DOI: 10.54254/2755-2721/69/20241498

Research on time-series financial data prediction and analysis based on deep recurrent neural network

Feng Yuan

Boston University, Boston, Econometrics and Quantitative Economics, 02134-4589, China

yuanfeng3856@126.com

Abstract. Time series data is widely available in a variety of industries. By forecasting time series, decision-makers can better grasp future trends and make more effective decisions. Financial time series data exhibit non-stationarity and high volatility. High-frequency fluctuations in financial products such as exchange rates, bonds and equities may reflect external shocks and risks in global financial markets, which are potentially dangerous and may threaten national economic security or even trigger financial crises. For financial time series data, a deep recurrent neural network first progressively processes each data point in the time series through its recurrent unit. Each recurring unit can adjust its own weights to better predict or analyze future values. Over time, these recurrent units continuously update their internal state, resulting in a comprehensive understanding of the characteristics of the entire data sequence. In addition, we add a gating mechanism to further improve the network's ability to control the flow of information, so that the model is more effective when retaining long-term dependencies, so as to improve the accuracy of prediction and the stability of the model. Experimental results show that our recurrent neural network model shows higher prediction accuracy and stability than other baseline models on financial time series datasets.

Keywords: Time-series Data, Financial Data Prediction, Deep Recurrent Neural Network, Gated Recurrent Units.

1. Introduction

With the development of economic globalization, the volatility and uncertainty of financial markets are increasing, which makes accurate forecasting of financial time series crucial. Financial time series forecasting is a key research direction in the fields of policy making, investment decision-making, and risk management. Through a detailed analysis of this data, we are able to not only identify potential problems, but also capture new investment opportunities [1]. For example, in the stock market, accurate analysis of time-series data on stock prices can predict significant fluctuations in stock prices, which in turn can guide investors to adjust their portfolios in a timely manner to reduce risk. In addition, time series analysis also plays an important role in optimizing resource allocation, adjusting economic policies, and promoting economic growth.

Specifically, in the macroeconomic field, through the prediction and analysis of economic indicators, policymakers can formulate more scientific economic policies to promote economic development and social progress. With the advancement of information technology and data storage technology, the continuous generation and accumulation of large amounts of time series data has increased the demand for its analysis and prediction [2]. Time series analysis and forecasting is not only an important research method, but also has a wide range of academic and practical value for its theoretical and applied exploration.

Financial time series data has inherent characteristics such as nonlinearity, complexity, self-similarity, and timely variation, which make time series forecasting challenging. Early time series forecasting models were mostly based on statistical assumptions, such as traditional methods such as autoregressive and moving average, which were effective in dealing with relatively simple forecasting problems [3]. However, these traditional methods are often difficult to cope with when there are many predictors, high feature dimensions, and complex change patterns.

In addition, early methods often require manual selection and extraction of features from large amounts of data, a process that not only relies on deep domain knowledge, but is also time-consuming, labor-intensive, and inefficient when working with high-dimensional time series data. In order to improve the prediction efficiency, various methods based on machine learning have been proposed. Machine learning has a natural advantage in the field of time series forecasting because this type of forecasting problem is inherently closely related to regression analysis in machine learning [4]. Driven by big data and artificial intelligence technology, the methods and applications of time series analysis have ushered in new development opportunities. Compared with the traditional hypothesis testing methods, the machine learning model can deal with the covariate change problem in non-stationary time series more effectively, and construct a more reasonable prediction model.

In recent years, more and more research has tended to use various hybrid methods and machine learning models to analyze time series data, which have been widely used in practical applications. Observations for time series data are often related to previous time steps, and this dependency can persist for a long time and become more complex over time. However, most machine learning algorithms focus primarily on short-term local patterns and only consider recent data, making it difficult to capture long-term dependencies [5]. Choosing the right machine learning algorithm can also be challenging, as the characteristics of time series data are significantly different from those of other data types. For example, linear regression algorithms may not work well when working with time series data with nonlinear features.

2. Related Work

In the field of time series forecasting, traditional methods are mainly based on the theories of mathematics and statistics. These methods typically utilize linear regression or least squares regression analysis to establish the relationship between historical data and future trends to predict future series. Typical examples of this approach include autoregressive (AR) models, moving average (MA) models, and autoregressive moving average (ARMA) models, which provide a basic theoretical framework for time series forecasting.

In recent years, Chenghao et al. [6] have used the Autoregressive Integral Moving Average (ARIMA) model combined with online learning algorithms to optimize the ARIMA model estimation under the assumption of loose assumptions on the noise term, which improves the application breadth and computational efficiency. Additionally, researcher Sofiane et al. [7] introduced a Bayesian process model based on the nonstationary covariance function, which shows good adaptability to a wide range of problems due to its conceptual simplicity and good performance. At the same time, Ningning et al. [8] proposed a novel two-stage method, which combines ensemble empirical mode decomposition and multi-dimensional K-nearest neighbor algorithm to effectively predict the closing and highest prices of stocks.

As far as we can concern, a series of time series forecasting methods based on machine learning and neural networks have gradually emerged, and these methods have shown good results in the field of financial time series forecasting. For example, Milton et al. [9] developed an adaptive hybrid forecasting model using a combination of wavelet transform and support vector regression (SVR) to predict the financial time series of foreign exchange securities. The model mainly uses the Discrete Wavelet

Transform (DWT) to decompose the financial time series data, and the resulting components are used as the input variables of the SVR to achieve prediction.

Compared to the traditional ARIMA and ARFIMA models, this method shows more robust prediction performance. In addition, Manolis et al. [10] used a special random forest data mining technique to address the medium to high volatility of the stock market, and their data mining-driven trading strategy yielded higher profits than standard linear regression models and other machine learning classifier-guided strategies.

In recent years, with the continuous research of deep neural networks, Melih et al. [11] proposed a convolutional neural network (CNN) prediction model consisting of three convolutional layers and five fully connected layers. The model effectively maps the nonlinear relationship between input and output. In the prediction experiment of the Taiwan Stock Exchange's market capitalization-weighted stock index, this model has shown excellent prediction results compared with other advanced forecasting tools, such as artificial neural networks that fuse different methods, long short-term memory networks, fuzzy logic-based methods, and some traditional methods.

In addition, Ioannis et al. [12] developed a multi-input deep neural network model for predicting cryptocurrency price movements, which combines the advantages of CNN and LSTM and is specifically used for prediction of Bitcoin and Ethereum. Experimental analysis shows that compared with the traditional fully connected deep neural network, the proposed model can effectively use the mixed cryptocurrency data, reduce overfitting and merging, and reduce the computational cost. At the same time, Vidal et al. [13] significantly improved the prediction accuracy of gold volatility by integrating LSTM into a convolutional neural network, combining two deep learning techniques. Their CNN-LSTM hybrid model, which captures both static and dynamic features of sequences using images as input, has significant performance advantages over the CNN model and the LSTM model alone.

3. Methodologies

3.1. Deep Recurrent Neural Network

Deep recurrent neural networks are built by increasing the number of recurrent neural network layers, which can more effectively capture complex and abstract time series data features. Deep recurrent neural networks are particularly important in the context of financial time series forecasting because of their ability to resolve deep time dependencies in data, such as long-term trends and cyclical fluctuations. For a given input sequence $X = (x_1, x_2, \ldots, x_T)$, the deep recurrent neural network updates its hidden state h_t by the following Equation1:

$$h_t = f(W_t h_{t-1} + W_x x_t + b) \tag{1}$$

Where h_t is the hidden state of time step t. x_t is the input for time step t. W_t and W_x are the weight matrices that hide states and inputs. b is an offset. $f(\cdot)$ is the ReLU activation function.

Additionally, the output y_t is generated by the hidden state h_t at each time step, as shown in following Equation 2.

$$y_t = W_y h_t + c \tag{2}$$

Where W_y is the weight matrix of the output layer and parameter c represents the offset term of the output layer.

The input layer works by receiving raw time series data. The hidden layer is made up of multiple layers of recurrent neural networks stacked on top of each other. Each layer attempts to capture different levels of abstract information about the data. The output layer is a simple, fully connected layer that is used to output the final forecast results, such as financial data for the next time series.

The main advantage of stacked multilayer recurrent neural networks is the ability to extract abstract features of time series data layer by layer. Each layer of the recurrent neural network can learn from simple to complex data patterns, and these patterns can be deepened layer by layer. For example, the first layer may only learn basic patterns of price fluctuations, while deeper layers can identify complex trading strategies or market response patterns.

3.2. Gated Recurrent Units

The gated recurrent unit is an advanced recurrent neural network architecture used to solve the gradient vanishing problem in standard recurrent neural networks. As a variant of long short-term memory networks, the gated recurrent unit achieves a similar function with a more concise structure, but is computationally more efficient in some cases.

At the heart of the gated recurrent unit are two gating mechanisms including the update gate and reset gate. These two gates control the flow of information, allowing the gated recurrent unit to capture long-term dependencies in time series data while avoiding the problem of vanishing gradients. The update gate determines how much of the previous memory is retained to the current cell state. The function of update gate acts as a combination of a forgetting gate and an input gate, which is expressed as following Equation 3.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \tag{3}$$

Where z_t is the output of the update gate for time step t. W_z is the weight matrix of the update gate. h_{t-1} is the hidden state of the previous time step. x_t is the input for the current time step. b_z is the offset of the update gate. $\sigma(\cdot)$ is a sigmoid function that ensures the output is between 0 and 1.

Resetting the gate determines how much information from the past will be forgotten, which is critical for capturing short-term dependencies in a time series. The specific calculation for resetting the gate is expressed as following Equation 4.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{4}$$

Where r_t is the reset gate output of time step t. W_r is the weight matrix for the reset door. Further, the candidate hidden state is a temporary state that combines the current input with the hidden state after the previous time step was adjusted by the reset gate to calculate the final hidden state. Finally, the final hidden state is up to the update gate to decide how much of the old state and how many new candidate states to keep.

The gated recurrent unit maintains similar functionality by simplifying the structure of the proposed model through its design of update and reset gates. The update door helps the model decide how much old information to keep in the new hidden state, while the reset door controls how much previous information should be forgotten. This mechanism allows the gated recurrent unit to efficiently capture both long-term and short-term dependencies when processing time series data, while maintaining computationally efficiency.

4. Experiments

4.1. Experimental Setups

We use the proposed deep recurrent neural network as the main model, which is configured as a two-layer structure with 128 and 64 neurons, respectively. The experiment was trained with a batch size of 64 samples, 50 training cycles, and the Adam optimizer with an initial learning rate set to 0.001. To reduce overfitting, we applied a 50% drop rate after each layer. The Mean Square Error (MSE) was selected for the loss function to accurately assess the prediction error.

This experiment mainly uses the Exchange public exchange rate dataset, which covers the daily exchange rate data of eight different countries from 1990 to 2016. During the experiment, a series of sequential time series samples are fed to the model, and the model then outputs the prediction data. The core of the experiment is to use the error index to evaluate the deviation between the predicted value of the model and the actual value, and use this to measure the prediction performance of each model. The smaller the error index, the better the prediction performance of the model. Conversely, a larger indicator indicates poor performance.

4.2. Experimental Analysis

Training error is a key metric for evaluating the performance of a machine learning model on a learning dataset, and it measures the difference between the predicted and actual results of the model during training. Specifically, training errors can reveal how accurate a model is when processing training data, and are often used to adjust model parameters, optimize algorithms, and compare the learning capabilities of different models. Following Figure 1 compares the training errors with existing prediction models for financial data.

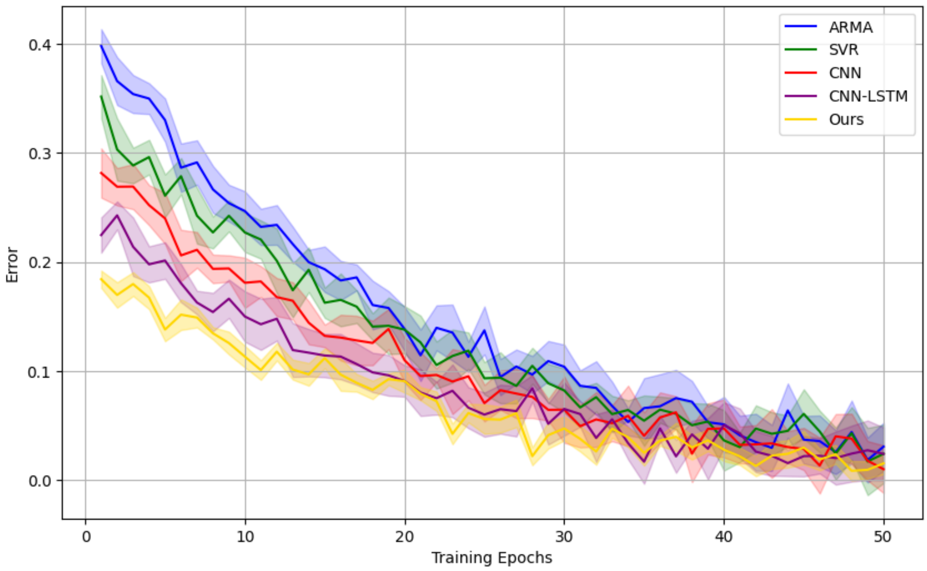

Figure 1. Training Error Comparison Across Prediction Models.

The error comparison graph shows that the training error of each model decreases nonlinearly and contains some random fluctuations, making the curve more natural and not completely smooth. The graph shows that the error of all models gradually decreases with the training cycle, but each model has small ups and downs during the descent, which is more in line with the actual training process.

The average percentage error is a scale measure of forecast accuracy that averages the ratio of the difference between the predicted and actual values and the actual values. A low average percentage error indicates better prediction performance, especially when data sets of different sizes need to be compared. Following Figure 2 compares the average percentage error results.

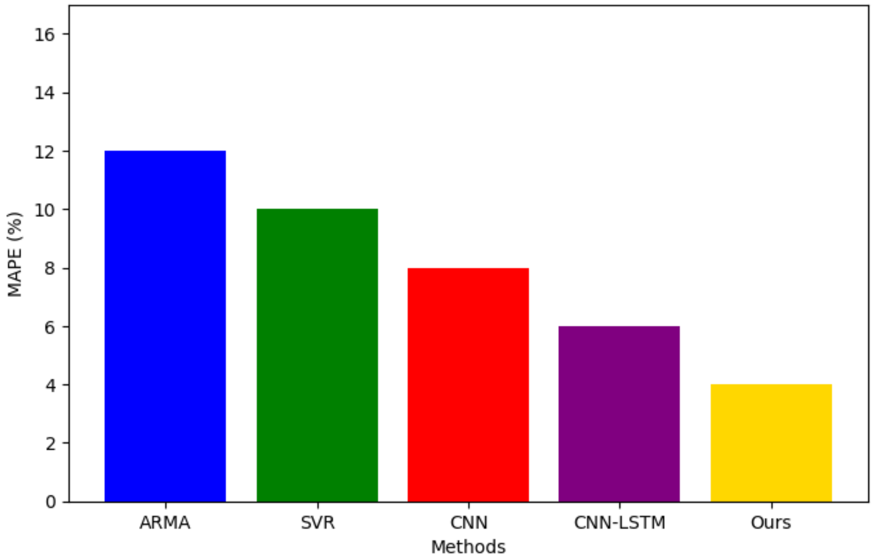

Figure 2. Comparison of Mean Absolute Percentage Error Across Models.

As can be seen in Figure 2, our method has the lowest MAPE value of only 4%, indicating that our model performs optimally in prediction accuracy relative to other methods. The percentages labeled above each histogram show the size of the error for each model, with lower values indicating that the model's predictions deviate less from the actual values and the better the performance.

5. Conclusion

In conclusion, we utilize deep recurrent neural networks to effectively predict and analyze time-series financial data. Our experiments show that this approach performs well in capturing the time-dependent and non-linear nature of the data, providing accurate predictions of market trends and value movements. Despite the challenges of overfitting and data scarcity, future research will focus on model optimization and integration with traditional financial theories to improve the applicability and robustness of models in real financial markets.

References

[1] Wu, Junran, et al. "Price graphs: Utilizing the structural information of financial time series for stock prediction." Information Sciences 588 (2022): 405-424.
[2] Masini, Ricardo P., Marcelo C. Medeiros, and Eduardo F. Mendes. "Machine learning advances for time series forecasting." Journal of economic surveys 37.1 (2023): 76-111.
[3] Cheng, Dawei, et al. "Financial time series forecasting with multi-modality graph neural network." Pattern Recognition 121 (2022): 108218.
[4] Liu, Zhenyu, et al. "Forecast methods for time series data: a survey." Ieee Access 9 (2021): 91896-91912.
[5] He, Kaijian, et al. "Financial time series forecasting with the deep learning ensemble model." Mathematics 11.4 (2023): 1054.
[6] Liu, Chenghao, et al. "Online arima algorithms for time series prediction." Proceedings of the AAAI conference on artificial intelligence. Vol. 30. No. 1. 2016.
[7] Brahim-Belhouari, Sofiane, and Amine Bermak. "Gaussian process for nonstationary time series prediction." Computational Statistics & Data Analysis 47.4 (2004): 705-712.
[8] Zhang, Ningning, Aijing Lin, and Pengjian Shang. "Multidimensional k-nearest neighbor model based on EEMD for financial time series forecasting." Physica A: Statistical Mechanics and its Applications 477 (2017): 161-173.

DOI: 10.54254/2755-2721/69/20241498

[9] Zhang, Li, et al. "Iterated time series prediction with multiple support vector regression models." Neurocomputing 99 (2013): 411-422.

[10] Raimundo, Milton Saulo, and Jun Okamoto. "SVR-wavelet adaptive model for forecasting financial time series." 2018 International Conference on Information and Computer Technologies (ICICT). IEEE, 2018.

[11] Kirisci, Melih, and Ozge Cagcag Yolcu. "A new CNN-based model for financial time series: TAIEX and FTSE stocks forecasting." Neural Processing Letters 54.4 (2022): 3357-3374.

[12] Livieris, Ioannis E., et al. "An advanced CNN-LSTM model for cryptocurrency forecasting." Electronics 10.3 (2021): 287.

[13] Vidal, Andrés, and Werner Kristjanpoller. "Gold volatility prediction using a CNN-LSTM approach." Expert Systems with Applications 157 (2020): 113481.