Hindawi
Mobile Information Systems
Volume 2022, Article ID 9915481, 8 pages
https://doi.org/10.1155/2022/9915481

*Research Article*

# Data Science Analysis Method Design via Big Data Technology and Attention Neural Network

**Yizhong Ren**

*College of Water Resources and Civil Engineering, China Agricultural University, Beijing 100083, China*

Correspondence should be addressed to Yizhong Ren; ryz@cau.edu.cn

Received 25 August 2022; Revised 18 September 2022; Accepted 26 September 2022; Published 11 October 2022

Academic Editor: Imran Khan

Because of the rapid expansion of big data technology, time series data is on the rise. These time series data include a lot of hidden information, and mining and evaluating hidden information is very important in finance, medical care, and transportation. Time series data forecasting is a data science analysis application, yet present time series data forecasting models do not completely account for the peculiarities of time series data. Traditional machine learning algorithms extract data features through artificially designed rules, while deep learning learns abstract representations of data through multiple processing layers. This not only saves the step of manually extracting features, but also greatly improves generalization performance for model. Therefore, this work utilizes big data technology to collect corresponding time series data and then uses deep learning to study the problem of time series data prediction. This work proposes a time series data prediction analysis network (TSDPANet). First, this work improves the traditional Inception module and proposes a feature extraction module suitable for 2D time series data. In 2D convolution, this solves the inefficiency of time series. Second, the notion of feature attention method for time series features is proposed in this study. The model focuses the neural network's data on the effectiveness of several measures. The feature attention module is used to assign different weights to different features according to their importance, which can effectively enhance and weaken the features. Third, this work conducts multi-faceted experiments on the proposed method.

## 1. Introduction

With the advancement of IoT and 5G technologies, society has begun to enter an era of mutual sensing and interconnected big data applications. Sensors put on objects can collect information in real time. This information is time series data, which is used to monitor the condition of an object and anticipate its future state. Time series data are a collection of ordered data acquired by the same entity at various times in time. For example, an e-commerce company's total monthly sales volume, the daily change in PM2.5 in a city, and the hourly traffic flow on a specific road. It is widely found in various fields such as industry, medical care, and finance. Usually, time series data contain rich information, mining, and analyzing this information can help people understand phenomena and predict the future. In the field of medicine, medical experts use continuous measurements of a patient's electrocardiogram to classify a patient's possible diseases. In the stock market, time series data consisting of daily closing prices in the history of the stock are analyzed to predict the stock price in the future. In the field of Internet security, network anomalies can be detected by analyzing network traffic data. In supply chain management, sales data for various products are forecasted and categorized. This makes it possible to plan the production and storage of the product in advance, avoiding the backlog of product that leads to additional costs. In addition, by categorical comparison with historical product sales data, a more rational plan for the production and sales of new products can be made. In industrial production, to improve the quality and safety of production, it is significant to monitor and collect data on complex equipment systems in real time. These data record the historical dynamic information of the system, can reflect the operation of the equipment, and play a very good role in improving management efficiency and rapid fault classification and diagnosis [1–5].

Deep learning is now a hot research area in artificial intelligence, owing to its ability to digest data by constructing a system comparable to the human brain. Deep learning models, in particular, process data through numerous hidden layers and map it to a high-level feature space. This can extract more abstract high-level data representation features or characteristics, and then investigate the feature representation of the data's implicit information. In addition to this, since deep learning computes high-level feature representations by directly inputting data. This does not require a lot of effort to do feature engineering and saves time and effort compared to traditional machine learning processes. Deep learning can learn abstract feature representations of data implicit information and can effectively model the correlation between sequence variables and long-term data dependencies. This can better understand and express the implicit information of the data, which improves accuracy. Analysis of time series data mainly includes time series forecasting and time series classification [6–10].

Time series forecasting finds the development trend of time series data by analyzing historical time series data. This can predict the possible value at the next moment or some time later. Traditional time series forecasting methods only perform linearly weighted forecasts based on historical values. This makes it difficult to model the nonlinearity of sequence data, resulting in low prediction accuracy. Traditional machine learning methods make predictions about future values by constructing time series features. Although it has excellent non-linear modeling ability, it does not consider the time dependence of sequence data and the correlation between variables, so it is difficult to achieve high-precision prediction. Not only does deep learning have excellent non-linear modeling skills, but recurrent neural networks and their derivatives can also effectively model long-term sequence dependencies. The self-attention mechanism captures the correlation between sequence data effectively, which is critical for boosting prediction accuracy. Time series classification examines the characteristics, shape, and other properties of the entire time series before assigning it a discrete name. Traditional time series classification methods identify and extract relevant information from sequences. When different patterns appear in sequences, redesign is time-consuming and labor-intensive. In addition, when the model cannot effectively extract different patterns of the sequence due to the preset rule method, the classification accuracy of the sequence will be seriously reduced, resulting in large errors in subsequent tasks. Deep learning can mine the feature representation of the implicit information of the data, which not only saves the complex feature engineering steps. At the same time, the classification accuracy and generalization ability of the model are greatly improved [11–15].

## 2. Related Work

The authors of reference [16] present a method for multistep forecasting that iteratively creates a model by minimizing the sum of squares of in-sample residuals from one step ahead. When the prediction length is reached, the predicted values are fed back into the same model to forecast the next set of data points. A nonlinear learning ensemble technique for multistep advance forecasting of long-term wind speed time series was proposed in reference [17]. The Ensem LSTM approach is built from an ensemble of LSTM, SVRM, and EO, and it learns time series features using LSTM clusters with varying hidden layers and neurons. The results of the LSTM network predictions are then aggregated into a nonlinear learning regression top layer consisting of SVRM, and the top layer parameters are optimized using the EO algorithm. Reference [18] proposed MARNN for multistep traffic flow prediction. It treats RNNs as dynamic networks for simulating dynamic features in traffic time series like recursive strategies and considers multioutput strategies for reducing accumulated error with increasing step size. Reference [19] proposes a successful prediction of extremely volatile and irregular financial time series data, such as stock market indices, was achieved by using a new self-evolving recursive neuro-fuzzy inference system. When applied to dynamic financial time series data, the method optimizes the model's parameters with MDHS technology, allowing for more accurate predictions than can be made with conventional neuro-fuzzy systems. There was a straightforward approach to multistage time series forecasting proposed in reference [20]. Through the use of historical observations, they constructed a set of predictive models for each horizon and then sought to reduce the squared multistep lead error associated with those predictions. The approach does not rely on the square of the one-step-ahead inaccuracy, and there are several direct-strategy multistep forecasting methods available as well. An ensemble approach involving decision trees, gradient boosted trees, and random forests was proposed in reference [21]. The weighted least squares method is used to obtain the ensemble weights, and a direct strategy is used for multistage time series forecasting of wind speed. In reference [22], a multioutput deep LSTM neural network model for multistep time advance prediction was developed. The dropout approach, L2 regularisation, and mini-batch gradient descent are all model components. This eliminates the error accumulation and propagation problems that are common in multistep forward predictions induced by overfitting during deep network training. Reference [23] proposed a model with a self-encoding-decoder structure for multistep advance prediction of time series. The basic structure of the model is composed of CNN and LSTM. This uses a convolutional network to learn correlations between variables and an LSTM network to learn sequential time series features. Reference [24] proposed a data augmentation model based on C-GAN to improve prediction performance. The method first uses a conditional generative adversarial network to learn a generator model. The model is used to generate new training data and integrate it into the original training data, thereby increasing the diversity of the training set.

The prediction accuracy of a neural network model used to estimate the hourly load value in reference [25] was much greater when compared to previous models. A direct feed of historical time series data is used to train the neural network. A CNN-based convolutional neural network model was created in the literature to handle the energy consumption time series prediction problem in the industrial production process [26]. By customizing convolution kernels in a number of directions, the model is able to extract the correlation features between input time series variables and the time-dependent characteristics of the same variable. The results of the experiments demonstrate that the model outperforms the neural network using a unidirectional convolution kernel in terms of generalization ability, prediction accuracy, and robustness. In order to make accurate predictions about periodic multivariate time series, the authors of reference [27] created a model called multiple CNN. Multiple CNNs are used to do a periodicity analysis of the time series, to extract proximity and long and short period information of the predictor variables, and to then combine the features of these three sections to produce predictions. Referring to reference [28], a fusion model was presented using CNN and LSTM to forecast immediate power needs. The experimental results demonstrate that the model outperforms a standalone CNN or LSTM in terms of prediction accuracy and robustness. Using a time series network to model long- and short-term dependency patterns in sequence data is proposed in reference [29]. It employs convolutional neural networks and recurrent neural networks to model intermediate-range dependencies in sequences, and it creates a novel recursive structure to capture very long-range dependencies in sequence data. In multivariate time series forecasting, it was proposed in reference [30] that CNN be used to extract time-invariant features from a single time series, and then RNN be used for time series prediction by merging the convolutional features of multiple time series. Two-stage attention was used to provide a model for RNNs in reference [31]. The usual attention approach is used at the decoder's input stage, where unique context vectors are generated at random intervals. The incorporation of an attention mechanism in the encoder's input stage of the two-stage attention model allows for the selection of feature factors and historical long-term temporal relationships. Reference [32] describes a unique focus strategy for multivariate time series forecasting. To get the hidden state output at any given time, the author first applies RNN to the original multivariate time series. The features for the hidden state variables are extracted using several convolution kernels, and then the attention mechanism is utilized to zero in on certain variables for the final prediction.

## 3. Proposed Method

First, this work improves the traditional Inception module and proposes a feature extraction module suitable for 2D time series data. This overcomes the inefficiency of time series in 2D convolution. Second, this paper proposes the concept of a feature attention mechanism for time series features. The model places the data focus of the neural network on the effectiveness of different metrics. The feature attention module is used to assign different weights to different features according to their importance, which can effectively enhance and weaken the features.

*3.1. CNN Algorithm.* Local connections, weighted nodes, pooling operations, and a multilayered architecture are all hallmarks of CNN networks. Therefore, the CNN network is better equipped to extract local features thanks to its local connections. Since the goal of the pooling operation is to minimize the dimensionality of the data, the sharing of weights has the additional benefit of drastically reducing the number of network parameters. As a result, the CNN network is also capable of representing time series data in a nonlinear fashion. The convolution kernel connects the preceding hidden layer's local region to the neurons of the convolutional layer. The convolution kernel is made up of weight matrices. When conducting the convolution procedure, the convolution kernel's sliding window travels through each section of the input matrix. It is multiplied by the matching elements of the area and aggregated after flipping to extract characteristics. Typically, many convolution kernels are established in the convolution layer, and distinct convolution kernels gain varied weights through network training, allowing different characteristics to be retrieved from the input. To improve nonlinear representation, the result of convolution operation is usually input into the nonlinear activation function to obtain a feature map. Therefore, multiple convolution kernels perform feature extraction on the previous hidden layer, and a series of feature maps can be obtained. The combination of multiple convolutional layers enables the CNN network to gradually extract more complex features.

$$x_j^k = f\left(\sum_i x_i^{k-1} k_{ij}^k + b_j^k\right). \tag{1}$$

The neurons of the pooling layer are also connected to the preceding hidden layer's local area, and a statistical value of the area is obtained using a specific mathematical procedure. This achieves dimensionality reduction while performing secondary feature extraction and enables the network to obtain a certain invariance to the feature changes of the input samples.

$$x_j^k = f\left(\beta_j^k down\left(x_j^{k-1}\right) + b_j^k\right). \tag{2}$$

Common pooling methods include max pooling, mean pooling, and random pooling. Maximum pooling achieves downsampling by maximizing the local area connected by the pooled kernel. Mean pooling implements downsampling by averaging the local regions of pooled kernel connections. Random pooling determines the probability matrix according to the element values in the local area, and randomly selects the output according to the probability matrix.

After convolution, the features will pass via the activation function layer, which is used to nonlinearly change the feature map formed by the convolution. The ReLU function is commonly employed as the activation function. The objective of the activation function in this case is the same as it is in the fully linked network: to boost the non-linear operation of the network. The ReLU function is generally selected because it has the characteristics of simple operation and easy convergence during training.

$$ReLU(x) = \max(0, x). \tag{3}$$

In the forward propagation process, CNN will randomly select data samples as input to enter the network, and the samples will enter the network in turn to obtain the output of the network. It should be noted that the most important network parameters are randomly initialized, and the initial values of network parameters will be randomly distributed in a small numerical range. Because a big initial value will effect network training, and if the parameters of each network are the same, training will not take place. When the outcome of this forward propagation is acquired, CNN will use the stochastic gradient descent process to update the network's parameters. The concept of minimizing the objective function is central to this method. The loss function determines how far the network's prediction was from the right label. Since the loss function is the objective function optimized by the stochastic gradient descent technique during training, choose the right one is essential. Backpropagation will update the network weights according to the current loss function through the algorithm to complete a round of network training.

*3.2. LSTM Algorithm.* LSTM is one of the improved models of RNN. As an improved model of RNN, it is mainly used to solve problems such as gradient disappearance. LSTMs, on the other hand, introduce new cellular states into the RNN and hence store long-term states. A gate structure controls the inside of the unit, which is used to increase or decrease the influence of different states on the present unit and to further process the information transmitted at the prior time point. The structure of the LSTM unit is demonstrated in Figure 1.

The forget gate determines which information should be discarded and which should be maintained in the cell unit state. It uses the Sigmoid function to handle the model's input at the present time and output at the previous moment as a vector of [0, 1]. The median of this vector represents the amount of information in the cell state to keep or discard.

$$f_t = \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right). \tag{4}$$

Input gates are primarily responsible for selectively recording new information into the cell state. It consists of two fully connected layers, the first layer decides which input information to retain, and the second layer will generate new candidate cell states based on the input data at this moment. The two layers together form an input gate to control how much of the input at the current moment will be retained to the final cell state.

$$\begin{aligned} i_t &= \sigma\left(W_i\left[h_{t-1}, x_t\right] + b_i\right), \\ \tilde{c}_t &= \tan h\left(W_c\left[h_{t-1}, x_t\right] + b_c\right), \\ c_t &= f_t c_{t-1} + i_{t-1}\tilde{c}_t. \end{aligned} \tag{5}$$

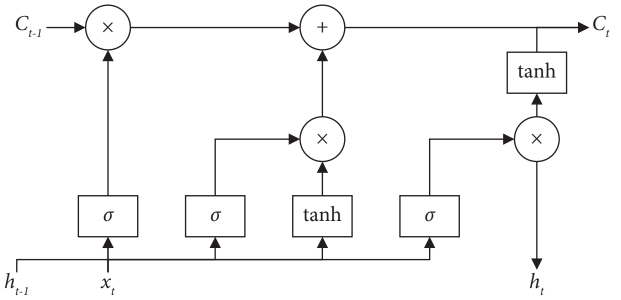

Figure 1: LSTM unit.

The output gate is responsible for controlling what information in the current cell state can enter the hidden layer of the output layer. It determines the output part of the cell state through a sigmoid layer. Then, go through the tanh layer and finally perform a dot product operation on the two to get the output.

$$\begin{aligned} o_t &= \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right), \\ h_i &= o_t * \tanh\ \left(c_t\right). \end{aligned} \tag{6}$$

LSTM networks are all unidirectional. Even if a multilayer recurrent neural network is added, the internal operation process is still only in a forward time sequence. The forward and reverse double guarantees are performed for sequence data to obtain information and improve the accuracy of sequence data prediction. BiLSTM came into being. It is composed of a forward LSTM and a backward LSTM in the horizontal direction. When inputting, BiLSTM will perform pre- and postorder calculations at the same time, and the two jointly determine the output through rules. Each unit of BiLSTM is composed of two LSTM cell units that are calculated in the forward and backward directions, and the horizontal comparison with a single LSTM unit can reflect the influence of the current moment and subsequent states on the current cell unit.

*3.3. TSDPANet for Time Series Data Forecast.* The TSDPANet model is designed to solve the decline for prediction accuracy due to the complexity of features when the network faces multiple input features. In this work, the attention mechanism is used for feature selection, which assigns higher weights to the features with more important prediction results and lowers the weights of irrelevant variables. In addition, the model is utilized after the input layer, while the feature weighting is calculated. This outputs the different weights the network assigns to each feature after iterative updates. The TSDPANet composite model established in this work is divided into three parts according to their functions: feature extraction module, feature attention module, and time series prediction module. The structure of TSDPANet is demonstrated in Figure 2.

This work proposes a fast feature extraction module for converting 2D feature photos to 3D features. The study integrated a huge number of input variables into the model, and the same type of indicators were placed next to each other once the data was processed. This lays the groundwork for extracting common features from nearby features using convolutional layers. The modified Inception feature extraction module will be used in this paper to be inserted between the input layer and the attention layer. This can

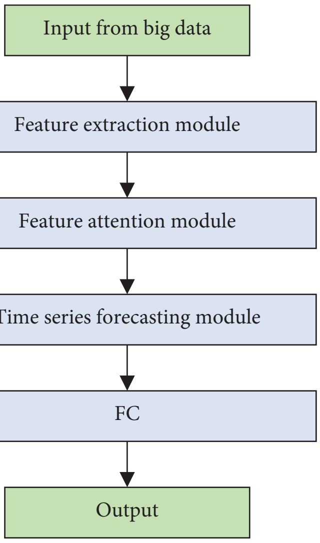

Figure 2: TSDPANet pipeline.

achieve independent extraction of correlated features in two different directions of input variables and time series. The Inception V1 module further extracts the information contained in the local location by computing between adjacent pixels in the data. Unlike image information, time-series data are only correlated at the same time or on the same metric. Therefore, ordinary matrices cannot extract the features of adjacent positions very well. Even add invalid information to the original information due to the participation of irrelevant information, thereby reducing the prediction accuracy. This module expands two-dimensional information into three-dimensional. Different from one-dimensional convolution, the two-dimensional convolution module will only properly blur the original feature matrix, but it will not regenerate the content of the feature direction like the former. Therefore, adding this feature extraction mechanism before the feature attention module will not cause the latter to fail in the feature direction.

The Inception structure must be modified in this work due to the mismatch between time series data and image samples. The optimized Inception's particular structure is shown in Figure 3, which ensures that the original information is not destroyed and allows for the efficient extraction of information between each index variable.

Initially, the attention mechanism assigns various weights to each layer based on the channel direction of different feature images created by convolution. By giving greater weights to feature layers that contain more effective information, SENET achieves efficient feature extraction and better classification accuracy. The CBAM module not only inherits the attention advantage of SENET model channel direction but also establishes the attention mechanism of feature map direction.

The feature attention mechanism in this paper adopts the channel attention module of the CBAM module, which consists of two encoder-decoder structures. The feature map is subjected to global max pooling and global average pooling, respectively, to obtain two different weight vectors whose length is the number of channels. Then, the multilayer perceptron is used to reduce and increase the dimension at a specific ratio, and activate the sigmoid function to jointly realize the nonlinear change of the weight. The two vectors are linearly added and multiplied by the original three-dimensional feature map, and the final output is a feature map that is added to the index by the attention weight. It first employs the Permute layer to transpose the information from the feature extraction module and exchange the feature dimension with the channel dimension. Then, on the spatial plane where the time and convolution layers are located, global max pooling and average pooling are conducted, and two vectors are created based on the order of the indicators. After sorting the Reshape layer into a one-dimensional vector, it is passed to the FC layer for dimensionality reduction, and the next FC layer is used to raise the dimension to the vector with the original number of dimensions. The obtained weight vector is dot-multiplied with the original feature map, and the Permute layer is used to exchange the dimension of the feature and the dimension of the channel back to the original position. It should be noted that the two vectors need to share weights when going through the FC layer for nonlinear transformation to ensure weight consistency.

This work selects BiLSTM to insert the composite model as the last module. This module receives the attention-weighted feature map, calculates it, and passes it to the fully connected layer for processing and outputs the final predicted value. The specific structure of the time series prediction module is demonstrated in Figure 4.

The number of neurons in the two Bi-LSTM layers is 64, which are used to receive feature information, and calculate the time series in both positive and negative directions to achieve the purpose of analyzing time series in a similar context. The dropout layer randomly removes each neuron of BiLSTM with a certain probability, so that every neuron in the BiLSTM layer can participate in the calculation and reduce the risk of overfitting. The Flatten layer reduces the BiLSTM layer's output to one dimension for use in the computation of future fully connected layers. The Flatten layer is utilized instead of the BiLSTM outputting only the last layer since this paper argues that the fully linked layer should have more options. The important information of each time point is obtained by iterative update, and the prediction accuracy is improved. The last two FC layers further extract the output information of the time series to obtain the optimal solution.

## 4. Experiment

*4.1. Experiment on TSDPANet Training.* This work first analyzes the training process of TSDPANet, this process is significant and necessary. And, the analysis target is the training loss. Its change process is demonstrated in Figure 5.

When the training epoch increases, the training loss of TSDPANet also decreases gradually. When the training level reaches around 120 epochs, the value of this parameter no longer changes significantly. This shows that TSDPANet has reached convergence at this time.

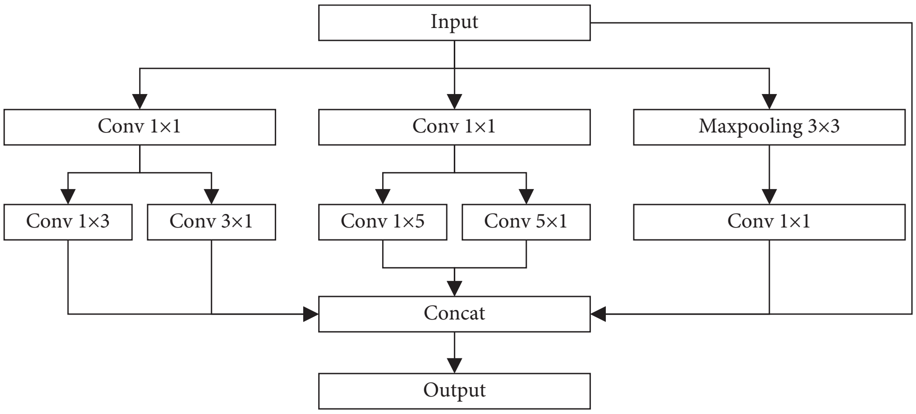

Figure 3: Feature extraction module.

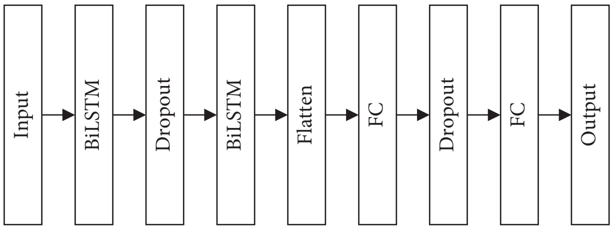

Figure 4: Time series prediction module.

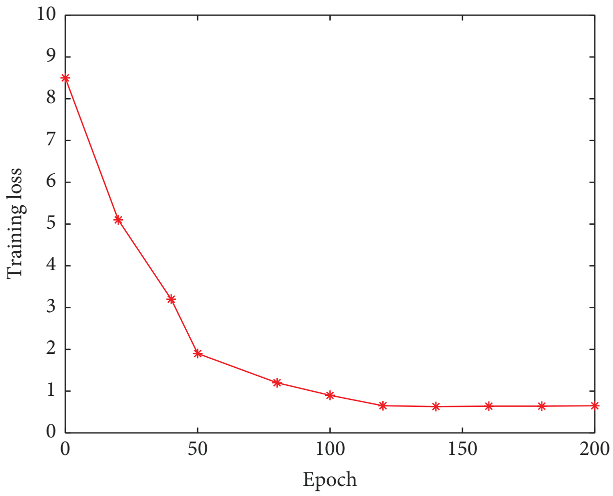

Figure 5: Training loss for TSDPANet.

*4.2. Method Comparison.* To verify the superiority of TSDPANet for time series data prediction, this paper compares the three methods of RNN, LSTM, and BiLSTM with TSDPANet. The indicators compared are RMSE and MAE, and the comparison data are demonstrated in Table 1.

Compared with other time series data prediction methods, TSDPANet achieves the lowest RMSE and MAE. This verifies the feasibility and superiority of applying TSDPANet to time series data prediction and also verifies the correctness of the method design in this paper.

*4.3. Experiment on Feature Extraction Module.* TSDPANet utilizes the improved Inception module as the feature extraction module. To verify the superiority of this improvement, this work compares the performance when using the traditional Inception module and the improved Inception module. The data obtained from the experiments are compared in Figure 6.

Table 1: Method comparison.

| Method | RMSE | MAE |
|---|---|---|
| RNN | 25.30 | 20.20 |
| LSTM | 22.70 | 17.50 |
| BiLSTM | 18.20 | 14.60 |
| TSDPANet | 15.60 | 12.30 |

TSDPANet can accomplish a certain degree of drop in both RMSE and MAE indicators after employing the upgraded Inception module compared to the standard Inception module. This comparative data verifies the correctness of the improvements to the Inception module.

*4.4. Experiment on CBAM.* TSDPANet utilizes the CBAM module as the attention module. To verify the superiority of CBAM, this work compares the performance when using CBAM and not using CBAM. The data obtained from the experiments are compared in Figure 7.

Compared with not using the CBAM module, after using the CBAM module, TSDPANet can achieve a certain degree of decline in both RMSE and MAE indicators. This comparative data verifies the correctness of using the CBAM module.

*4.5. Experiment on Time Series Prediction Module.* TSDPANet mainly uses BiLSTM as the processing module of time series data features. To verify the superiority of BiLSTM for time series data prediction, this work compares the performance of TSDPANet when using LSTM and BiLSTM. The data obtained from the experiments are listed in Table 2.

TSDPANet can achieve a certain degree of reduction in both RMSE and MAE indicators after employing the BiLSTM module in comparison to the standard LSTM module. This comparative data verifies the correctness of the BiLSTM module.

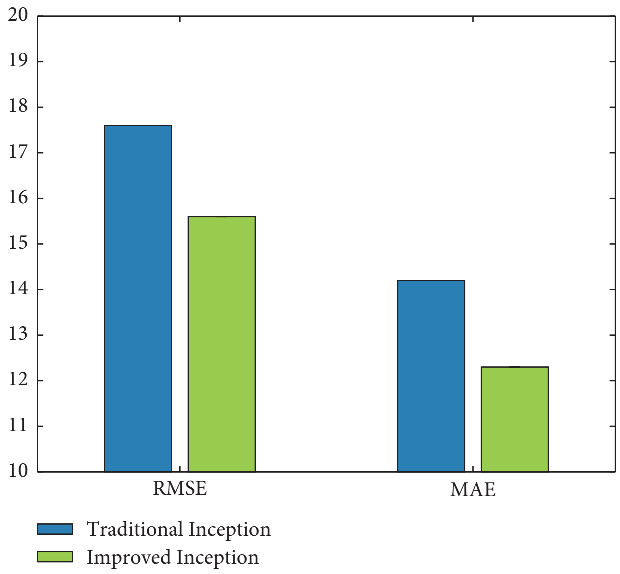

Figure 6: Comparison of Traditional Inception and improved Inception.

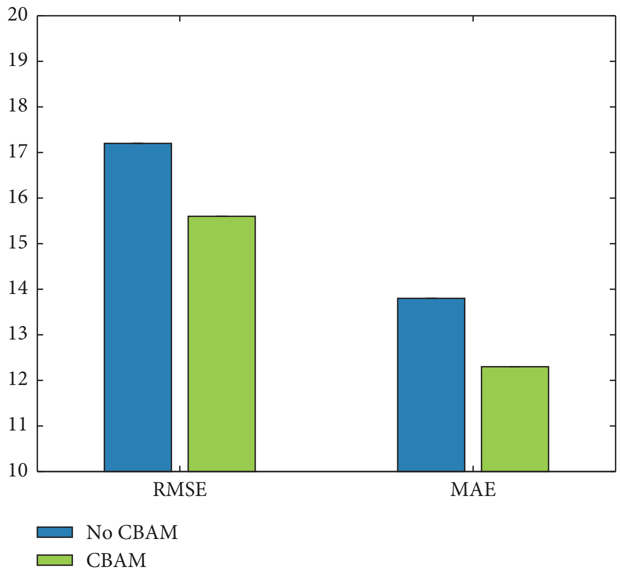

Figure 7: Comparison of using CBAM and not using CBAM.

Table 2: Comparison of using LSTM and BiLSTM.

| Method | RMSE | MAE |
|---|---|---|
| TSDPANet-LSTM | 17.10 | 13.50 |
| TSDPANet-LSTM | 15.60 | 12.30 |

## 5. Conclusion

We are now approaching the age of artificial intelligence, which is defined by increased connection and a greater capacity for shared understanding as a result of breakthroughs in big data and the Internet of Things. The current networked interconnectedness of physical goods and systems promises to provide a deluge of time series data. Decision-makers can direct and assist in making aggregate decisions through a thorough scientific examination of these time series data. As a result, the time series analysis issue piqued the interest of an increasing number of academics. However, most time series analysis problems necessitate custom-built features via traditional machine learning. As computational capabilities increased, deep learning found increasing success in a variety of contexts. Because deep learning can learn high-level feature representations of data's implicit information from beginning to end, it can replace laborious manual processes like feature engineering and extraction. Therefore, this paper will use deep learning methods to scientifically analyze time series data. This paper uses big data technology to collect corresponding time series data and then uses deep learning to study the problem of time series data prediction. This work proposes a time series data prediction analysis network. To begin, this work improves on the classic Inception module and presents a feature extraction module appropriate for 2D time series data. In 2D convolution, this solves the inefficiency of time series. Second, the notion of the feature attention method for time series features is proposed in this study. The model focuses the neural network's data on the effectiveness of several measures. The feature attention module is used to assign different weights to different features according to their importance, which can effectively enhance and weaken the features. Third, this work conducts multi-faceted experiments on the proposed method.

## Data Availability

The datasets used during the current study are available from the author on reasonable request.

## Conflicts of Interest

The author declares that there are no conflicts of interest.

## References

[1] B. Chen, Y. Liu, C. Zhang, and Z. Wang, "Time series data for equipment reliability analysis with deep learning," *IEEE Access*, vol. 8, pp. 105484–105493, 2020.

[2] Q. Wen, L. Sun, and F. Yang, "Time series data augmentation for deep learning: a survey," 2020, http://arxiv.org/abs/2002.12478.

[3] R. Yasrab, J. Zhang, P. Smyth, and M. P Pound, "Predicting plant growth from time-series data using deep learning," *Remote Sensing*, vol. 13, no. 3, p. 331, 2021.

[4] M. Dibaei, X. Zheng, Y. Xia et al., "Investigating the prospect of leveraging blockchain and machine learning to secure vehicular networks: a survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 683–700, 2022.

[5] S. El-Sappagh, T. Abuhmed, S. Riazul Islam, and K. S Kwak, "Multimodal multitask deep learning model for Alzheimer's disease progression detection based on time series data," *Neurocomputing*, vol. 412, pp. 197–215, 2020.

[6] S. Barra, S. M. Carta, A. Corriga, A. S. Podda, and D. R. Recupero, "Deep learning and time series-to-image encoding for financial forecasting," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 3, pp. 683–692, 2020.

[7] B. Yan and M. Aasma, “A novel deep learning framework: prediction and analysis of financial time series using CEEMD and LSTM,” *Expert Systems with Applications*, vol. 159, p. 113609, 2020.

[8] L. Feng, A. Ali, M. Iqbal, A. K. Bashir, S. A. Hussain, and S. Pack, “Optimal haptic communications over nanonetworks for E-health systems,” *IEEE Transactions on Industrial Informatics*, vol. 15, no. 5, pp. 3016–3027, 2019.

[9] H. Liu and Z. Long, “An improved deep learning model for predicting stock market price time series,” *Digital Signal Processing*, vol. 102, p. 102741, 2020.

[10] L. Ni, Y. Li, X. Wang, J. Zhang, J. Yu, and C. Qi, “Forecasting of forex time series data based on deep learning,” *Procedia Computer Science*, vol. 147, pp. 647–652, 2019.

[11] J. F. Torres, D. Hadjout, A. Sebaa, F. Martinez-Alvarez, and A. Troncoso, “Deep learning for time series forecasting: a survey,” *Big Data*, vol. 9, no. 1, pp. 3–21, 2021.

[12] J. Azar, A. Makhoul, R. Couturier, and J. Demerjian, “Robust IoT time series classification with data compression and deep learning,” *Neurocomputing*, vol. 398, pp. 222–234, 2020.

[13] Z. Zheng and A. K. Bashir, “Graph-enabled intelligent vehicular network data processing,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 5, pp. 4726–4735, 2022.

[14] H. Liang, “An intelligent prediction for sports industry scale based on time series algorithm and deep learning,” *Computational Intelligence and Neuroscience*, vol. 2022, p. 2345, 2022.

[15] I. E. Livieris, S. Stavroyiannis, E. Pintelas, and P. Pintelas, “A novel validation framework to enhance deep learning models in time-series forecasting,” *Neural Computing & Applications*, vol. 32, no. 23, pp. 17149–17167, 2020.

[16] G. Chevillon, “Direct multi-step estimation and forecasting,” *Journal of Economic Surveys*, vol. 21, no. 4, pp. 746–785, 2007.

[17] J. Chen, G. Q. Zeng, W. Zhou, W. Du, and K. D. Lu, “Wind speed forecasting using nonlinear-learning ensemble of deep learning time series prediction and extremal optimization,” *Energy Conversion and Management*, vol. 165, pp. 681–695, 2018.

[18] D. Yang, H. Yang, and P. Wang, “Multi-step traffic flow prediction using recurrent neural network,” in *Proceedings of the IEEE International Conferences on Ubiquitous Computing and Communications and Data Science and Computational Intelligence and Smart Computing*, pp. 803–808, IEEE, Shenyang, China, 21-23 October 2019.

[19] R. Dash and P. K. Dash, “Efficient stock price prediction using a self evolving recurrent neuro-fuzzy inference system optimized through a modified differential harmony search technique,” *Expert Systems with Applications*, vol. 52, pp. 75–90, 2016.

[20] D. R. Cox, “Prediction by exponentially weighted moving averages and related methods,” *Journal of the Royal Statistical Society: Series B*, vol. 23, no. 2, pp. 414–422, 1961.

[21] A. Galicia, R. Talavera-Llames, A. Troncoso, I. Koprinska, and F. Martinez-Alvarez, “Multi-step forecasting for big data time series based on ensemble learning,” *Knowledge-Based Systems*, vol. 163, pp. 830–841, 2019.

[22] Y. Zhou, F. J. Chang, L. C. Chang, I. F. Kao, and Y. S. Wang, “Explore a deep learning multi-output neural network for regional multi-step-ahead air quality forecasts,” *Journal of Cleaner Production*, vol. 209, pp. 134–145, 2019.

[23] N. C. Petersen, F. Rodrigues, and F. C. Pereira, “Multi-output bus travel time prediction with convolutional LSTM neural network,” *Expert Systems with Applications*, vol. 120, pp. 426–435, 2019.

[24] A. Koesdwiady, A. El Khatib, and F. Karray, “Methods to improve multi-step time series prediction,” in *Proceedings of the 2018 International Joint Conference on Neural Networks*, pp. 1–8, IEEE, Rio de Janeiro, Brazil, 08-13 July 2018.

[25] D. Srinivasan, A. C. Liew, and C. S. Chang, “A neural network short-term load forecaster,” *Electric Power Systems Research*, vol. 28, no. 3, pp. 227–234, 1994.

[26] Q. Xu, X. Hao, X. Shi, Z. Zhang, Q. Sun, and Y. Di, “Control of denitration system in cement calcination process: a novel method of deep neural network model predictive control,” *Journal of Cleaner Production*, vol. 332, p. 129970, 2022.

[27] K. Wang, K. Li, L. Zhou et al., “Multiple convolutional neural networks for multivariate time series prediction,” *Neurocomputing*, vol. 360, pp. 107–119, 2019.

[28] C. Tian, J. Ma, C. Zhang, and P. Zhan, “A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network,” *Energies*, vol. 11, no. 12, p. 3493, 2018.

[29] G. Lai, W. C. Chang, and Y. Yang, “Modeling long-and short-term temporal patterns with deep neural networks,” *International ACM SIGIR Conference on Research & Development in Information Retrieval*, , pp. 95–104, ACM, 2018.

[30] R. G. Cirstea, D. V. Micu, and G. M. Muresan, “Correlated time series forecasting using multi-task deep neural networks,” *ACM International Conference on Information and Knowledge Management*, , pp. 1527–1530, ACM, 2018.

[31] Y. Qin, D. Song, and H. Chen, “A dual-stage attention-based recurrent neural network for time series prediction,” 2017, http://arxiv.org/abs/1704.02971.

[32] S. Y. Shih, F. K. Sun, and H. y Lee, “Temporal pattern attention for multivariate time series forecasting,” *Machine Learning*, vol. 108, no. 8-9, pp. 1421–1441, 2019.