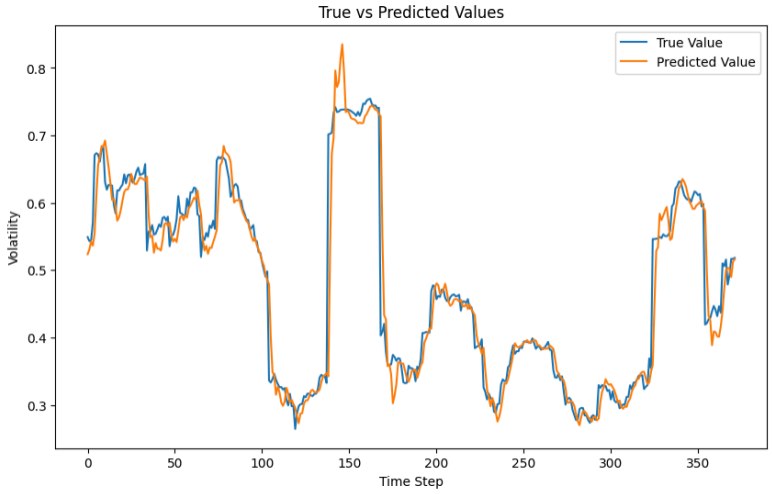

Figure 8. Comparison of ANN's predicted values and actual values

Figure 9. Comparison of SVM's predicted values and actual values

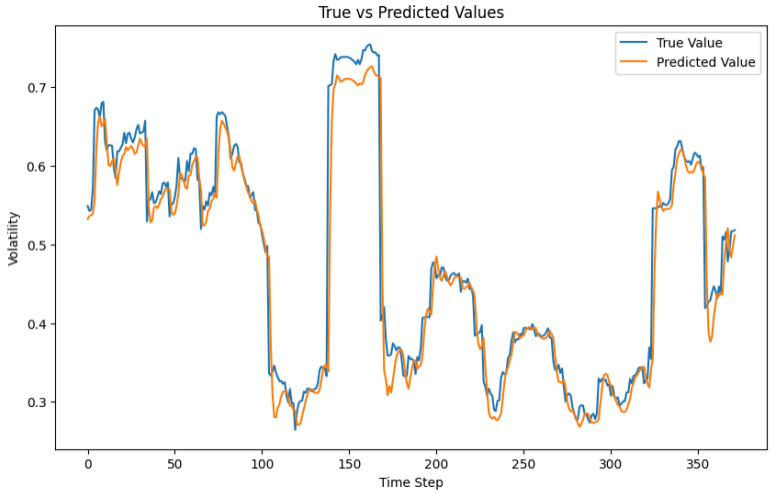

Figure 10. Comparison of CNN-LSTM-AM's predicted values and actual values

DOI: 10.54254/2754-1169/2024.25234

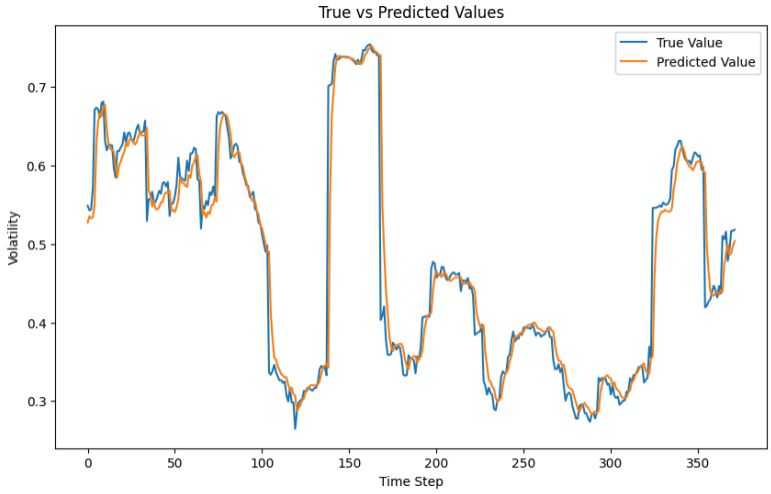

Figure 11. Comparison of CNN-BiLSTM's predicted values and actual values

Figure 12. Comparison of Random Forest Regressor's predicted values and actual values

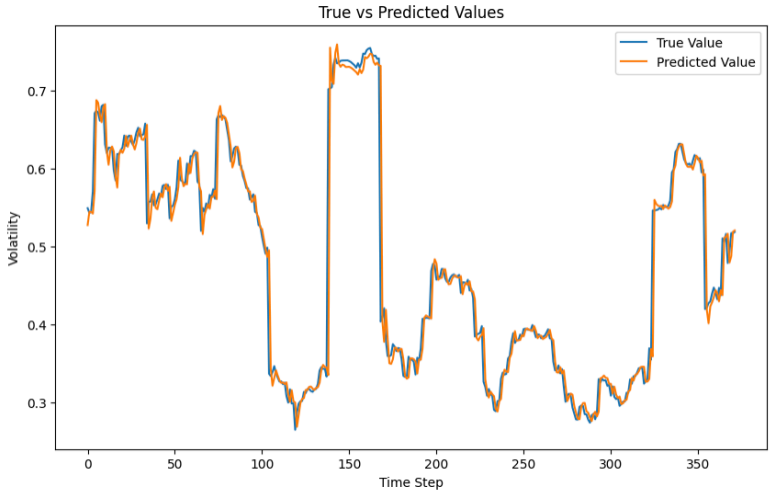

Figure 13. Comparison of CNN-BiLSTM-AM's predicted values and actual values

DOI: 10.54254/2754-1169/2024.25234

7. Conclusion

Predicting stock's volatility is essential for investment strategies. We introduce a novel forecasting model for stock volatility, named CNN-BiLSTM-AM, which utilizes historical stock data sets to forecast future volatility. This model integrates CNN and BiLSTM networks: Deep features are extracted from the input data by the CNN, while BiLSTM processes these features in a time series format for learning and prediction. Additionally, an attention mechanism is applied to prioritize key features, minimizing irrelevant ones and enhancing model accuracy. We evaluate the CNN-BiLSTM-AM model against seven other models, including LSTM-AM, CNN-AM, ANN, SVM, CNN-LSTM-AM, CNN-BiLSTM, and RandomForestRegressor. Our experiments show that the CNN-BiLSTM-AM model offers superior prediction performance for stock volatility.

7.1. Limitations of the model

Due to the model's high complexity, there is a risk of overfitting. The model might perform well on training data but have poor generalization on new data. More regularization and cross-validation are needed to mitigate overfitting.

The CNN-BiLSTM-AM model is highly complex and can thusly withstand greater computational resources and longer training times. This may result in a traffic jam during the processing of vast data or difficulty in making real-time predictions.

The economy is exposed to unanticipated external pressures like natural calamities, sudden political changes, and the global pandemic. Up to this time, these eventualities are without the model's predictability range.

7.2. Future work

There are several areas for future improvement and exploration:

Improving Data Volume: A decade is a short period to provide an appropriate forecast model. Yet, increasing the span of data to span across decades enables the model to account for various historical market trends and cycles as well as occasional anomalies. Models developed with this kind of data set would be more effective in training and producing highly resilient models too.

Introducing More Features: Adding attributes such as technical indicators, fundamental indexes, and investor behavior indices can enable the model to grasp the market behavior better. Technical indicators illustrate the short-term behavior of a stock, while fundamental metrics provide an economy-wide perspective, and sentiment indices depict individual investors' risk attitudes. Contrasting these aspects will increase the model's complexity and therefore allow for a better future predictive accuracy.

Applying to Different Markets: Using the asset price model in the various financial markets, including commodities, foreign exchange, and cryptocurrencies, is an important way of showing its validity. Every marketplace consists of unique factors that are of greatest significance in terms of determining whether the model can adapt and generalize. Thus, by taking the market to these fields, we risk that the utility and efficiency of the model will be more inclusive.

These improvements and expansions will enhance the robustness and applicability of the model, providing more reliable tools for financial forecasting and decision-making.

DOI: 10.54254/2754-1169/2024.25234

Acknowledgement

Haoyu Wang, Qiyu Liu, Zihe Fang, and Yunbo Wang contributed equally to this work and should be considered co-first authors.

References

[1] Abrishami, S.; Turek, M.; Choudhury, A.R.; Kumar, P. Enhancing Profit by Predicting Stock Pricesusing Deep Neural Networks. In Proceedings of the IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 4–6 November 2019; pp. 1551–1556.
[2] Aggarwal, S.; Aggarwal, S. Deep Investment in Financial Markets using Deep Learning Models. Int. J. Comput. Appl. 2017, 162, 40–43.
[3] Nguyen, T.T.; Yoon, S. A Novel Approach to Short-Term Stock Price Movement Prediction using Transfer Learning. Appl. Sci. 2019, 9, 4745.
[4] Lecun, Y.; Bengio, Y. Convolutional Networks for Images, Speech, and Time-Series; MIT Press: Cambridge, MA, USA, 1997.
[5] Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems, NIPS'12, Lake Tahoe, NV, USA, 3–6 December 2012; Curran Associates Inc.: Red Hook, NY, USA, 2012; Volume 1, pp. 1097–1105.14
[6] Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning (ICML), Haifa, Israel, 21–24 June 2010; pp. 807–814.
[7] Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. J. Mach. Learn. Res. 2014, 15, 1929–1958.
[8] Chen, J.F.; Chen, W.L.; Huang, C.P.; Huang, S.H.; Chen, A.P. Financial Time-Series Data Analysis Using Deep Convolutional Neural Networks. In Proceedings of the 7th International Conference on Cloud Computing and Big Data (CCBD), Macau, China, 16–18 November 2016; pp. 87–92.
[9] Sezer, O.B.; Ozbayoglu, A.M. Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. Appl. Soft Comput. 2018, 70, 525–538.
[10] Gross, W.; Lange, S.; B¨odecker, J.; Blum, M. Predicting Time Series with Space-Time Convolutional and Recurrent Neural Networks. In Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium, 26–28 April 2017; pp. 26–28.
[11] Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. Neural Comput. 1997, 9, 1735–1780.
[12] Kastner, S.; Ungerleider, L.G. Mechanisms of visual attention in the human cortex. Annu. Rev. Neurosci. 2000, 23, 315–341.
[13] Tran, D.T.; Iosifidis, A.; Kanniainen, J.; Gabbouj, M. Temporal attention-augmented bilinear network for financial time-series data analysis. IEEE Trans. Neural Netw. Learn. Syst. 2018, 30, 1407–1418.
[14] Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. Proc. IEEE 1998, 86, 2278–2324.
[15] Lu, W.; Li, J.; Li, Y.; Sun, A.; Wang, J. A CNN-LSTM-Based Model to Forecast Stock Prices. Complexity 2020, 2020, 6622927.