mathematics

MDPI

Article

On Stock Volatility Forecasting under Mixed-Frequency Data Based on Hybrid RR-MIDAS and CNN-LSTM Models

Wenfeng Ma [1,†], Yuxuan Hong [2,†] and Yuping Song [1,*]

[1] School of Finance and Business, Shanghai Normal University, Shanghai 200234, China; 1000515975@smail.shnu.edu.cn
[2] College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai 201418, China; 1000555508@smail.shnu.edu.cn
* Correspondence: songyuping@shnu.edu.cn; Tel.: +86-021-64324524
† These authors are co-first authors who have contributed equally to this work.

Abstract: Most of the deep-learning algorithms on stock price volatility prediction in the existing literature use data such as same-frequency market indicators or technical indicators, and less consider mixed-frequency data, such as macro-data. Compared with the traditional model that only inputs the same-frequency data such as technical indicators and market indicators, this study proposes an improved deep-learning model based on mixed-frequency big data. This paper first introduces the reserve restricted mixed-frequency data sampling (RR-MIDAS) model to deal with the mixed-frequency data and, secondly, extracts the temporal and spatial features of volatility series by using the parallel model of CNN-LSTM and LSTM, and finally utilizes the Optuna framework for hyper-parameter optimization to achieve volatility prediction. For the deep-learning model with mixed-frequency data, its RMSE, MAE, MSLE, MAPE, SMAPE, and QLIKE are reduced by 18.25%, 14.91%, 30.00%, 12.85%, 13.74%, and 23.42%, respectively. This paper provides a more accurate and robust method for forecasting the realized volatility of stock prices under mixed-frequency data.

Keywords: mixed frequency data; RR-MIDAS model; volatility prediction; deep learning; Optuna framework

MSC: 68T09; 62P05; 68T07

JEL Classification: C32; G17

Citation: Ma, W.; Hong, Y.; Song, Y. On Stock Volatility Forecasting under Mixed-Frequency Data Based on Hybrid RR-MIDAS and CNN-LSTM Models. *Mathematics* **2024**, *12*, 1538. https://doi.org/10.3390/math12101538

Academic Editors: Consuelo Nava, Tiziana Ciano and Massimiliano Ferrara

Received: 21 April 2024
Revised: 12 May 2024
Accepted: 14 May 2024
Published: 15 May 2024

Correction Statement: This article has been republished with a minor change. The change does not affect the scientific content of the article and further details are available within the backmatter of the website version of this article.

1. Introduction

The volatility of the financial market is an important indicator for measuring the degree of price fluctuations of financial assets, which plays a very important role in practical applications such as investment decision making [1], asset pricing [2], and risk management [3]. Therefore, the construction of a more accurate volatility estimation and forecasting model has very important theoretical value and application significance.

The research on volatility estimation and prediction has gone through the process from using low-frequency data to using high-frequency data for prediction. In the context of low-frequency data, Engle [4] proposed an autoregressive conditional heteroskedasticity (ARCH) model that considers changes in volatility. On this basis, Bollerslev [5] proposed a generalized autoregressive conditional heteroskedasticity (GARCH) model in order to better characterize the heteroskedasticity of the residuals of financial asset return series. Based on the assumption that conditional variance obeys an unobservable stochastic process, Taylor [6] proposed the stochastic volatility (SV) model. However, the drawback of the above models is that a large amount of intraday price information is lost when estimating stock volatility, and high-frequency data can provide more fine-grained price fluctuation information, which can help improve the accuracy of volatility estimation.

With the ease and decreasing cost of access to high-frequency data, high-frequency data have become more common in the study of financial asset return volatility, which provides a new entry point for financial volatility forecasting. In the context of high-frequency data, Andersen et al. [7] proposed an estimator of realized volatility based on the sum of squares of intraday high-frequency returns as a new way to measure daily volatility. Corsi [8] proposed the heterogeneous autoregression (HAR) model for financial volatility forecasting, which introduces information about historical volatility over different time scales to more sensitively capture volatility's short-term and long-term changes. However, the complexity and variability of the financial market itself means that more factors need to be considered when building forecasting models, like macro-factors, sentiment factors [9], and investor attention factor [10], etc. Furthermore, the relationship between these variables is often nonlinear and dynamically changing, which makes it difficult for traditional time-series models to adapt to changes in the market, and the forecasting accuracy is limited. In addition, traditional econometric models can only be applied to the modeling of smooth series, but the price series in financial markets are often trending and seasonal, which makes it difficult for them to meet the requirement of smoothness.

In recent years, nonlinear machine-learning models have been gradually adopted in financial time-series forecasting and can fully exploit the nonlinear relationship between variables and strong feature-learning ability, which improves the forecasting performance of the models to some extent. Currently, nonlinear machine prediction models include support vector machines and random forests, etc. For example, Liu et al. [11] predicted the volatility of the shipping market index using the AR-SVR-GARCH model, Li and Qiao [12] predicted the realized the volatility of the CSI 300 index using the SW-SVR model. To predict the share price movement of a clean energy exchange-traded fund, Sadorsky [13] predicted the volatility of the Clean Energy Exchange Traded Fund (CETF) using random forests, and Zhuo and Morimoto [14] used a hybrid HAR model and SVR model to forecast realized volatility. However, the ability of machine learning to portray the correlation of data before and after a financial time series is poor, and deep-learning models have been gradually developed to further improve the prediction accuracy and better express the correlation of data before and after a time series. Long-short-term memory (LSTM) networks, as an improved model of recurrent neural network (RNN), focus on coping with long-term dependence and do not require complex hyper-parameter tuning, and they are able to automatically memorize the historical information of a longer period of time, but the LSTM itself has a relatively complex model structure and is computationally intensive when the time span of the data is large. Convolutional neural networks (CNN) are able to learn the temporal and spatial features of time series without complex information processing by using convolutional and pooling layers as feature extractors [15]. However, CNN is deficient in capturing the long-term serial features of financial time series. Zhou [16] found that the deep-learning model CNN-LSTM has a strong learning ability and overfitting resistance to nonlinear relationships. Lu et al. [17] and Chen [18] utilized the CNN-LSTM model for stock price prediction and found that the model prediction accuracy was higher. However, CNN-LSTM models in the existing literature mainly focus on stock price trend prediction and less on the prediction of stock price volatility.

In addition, most of the above models consider using the same frequency of market indicators or technical indicators to predict the time series. More existing studies show that macro-variables play an important role in predicting stock market volatility; for example, Amendola et al. [19] use the GARCH-MIDAS model to study the asymmetric effect of macro-variables on stock volatility, Shang and Zheng [20] introduce an SV-MIDAS model with input macro-variables to predict the stock price volatility, and Li et al. [21] use a GARCH-MIDAS model to introduce macro-variables to predict stock volatility under economic policy uncertainty. Compared with stock market data and technical indicators, macro-variables are usually low-frequency variables. Currently, when dealing with mixed-frequency data, some scholars use linear interpolation to deal with the mismatch between high-frequency and low-frequency information, but when dealing with financial time-

series data, linear interpolation may lead to distortion of the trend, resulting in the loss of information. Ghysels et al. [22] proposed that the MIDAS model can frequency-align high-frequency variables into low-frequency variables, combining data from different frequencies to predict volatility. However, MIDAS has high computational complexity and performs poorly when dealing with nonlinear relationships, and more flexible methods need to be considered to deal with datasets with complex nonlinear relationships. RU-MIDAS, proposed by Foroni et al. [23], enables the prediction of the trend of a low-frequency variable to a high-frequency variable, but it can achieve good empirical results only when the frequency multiplicity difference is small. The RR-MIDAS model proposed by Xu et al. [24] simplifies the calculation process and still shows good prediction accuracy when the frequency multiplicity difference is 22. Wu et al. [25] used the GARCH-MIDAS model to process the mixed-frequency data and predict volatility and found that the model was more accurate and robust. In summary, when using mixed-frequency data to predict volatility, more scholars have noticed the advantages of the MIDAS model in mixed-frequency data processing, and more scholars only use the MIDAS econometric model to homogenise and predict mixed-frequency data without considering the advantages of the deep-learning model in multivariate time-series data feature learning and data prediction.

The introduction of macro-factors in forecasting stock price volatility is necessary to fill the gaps in existing research, taking into account the dual impact of macro-factors on business operations and discount rates. Aiming at the differences in data frequency between markets, fundamental and macro-factors and realized volatility, as well as the problem of information loss that may be caused by traditional homogenization processing, this paper proposes the RR-MIDAS-CNN-LSTM-PARALLEL (RM-CNN-LSTM-P) model. First, the data with different frequencies are reverse-mixed using the RR-MIDAS model. Then, the spatial and temporal features of the time series are extracted from market indicators, fundamental indicators, and macro-variables by CNN-LSTM, respectively. Meanwhile, the temporal features of realized volatility are processed using a parallel LSTM network to capture its dynamic changes. To further enhance the model performance, the Optuna framework is employed to tune the model hyperparameters. Finally, the extracted temporal and spatial features are fed into the fully connected layer to predict the realized volatility at the next moment.

This paper makes four contributions in stock price volatility forecasting. First, this study recognizes the impact of macro-factors on firms' operations and cash flow discount rates, and therefore introduces macro-indicators in addition to traditional market and fundamental indicators to enhance the forecasting accuracy of realized stock price volatility. Second, for the frequency difference between macroeconomic data and stock market data, this study adopts the RR-MIDAS model to deal with the problem of mixed-frequency data, which effectively avoids the information distortion and estimation bias that may be caused by the traditional interpolation method, and significantly improves the volatility prediction performance. Then, in terms of model construction, this paper introduces an LSTM network in parallel on the basis of CNN-LSTM architecture, which is specifically used to extract the temporal features of realized volatility, and this improvement significantly improves the prediction accuracy of the model. In addition, considering the high number of hyperparameters of deep-learning models, this study utilizes the Optuna framework to tune the model hyperparameters and verifies the superiority of the proposed model by comparing it with other commonly used models. Finally, to test the robustness of the model, this paper compares the prediction results of the RM-CNN-LSTM-P model after 500 experiments with 17 other models and performs the DM test, which shows that the model not only predicts accurately, but also has high robustness.

The structure of this paper is as follows: Section 2 introduces the principles of the econometric model RR-MIDAS and the deep-learning model CNN-LSTM-P, comprehensively summarizes the evaluation criteria and the principles of the DM test, and then details the research process. Section 3 introduces the data sources, basic data characteristics and Optuna tuning framework, argues the correlation between explanatory variables and

volatility, and demonstrates the optimal hyperparameters of the RM-CNN-LSTM-P model under the Optuna tuning framework. The test set volatility prediction accuracies and rankings of 18 volatility prediction models under six loss functions are further compared, and finally, influence experiments and DM tests are conducted to argue the importance of the three macro-variables in improving the prediction accuracy of the models, and to verify that the RM-CNN-LSTM-P model is significantly better than the other models. Section 4 discusses the difference in prediction performance between this paper and the linear interpolation model, and the parallel LSTM model with CNN removed, and Section 5 provides conclusions and extensions.

2. Methodology and Evaluation Criteria

2.1. RR-MIDAS Model

When predicting changes in stock price volatility, the problem arises of predicting high-frequency data with low-frequency data, and there may be a large frequency multiplicity difference between the variables. The mixed-frequency data sampling (MIDAS) model and the reverse unconstrained mixed-frequency data sampling (RU-MIDAS) model proposed by Foroni et al. [23] on this basis could not solve this type of problem better. Subsequently, Xu et al. [24] constructed a reverse constrained mixed-frequency data sampling model (RR-MIDAS), which can realize the real-time prediction of low-frequency data to high-frequency data and adapt to the situation of large frequency multiplicity difference. The RR-MIDAS containing K low-frequency explanatory variables is defined as follows:

$$\begin{aligned}
\mathbf{y}_{t+h/m} &= \boldsymbol{\beta}_0 + \sum_{a=1}^{K} \boldsymbol{\beta}_{a,h} \sum_{b=0}^{l_a} \boldsymbol{\rho}_a(\alpha_{h_a}; b)\boldsymbol{x}_{a,t-b} + \boldsymbol{u}_{0,h} \\
&= \boldsymbol{\beta}_0 + \boldsymbol{\beta}_{1,h}\left[\boldsymbol{\rho}_1(\alpha_{h_1}; 0)\boldsymbol{x}_{1,t-0} + \boldsymbol{\rho}_1(\alpha_{h_1}; 1)\boldsymbol{x}_{1,t-1} + \ldots + \boldsymbol{\rho}_1(\alpha_{h_1}; l_1)\boldsymbol{x}_{1,t-l_1}\right] + \\
&\boldsymbol{\beta}_{2,h}\left[\boldsymbol{\rho}_2(\alpha_{h_2}; 0)\boldsymbol{x}_{2,t-0} + \boldsymbol{\rho}_2(\alpha_{h_2}; 1)\boldsymbol{x}_{2,t-1} + \ldots + \boldsymbol{\rho}_2(\alpha_{h_2}; l_2)\boldsymbol{x}_{2,t-l_2}\right] + \ldots\ldots + \\
&\boldsymbol{\beta}_{K,h}\left[\boldsymbol{\rho}_K(\alpha_{h_K}; 0)\boldsymbol{x}_{K,t-0} + \boldsymbol{\rho}_K(\alpha_{h_K}; 1)\boldsymbol{x}_{K,t-1} + \ldots + \boldsymbol{\rho}_K(\alpha_{h_K}; l_K)\boldsymbol{x}_{K,t-l_a}\right] + \boldsymbol{u}_{0,h} \\
&= \boldsymbol{\beta}_0 + (\boldsymbol{\beta}_{1,h}, \boldsymbol{\beta}_{2,h}, \ldots\ldots \boldsymbol{\beta}_{K,h})(\widetilde{\boldsymbol{x}}_{1,t}\widetilde{\boldsymbol{x}}_{2,t} \ldots\ldots \widetilde{\boldsymbol{x}}_{K,t})^T + \boldsymbol{u}_{0,h} \\
&= \boldsymbol{\beta}_0 + \widetilde{B}_h\widetilde{X}_t^T + \boldsymbol{u}_{0,h}
\end{aligned} \quad , \tag{1}$$

where h is the number of sampling steps, $h = 1, 2, 3, 4 \ldots$; m is the frequency multiplicity difference between the explanatory variable x and the response variable y. In this paper, the frequency multiplicity difference between the monthly variable and the daily variable is $m = 20$; $\boldsymbol{\rho}_a(\alpha_{h_a}; b)$ is the weight constraint term; $\boldsymbol{\rho}_a(\alpha_{h_a}; b)\mathbf{x}_{a,t-b}$ is the lag polynomial of the variable x_a when the sampling step is h; b is the number of forward steps of the variable x_a when the sampling step is h, $b = 0, 1, 2, \ldots, l_a$; l_a is the maximum lag order of the variable x_a; β_0 is the constant term; $\beta_{a,h}$ is the regression parameter of the variable x_a when the sampling step is h; $u_{0,h}$ is the random error.

In order to effectively improve some defects of RU-MIDAS, the RR-MIDAS model introduces a weight constraint function to reduce the number of parameters. In previous studies, Breitung and Roling [26] mentioned that the weight constraint functions are Almon and Beta functions, in which the Almon function is commonly used to fit time-series data with nonlinear trends, especially in economics and statistics, and it is a very flexible nonlinear function.The Almon function is defined as follows:

$$\rho_a(\alpha_{h_a}; b) = \exp\left(\alpha_1 + \alpha_2 b^2\right) / \sum_{n=0}^{l_a} \exp\left(\alpha_1 + \alpha_2 n^2\right). \tag{2}$$

Mishra et al. [27] used the MIDAS–Almon weighting method to process the mixed frequency data to predict the trend of GDP. The Almon function is a smoothing technique based on the construction of polynomials, which efficiently smooths the data and helps to minimize the effect of noise or seasonal fluctuations. As it can adapt to a variety of trend shapes, by adjusting the parameters it can more accurately match the shape of the

data, and the function also has asymptotic zero error characteristics. Therefore, the Almon function can better meet the needs of this paper to predict the stock price volatility. In this paper, the RR-MIDAS model is chosen and the flexible Almon constraint function is applied to the lag term of low-frequency variables to reduce the number of parameters and solve the problem of too many parameters to be estimated in the RU-MIDAS model. To simplify the operation, this paper refers to the treatment of Xu et al. [28] and fixes $\alpha_1 = 1$ for the two parameters α_1 and α_2 of the Almon function.

2.2. CNN-LSTM Neural Network

Li et al. [29] proposed that CNNs can not only extract image features but also extract the relationship between multidimensional time-series data in spatial structure. CNN is mainly composed of a convolutional layer, pooling layer, and fully connected layer, and its structure is shown in Figure 1.

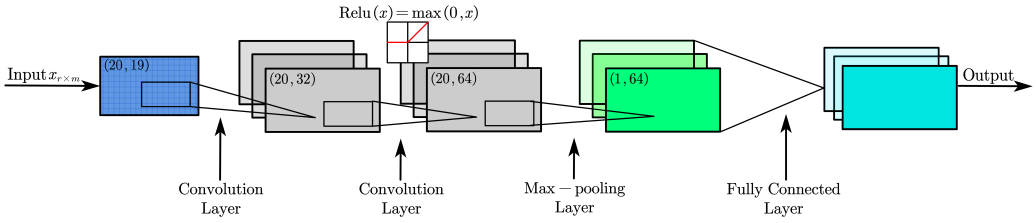

Figure 1. Structure of CNN, illustrating the flow from input through convolutional layers to the output. Note: In this figure, the symbols (X, Y) denote the configuration of the convolutional layers in the neural network. The first number X indicates the dimension of the output sequence after convolution. The second number Y indicates the number of filters or channels in the layer.

Common CNNs are one-dimensional convolution, two-dimensional convolution, and three-dimensional convolution. The data analyzed in this paper are time series related to stock price fluctuations and the variables are coupled. One-dimensional convolution is commonly used in time-series analysis; therefore, this paper adopts one-dimensional convolution kernels (1-DCNN) to extract the time-series features, which can speed up the training speed and improve the generalization performance at the same time.The process of the CNN convolution operation is as follows:

$$C_b^l = f\left(\sum C_a^{l-1} * W_{ab}^l + B_b^l\right) \tag{3}$$

$$MP_b^l = \text{Maxpooling}\left(C_b^l\right) \tag{4}$$

$$F_b^l = \text{Flatten}\left(MP_b^l\right) \tag{5}$$

$$M_b^{l+1} = \text{FC}\left(F_a^l \cdot W_{ab}^l + B_b^{l+1}\right), \tag{6}$$

where C_a^{l-1} and C_b^l are the inputs and outputs of the convolutional layer, W_{ab}^l and B_b^l denote the convolutional kernel and bias term of the convolutional layer, respectively, $*$ denotes the convolutional operation, f is the activation function, Maxpooling is the pooling layer, Flatten is the spreading layer, and FC is the fully connected layer.

Although CNNs can effectively extract the spatial features of multivariate data, they ignore the information in the time dimension. Hidden variable models have the problem of long-term information preservation and short-term input missing. Take the classical RNN model as an example; it is often difficult for it to effectively capture long-term dependent information because of the gradient vanishing problem when dealing with sequence data. Furthermore, LSTM is good at extracting temporal information with long-term memory capability, which is one of the methods to solve the problem. In this paper, the input data are extracted by a one-dimensional convolution operation, and then the activation function,

maximum pooling layer, spreading layer, and fully connected layer are input successively, and then the extracted temporal and spatial features are input into the LSTM. The structure of the LSTM is shown in Figure 2.

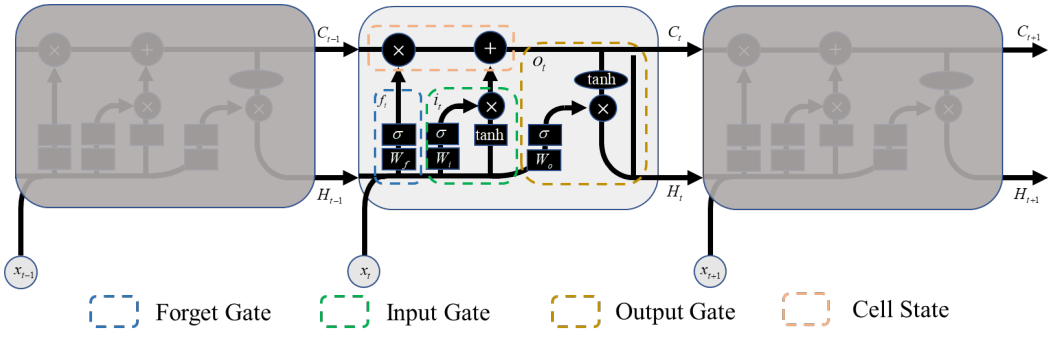

Figure 2. LSTM network architecture, detailing the components responsible for capturing long-term dependencies.

In the CNN-LSTM model, the data processed by steps (3) to (6) or other unprocessed time-series data can be passed to LSTM to extract the temporal features. RNN contains the hidden state H_t, and LSTM is optimized on the basis of RNN with the addition of the unit state C_t and three gate structures, which are the forget gate, the input gate, and the output gate.

The unit state C_t is the key for LSTM to be able to stand out from RNN. The C_t enables the long-term memory of the neuron, and the forget gate and input gate are able to correct the long-term memory.

The forget gate manages and determines which existing information should be retained or forgotten, which helps to keep the state of the unit up-to-date and ensures that only relevant information is stored. The forget gate is denoted as f_t:

$$f_t = \sigma\Big(W_f \cdot [H_{t-1}, X_t] + B_f\Big), \tag{7}$$

where H_{t-1} is the output value of the previous neuron, X_t is the latest input value of the current neuron, W_f is the weight parameter, and B_f is the bias variable. When f_t is 1 and i_t is 0, the memory units C_{t-1} of the previous layer are all retained to the current layer. This design alleviates the gradient vanishing problem and is more capable of extracting long-term dependencies in time-series data.

In LSTM, input gates are used to evaluate and select new information delivered to the network, which is responsible for regulating which parts of the current input are critical and determining what should be retained in the cell state:

$$i_t = \sigma(W_i \cdot [H_{t-1}, X_t] + B_i), \tag{8}$$

where i_t is the input gate, W_i is the weight parameter, and B_i is the bias variable.

The new cell state is determined by multiplying the old cell state with the output of the oblivion gate and adding it to the product of the candidate state and the input gate, and the new cell state C_t is:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tanh(W_c \cdot [H_{t-1}, X_t] + B_c), \tag{9}$$

where $\tanh(\cdot)$ is the nonlinear activation function, W_c is the weight parameter, and B_c is the bias variable.

The output gate decides which information from the final unit state is conducted through the network and converts the filtered short-term memory into input for the next

step. The output gate is multiplied with the activation function to obtain the new hidden layer H_t:

$$H_t = o_t \cdot \tanh(C_t). \tag{10}$$

The output gate o_t updates the hidden state with a value that can determine the amount of information passed from the memory unit to the prediction part. Therefore, the output result is not only affected by the new data, but also by the hidden state output from the previous layer of the structure. The formula for the output gate is as follows:

$$o_t = f(W_o \cdot [H_{t-1}, X_t] + B_o), \tag{11}$$

where W_o is the weight parameter, H_{t-1} denotes the hidden state of the previous layer, B_o is the bias variable, and X_t denotes the input new data.

In this paper, the output of the LSTM is fed into the fully connected layer to obtain the feature values; specifically, the hidden states of the LSTM are mapped to the predicted values by means of the fully connected layer in the CNN-LSTM model. The fully connected layer is a common neural network layer, where each neuron is connected to all neurons in the previous layer and each connection has a different weight; the output of this layer is obtained by multiplying and summing the output of the previous layer with the corresponding weights and adding a bias term. The calculation of the fully connected layer is shown in the following equation:

$$\hat{X}_{t+1} = FC(H_t) = W_{FC} \cdot H_t + B_{FC}, \tag{12}$$

where $\hat{X}_{t+1}$ denotes the predicted value, W_{FC} denotes the weight matrix of the fully connected layer, and B_{FC} denotes the bias term of the fully connected layer.

2.3. CNN-LSTM Model Incorporating Macroeconomic Variables

Considering that low-frequency macro-variables have an impact on realized volatility, this paper introduces the inclusion of low-frequency macro-variables on the basis of technical indicators and fundamental market indicators data to form a mixed-frequency dataset for predicting realized volatility. In order to deal with the reverse mixing problem with large frequency multiplicity differences, the reverse constrained mixed-frequency sampling method (RR-MIDAS) is utilized to process the mixed-frequency data.

In this paper, the RM-CNN-LSTM-P model is established to solve the mixed-frequency data problem and the feature-learning problem of multivariate time-series data, where RM- denotes the RR-MIDAS model, and -P denotes the parallel LSTM. On the one hand, the RR-MIDAS model preprocesses the mixed-frequency information, which is then fed into a neural network for feature extraction. Specifically, the CNN component is utilized to extract spatial features related to volatility fluctuations, and the LSTM network captures the temporal dynamics of the data series. This dual learning process can understand the input data more comprehensively; on the other hand, this paper parallelizes the LSTM structure in the model to deal with volatility historical data specifically, which helps to capture the nonlinear time-varying features in the volatility data and improves the model's comprehension of the volatility historical information. Finally, in this paper, the computing results of the above two are fused with features to synthesize the spatial and temporal information to obtain a more comprehensive and accurate volatility prediction value.

The framework of this paper is shown in Figure 3. The RM-CNN-LSTM-P model is able to make full use of spatial and temporal information when processing inverse mixing and multivariate time-series data and improve the learning ability of complex data patterns. At the same time, the model also takes advantage of the computational efficiency of one-dimensional convolution and the processing of long sequences by identifying the local features of longer sequences through CNN, passing their outputs to LSTM for processing, and automatically optimizing the model hyperparameters using the Optuna tuning framework, which improves the model's learning ability and prediction accuracy.

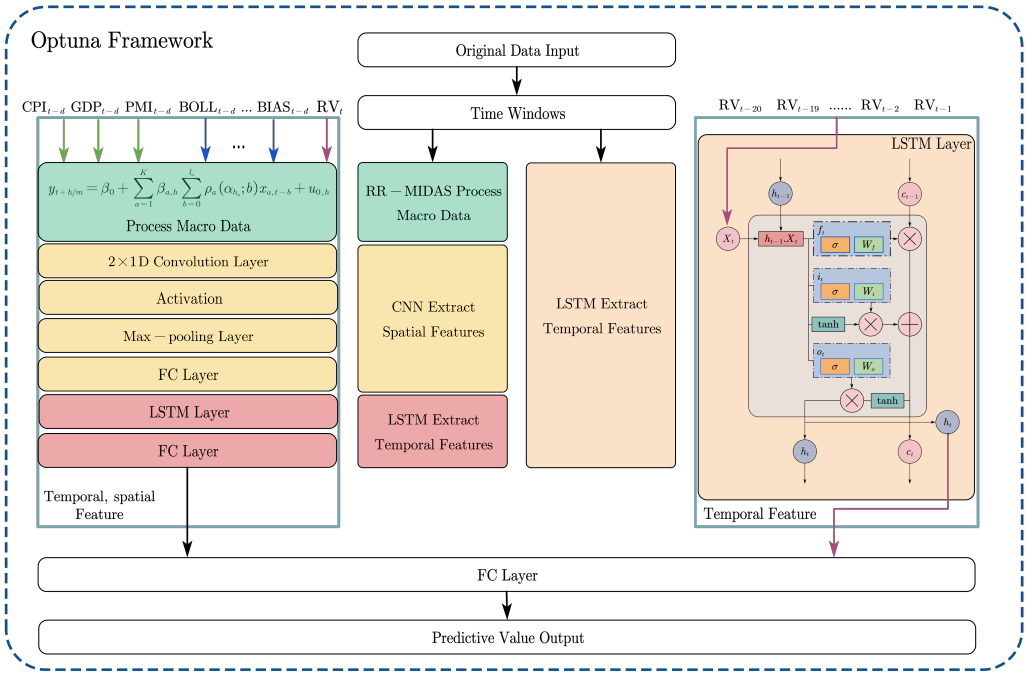

Figure 3. Research framework overview, showcasing the integration of mixed-frequency data and deep-learning models for volatility forecasting.

2.4. Evaluation Criteria

2.4.1. Loss Function Criteria

In order to evaluate the performance of different models, the square root mean square error (RMSE), mean absolute error (MAE), mean square logarithmic error (MSLE), mean absolute percentage error (MAPE), symmetric mean absolute percentage error (SMAPE), and Quasi-Like (QLIKE) loss function are used as the evaluation criteria in this paper. The formula for each indicator is as follows:

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{N}\left(RV_t - \hat{RV}_t\right)^2}{N}} \tag{13}$$

$$\text{MAE} = \frac{\sum_{t=1}^{N}\left|RV_t - \hat{RV}_t\right|}{N} \tag{14}$$

$$\text{MSLE} = \frac{\sum_{t=1}^{N}\left(\ln(1+RV_t) - \ln\left(1+\hat{RV}_t\right)\right)^2}{N} \tag{15}$$

$$\text{MAPE} = \frac{\sum_{t=1}^{N}\left|\frac{RV_t - \hat{RV}_t}{RV_t}\right|}{N} \tag{16}$$

$$\text{SMAPE} = \frac{\sum_{t=1}^{N}\frac{2\times\left|RV_t - \hat{RV}_t\right|}{|RV_t| + \left|\hat{RV}_t\right|}}{N} \tag{17}$$

$$\text{QLIKE} = \frac{\sum_{t=1}^{N}\left(\ln \hat{RV}_t^2 + RV_t^2/\hat{RV}_t^2\right)}{N}, \tag{18}$$

where N is the number of samples, RV_t represents the true value, and $\hat{RV}_t$ is the corresponding predicted value. The smaller each evaluation index is, the better the performance of the model.

2.4.2. Diebold–Mariano Test

The DM test [30] is a statistical test for comparing the performance of two time-series forecasting models.The main purpose of the DM test is to assess whether there is a significant difference in the forecasting performance of one model with respect to the other.The null hypothesis of the test is that the forecasting performance of the two models is close to each other, which can be denoted as $H_0 : E_A = E_B$, while the alternative hypothesis is that their performance is significantly different, which can be denoted as $H_1 : E_A \neq E_B$. Assuming that two prediction models, A and B, perform a prediction task with a time span of T, the prediction results in that time range are obtained, and the prediction errors $E_A = [|x_{a1}|, |x_{a2}|, \ldots, |x_{aT}|]$ and $E_B = [|x_{b1}|, |x_{b2}|, \ldots, |x_{bT}|]$ are computed based on the true values, where x_{a1} and x_{b1} are the differences between the predicted values and the true values of model A and model B, respectively.

The DM statistic values are as follows:

$$\mathrm{DM} = \frac{D_{\mathrm{mean}}}{D_{\mathrm{std}}}, \tag{19}$$

where $D = [d_1, d_2, \cdots, d_T] = [x_{a1} - x_{b1}, x_{a2} - x_{b2}, \cdots, x_{aT} - x_{bT}]$, $D_{\mathrm{mean}} = \frac{\sum_{t=1}^{T} d_t}{T}$, $D_{\mathrm{std}} = \sqrt{\frac{\sum_{t=1}^{T} (d_t - D_{\mathrm{mean}})^2}{T-1}}$.

The distribution of the DM statistic under the null hypothesis obeys the standard normal distribution Z. There is a significant difference between the predictive performance of Model A and Model B when $|\mathrm{DM}| > Z_{1-\alpha}$.

3. Empirical Research

In order to improve the prediction accuracy of volatility, this paper combines the realized volatility calculated based on 5-min high-frequency financial data and its influencing factors to construct a new volatility prediction index system. In addition, the prediction accuracy of the RM-CNN-LSTM-P model is compared with that of the traditional econometric model, the machine-learning model, and the deep-learning model with and without the introduction of macro-variables, respectively.

3.1. Data Sources and Description

Building on the work of Lei et al. [10] and Song et al. [31], this study selects the following variables as influential factors for volatility prediction. Gross domestic product (GDP), consumer price index (CPI), and purchasing manager's index (PMI) are selected to construct the macro-factor dataset, and the frequencies and interpretations of the target variables and each explanatory variable are shown in Table 1.

CPI, GDP, and PMI data were obtained from Sina Finance (https://finance.sina.com.cn/, accessed on 13 May 2024). RV, BIAS, DMA, CDP, AR, BR, RV_V, and CR were calculated from the trading volume, overnight spreads, turnover rate, and other data related to the SSE index, and the basic data related to the calculations were obtained from the Wind database. SSE BOLL, MACD, VMACD, MA, RSI, KDJ, GL, VL, and OI were obtained from the Wind database.

The datasets of the above macro-factors are low-frequency variables, in which GDP is season data and the remaining two macro-indicators are monthly data, and this paper uses linear interpolation to interpolate the quarterly data to obtain monthly GDP data. All data were taken from 1 March 2011 to 31 October 2022, a total of 2832 trading days. In order to maintain the consistency of the frequency multiplicity difference between low-frequency data and high-frequency data, this paper treats the monthly trading days of the high-frequency data as 20 days and randomly removes the excess trading day data in the months that are more than 20 days; the trading days that are less than 20 days are supplemented by the linear interpolation to 20 days, and the final trading days are 2800 days.

Table 1. Frequency of variables and specific explanations. This table outlines the frequency of the target variable and influencing variables, including macroeconomic indicators such as GDP, CPI, and PMI, along with their respective interpretations.

Variable Type		Variable	Frequency	Explanation
Target Variable		RV	Day	The realized volatility of high-frequency prices
Influencing Variables	Macro Variables	CPI	Month	Consumer Price Index
		GDP	Season	Gross Domestic Product
		PMI	Month	Purchasing Managers' Index
	Micro Variables	BOLL	Day	Bollinger Bands
		MACD	Day	Moving average convergence
		VMACD	Day	Volume moving average convergence divergence
		MA	Day	Moving average
		RSI	Day	Relative Strength Indicator
		KDJ	Day	Stochastic Index
		OI	Day	Open interest volume
		GL	Day	Rise and fall range
		VL	Day	Trading Volume
		RV_V	Day	The realized volatility of the trading volume
		CR	Day	Center of Gravity Indicator
		DMA	Day	Parallel line difference indicator reflecting the energy of buying and selling forces and future price trend
		CDP	Day	Contrarian operation indicator
		AR	Day	Market buying and selling sentiment
		BR	Day	The degree of market buying and selling desire
		BIAS	Day	The degree of deviation from the moving average

Note: Some of the indicators in the table are calculated as follows: CR: The sum of the buyer's power since the Nth day − The sum of the seller's power since the Nth day. DMA: 5-day moving average − 10-day moving average. CDP: (Previous day's highest price + previous day's lowest price + previous day's closing price^2)/4. AR: (Closing price − opening price)/(opening price − the lowest price). BR: (The highest price − closing price)/(closing price − the lowest price). BIAS: (Closing price of the day − 5-day average price)/5-day average price.

To perform linear interpolation, consider two known points, $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$, where $x_1 < x_2$, and we want to find the value of function $f(x)$ at a point x that lies between x_1 and x_2. The linear interpolation formula is given by:

$$f(x) = y_1 + \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x - x_1). \tag{20}$$

In this paper, we calculate the daily realized volatility based on the methodology in Andersen et al. [7], which is the logarithm of the daily adjacent 5-min closing price $x_{t,d}$ to compute the sum of the returns $r_{t,d}$ squared, which is defined as follows:

$$RV_t = \sum_{d=1}^{48} [(\ln x_{t,d} - \ln x_{t,d-1}) \cdot 100]^2, \tag{21}$$

where t represents the trading day and $x_{t,d}$ is the closing price for every five minutes of trading day t, $d = 1, 2, 3, \ldots, 48$.

Figure 4 illustrates the Spearman correlation coefficients between the indicators of forecasting and between the variables with realized volatility.

As can be seen in Figure 4, the correlation between RV and BIAS, DMA, VL, and OI is high, and there is also some correlation with CPI, GDP, and PMI. The covariance test between the indicators shows low correlation, which indicates that introducing these variables simultaneously into the model does not lead to serious multicollinearity problems, especially with the macro-variables, and the correlation is low (the absolute value of which is between 0 and 0.2); there is a certain degree of correlation between the indicators and realized volatility, and some of the indicators have a correlation of more than 0.3 (e.g., OI, BIAS); in particular, the correlation between the macroeconomic variables and the

realized volatility is more prominent. Therefore, when constructing the forecasting model the introduction of these variables may improve the forecasting accuracy of the model.

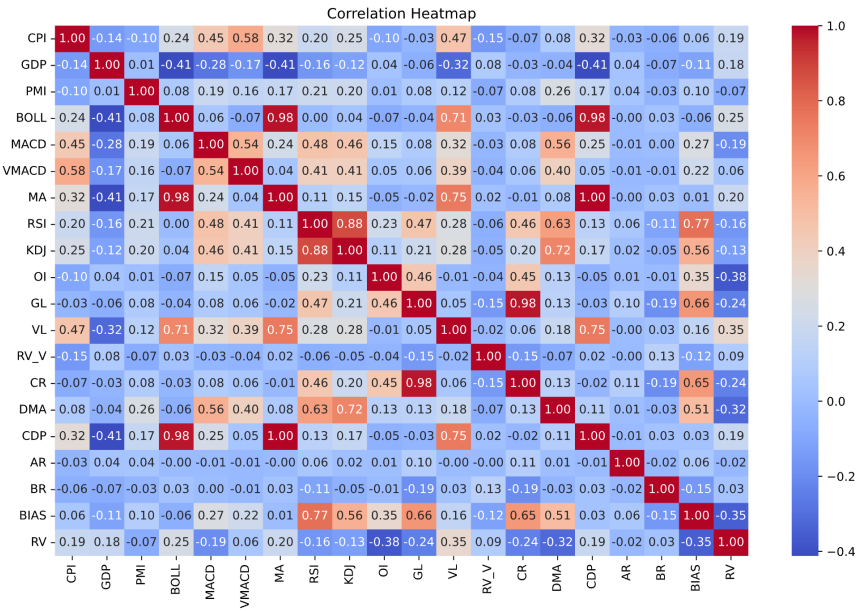

Figure 4. Spearman correlation coefficients analysis, depicting the relationships between various indicators and realized volatility.

In this paper, the target variable is realized volatility (RV), and the explanatory variables include micro variables (basic quotes, technical indicators) and macro-variables. When training the model, this paper splits the dataset into a training set (from 1 March 2011 to 31 December 2020) and a test set (from 1 January 2021 to 31 October 2022). In addition, in order to avoid model training overfitting, the latter 20% of the training set is selected as the validation set in this paper. In addition, in order to eliminate the interference of the quantiles on model training, this paper standardizes all variables and back-standardizes the predicted values of the model. The standardization calculation formula is as follows:

$$x_{\text{normalized}} = \frac{x - \text{mean}(x)}{\text{std}(x)} \tag{22}$$

$$y_{\text{normalized}} = \frac{y - \text{mean}(y)}{\text{std}(y)}. \tag{23}$$

In the above equation, x and y are eigenvectors, and $\text{mean}(\cdot)$ and $\text{std}(\cdot)$ represent the mean and standard deviation of the corresponding variables, respectively. This paper scrolls through the past 20 days of historical data to predict the next day's realized volatility, as shown in Figure 5.

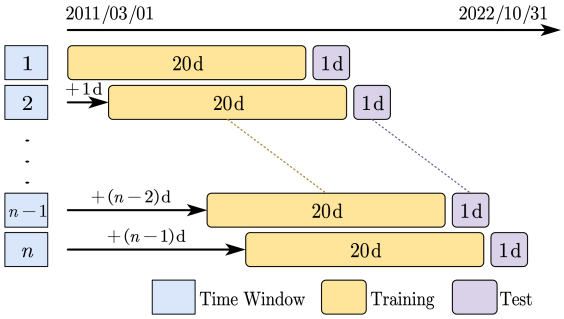

Figure 5. Historical data forecasting approach, demonstrating the use of rolling time windows for predicting future realized volatility.

3.2. Optuna Framework

The RM-CNN-LSTM-P model for predicting the realized volatility of stock prices proposed in this paper contains multiple nonlinear hyperparameters, and the tuning of these nonlinear parameters is difficult; the grid search method and stochastic search are a way to find better hyperparameters, but they have the disadvantages of being more time-consuming and consuming more resources.

To tune the nonlinear hyperparameters in the model, this paper introduces the widely used Optuna tuning framework, which improves the predictive ability of the model by automatically tuning the appropriate hyperparameters. The framework, developed by Akiba et al. [32], has two basic concepts: study, an optimization process based on an objective function; and trial, a single execution process of the objective function. The tuning framework performs pruning operations on poorly performing trials to improve efficiency while selecting the best hyperparameters for the model to achieve optimal results. In the tuning process of the Optuna framework, this paper chooses the tree-structured Parzen estimator [33] algorithm as a sampler to find the optimal hyperparameter search space.

There are 67 hyperparameters in the model of this paper: first, the hyperparameters in RR-MIDAS include the maximum lag orders l_1, l_2, l_3 of the three low-frequency variables CPI, GDP, and PMI, and their corresponding Almon coefficients α_{h1}, α_{h2}, α_{h3}, where $h_i = 1, 2, 3, \ldots, m$ (frequency multiplicity $m = 20$ in this paper). Thus there are 63 hyperparameters in total. Secondly, there are two hyperparameters in the neural network part of CNN and LSTM, including the number of units in the serial LSTM module with CNN *unit*1 and the number of units in the parallel LSTM module with CNN *unit*2. Finally, the model training part consists of two hyperparameters: the batch size, *batch_size*, and the learning rate of Adam's optimizer, *learning_rate*.

In the Optuna tuning process, each trial tries to minimize the loss metric and calculate various error metric values. In this paper, we set the number of experiments, $trial = 500$, and the return value is the loss MAE value of the model on the validation set. The hyperparameter optimization process of the RM-CNN-LSTM-P model is shown in Figure 6, where the horizontal coordinate represents the number of experiments, the vertical coordinate represents the MAE value, the blue dots represent the MAE at the end of one experiment, and the red line represents the minimum loss value up to each experiment before the experiment is carried out up to 500 times. Table 2 demonstrates the hyperparameters of the RM-CNN-LSTM-P model after Optuna framework tuning.

Table 2. Optimal hyperparameter of the RM-CNN-LSTM-P model. The table presents the optimal hyperparameters determined through the Optuna framework for the RR-MIDAS and CNN-LSTM components of the model.

Part	Hyperparameter	Optimal Hyperparameter Value
RR-MIDAS	l_1, l_2, l_3	2, 2, 2
	α_{h1}	[0.763, 1.054, 0.970, 0.311, 0.800, 0.292, 0.493, 0.564, 0.295, 0.799, 0.802, 0.781, 1.150, 1.289, 0.691, 0.920, 1.455, 0.908, 0.384, 1.005]
	α_{h2}	[0.734, 0.159, 1.399, 0.697, 0.558, 1.137, 0.577, 0.767, 0.864, 0.245, 0.330, 1.171, 0.796, 0.527, 1.205, 0.988, 1.388, 0.288, 1.092, 0.820]
	α_{h3}	[1.285, 0.884, 0.474, 0.824, 0.183, 0.444, 0.596, 0.707, 0.901, 1.086, 0.147, 1.010, 0.737, 0.542, 1.069, 0.373, 1.227, 0.709, 1.020, 1.466]
CNN-LSTM	*unit*1	170
	*unit*2	99
Model	*batch_size*	101
	learning_rate	0.160

Note: The RM-CNN-LSTM-P model features a CNN with two convolutional layers using 32 and 64 filters. This is followed by two LSTM layers with a variable number of units (parameterized as *unit*1) and a time step of 20. Additionally, there is a parallel LSTM component with one LSTM layer having a different variable number of units (*unit*2).

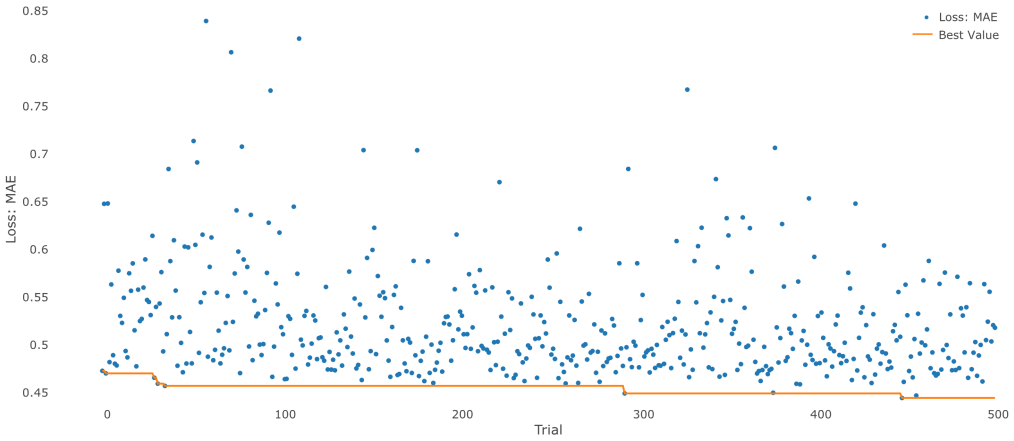

Figure 6. Optuna running process, showing the dynamic optimization process of the Optuna framework during the hyperparameter tuning phase of the RM-CNN-LSTM-P model.

3.3. Prediction Results

In order to examine the prediction performance of the proposed models in this paper, in this section the RM-CNN-LSTM-P model is compared with the deep-learning models RM-CNN, RM-LSTM, RM-CNN-GRU, and RM-CNN-LSTM with the introduction of macroscopic data, also with the CNN, LSTM, CNN-GRU, and CNN-LSTM model without the introduction of macroscopic data. The machine-learning models RM-SVR and RM-RF with the introduction of macro-data, the deep-learning models SVR and RF without the introduction of macro-data, as well as the classical econometric models HAR, RM-RIDGE, RM-LINEAR, RIDGE, and LINEAR are also compared. The results are shown in Table 3.

Table 3. Analysis of model prediction performance based on multiple error metrics. This table compares and ranks the predictive performance of different models using different loss functions.

Model	RMSE	MAE	MSLE	MAPE	SMAPE	QLIKE	Rank
RM-CNN-LSTM-P	0.833 (1)	0.445 (1)	0.091 (1)	0.536 (2)	0.471 (1)	3.627 (8)	1
RM-LSTM	0.893 (2)	0.535 (6)	0.108 (2)	0.779 (6)	0.563 (7)	2.524 (6)	2
LSTM	0.917 (3)	0.536 (7)	0.111 (3)	0.832 (7)	0.559 (6)	2.510 (5)	3
RM-CNN-LSTM	1.016 (8)	0.508 (2)	0.128 (6)	0.512 (1)	0.519 (2)	5.162 (13)	4
RM-CNN-GRU	1.000 (5)	0.520 (4)	0.124 (4)	0.644 (5)	0.543 (4)	4.014 (10)	4
CNN-GRU	1.010 (7)	0.517 (3)	0.127 (5)	0.616 (4)	0.537 (3)	4.522 (11)	5
CNN-LSTM	1.019 (9)	0.523 (5)	0.130 (7)	0.615 (3)	0.546 (5)	4.736 (12)	6
RM-CNN	0.997 (4)	0.602 (8)	0.142 (9)	0.905 (8)	0.711 (9)	42.410 (15)	7
RM-SVR	1.136 (10)	0.822 (11)	0.211 (11)	1.734 (11)	0.732 (11)	1.218 (2)	8
SVR	1.181 (11)	0.839 (12)	0.223 (12)	1.755 (12)	0.723 (10)	1.719 (4)	9
CNN	1.005 (6)	0.624 (9)	0.157 (10)	0.954 (9)	0.733 (12)	274.505 (18)	10
HAR	2.314 (17)	0.706 (10)	0.141 (8)	1.123 (10)	0.632 (8)	5.268 (14)	11
RM-RF	1.641 (12)	1.282 (14)	0.405 (13)	2.896 (14)	0.871 (14)	1.216 (1)	12
RF	1.657 (13)	1.280 (13)	0.407 (14)	2.894 (13)	0.862 (13)	1.268 (3)	13
RM-RIDGE	2.176 (15)	1.680 (16)	0.482 (17)	3.176 (16)	1.113 (16)	2.792 (7)	14
RIDGE	2.064 (14)	1.555 (15)	0.441 (15)	2.904 (15)	1.088 (15)	130.453 (17)	15
RM-LINEAR	2.438 (18)	1.889 (18)	0.553 (18)	3.705 (18)	1.175 (17)	3.877 (9)	16
LINEAR	2.313 (16)	1.753 (17)	0.481 (16)	3.278 (17)	1.185 (18)	128.524 (16)	17

Note: "RM-" is the RR-MIDAS model, which indicates that macro-variables are treated by RR-MIDAS as explanatory variables and integrated with the rest of the high-frequency variables to be input into the subsequent model. The values in brackets indicate the ranking of each model based on the respective error metric. Rank is the result of the summation of the rankings of each indicator and then sorted.

For all models, the prediction performance of the models improves after adding macroeconomic variables to the explanatory variables, which initially shows the importance of macro-variables for prediction; the prediction performance of the deep-learning model

is better than that of the machine-learning model, and the prediction performance of the machine-learning model is better than that of the ridge model and the linear model; among them, the RM-CNN-LSTM-P model proposed in this paper has the best prediction performance among all the models.

First, the RM-CNN-LSTM-P model reduces RMSE, MAE, MSLE, SMAPE, and QLIKE by 18.01%, 12.40%, 28.91%, 9.25%, and 29.74%, respectively, compared with the RM-CNN-LSTM model, which indicates that the parallel LSTM approach proposed in this paper has a better prediction performance than the one that directly inputs the RV together with other explanatory variables into the model. The parallel LSTM can effectively extract temporal features, which in turn improves the prediction performance of the model.

Secondly, the optimal RM-CNN-LSTM-P among deep-learning models reduces RMSE, MAE, MSLE, MAPE, and SMAPE by 26.67%, 45.86%, 56.87%, 69.09%, and 35.66%, respectively, compared with the optimal RM-SVR among machine-learning models, and comparing the best predictive performance of RM-SVR among machine-learning models RMSE by 50.91%, QLIKE by 76.87%, and RM-SVR model by higher ranking compared to the HAR model, which has the best predictive performance among traditional metrics models. SVR fits the data by finding the optimal hyperplane, and it may not perform well for datasets with more features or higher complexity. In contrast, deep-learning models can better handle complex data structures and features, and therefore perform better in this comparison. Additionally, traditional statistical models are usually based on a number of assumptions and simplifications and may not adapt well to complex data structures.

In addition, the QLIKE error is particularly sensitive to the deviations of individual forecasts. Unlike other error metrics that are based on linear responses to deviations between predicted $\hat{RV}_t$ and actual realized volatility values, RV_t, QLIKE incorporates a logarithmic transformation that can amplify the impact of large errors, especially underestimations.

The term $\ln \hat{RV}_t^2$ penalizes underestimates more than overestimates, as the logarithm of a small number becomes increasingly negative. Also, the term $RV_t^2/\hat{RV}_t^2$ can become very large if $\hat{RV}_t$ is significantly underpredicted, leading to a significant increase in the QLIKE value. This asymmetric treatment of errors means that the QLIKE metric places a higher cost on underprediction.

3.4. Result Explanations

Using RM-LINEAR as the benchmark model, the percentage change of other model metrics is shown in Table 4.

Table 4. Comparison of the predictive performance of the models with RM-LINEAR. The table illustrates the percentage change in predictive performance metrics when comparing each model to the RM-LINEAR benchmark.

Model	RMSE	MAE	MSLE	MAPE	SMAPE	QLIKE
RM-CNN-LSTM-P	−65.83%	−76.44%	−83.54%	−85.53%	−59.91%	−6.45%
RM-LSTM	−63.37%	−71.68%	−80.47%	−78.97%	−52.09%	−34.90%
RM-CNN-LSTM	−58.33%	−73.12%	−76.85%	−86.18%	−55.83%	33.14%
RM-CNN-GRU	−58.98%	−72.47%	−77.58%	−82.62%	−53.79%	3.53%
RM-CNN	−59.11%	−68.13%	−74.32%	−75.57%	−39.49%	993.89%
RM-SVR	−53.40%	−56.48%	−61.84%	−53.20%	−37.70%	−68.58%
RM-RF	−32.69%	−32.13%	−26.76%	−21.84%	−25.87%	−68.64%
RM-RIDGE	−10.75%	−11.06%	−12.84%	−14.28%	−5.28%	−27.99%

Except for individual values, the RMSE, MAE, MSLE, MAPE, SMAPE, and QLIKE metrics of the predicted values of each model decreased, and the model prediction accuracies were improved to different degrees, among which RM-CNN-LSTM-P is the optimal deep-learning model, the deep-learning model is better than the machine-learning model, and the econometric model is the worst.

Taking RM-CNN-LSTM-P as an example, all of its indexes have a significant decrease compared with the benchmark model, especially on MAPE, which reaches 85.53%, which means that the model has a great improvement in prediction accuracy compared with the base model. This indicates that LSTM is particularly suitable for processing and predicting long-term dependency problems in time-series data.

CNN-GRU combines the use of convolutional neural networks and GRU. CNN is good at extracting spatial features in time-series data, while GRU is a simplified version of LSTM with fewer parameters and lower computational complexity, but its performance is slightly weaker than that of LSTM in the complex scenarios in this paper.

CNN is a deep-learning model commonly used to process data with an obvious spatial structure, which uses convolutional layers to automatically and efficiently capture spatial hierarchical features of the data, but it is not able to capture temporal features better, so its prediction performance is not good when used independently.

Among the machine-learning models, for RM-SVR and RM-RF, while also showing improvements over the benchmark models, the improvements are much smaller compared to the deep-learning models. Compared to SVR, CNNs typically have better performance when dealing with large datasets because they can capture deeper features, which is difficult to do with traditional machine-learning methods. For example, RM-SVR shows an improvement of 53.20% on MAPE, while the deep-learning model shows at least 75.57% improvement, which shows the advantage of deep learning in dealing with complex nonlinear problems.

The econometric model RM-RIDGE shows the smallest improvement, with only a small decrease in all metrics, which may be due to the fact that econometric models are inherently linear and have a limited ability to deal with nonlinear and complex data structures, and therefore have the worst performance in comparison with the other models.

The main reason why the deep-learning model outperforms the machine-learning model and the econometric model is because of its stronger pattern recognition and feature extraction capabilities when dealing with nonlinear, complex, and high-dimensional data structures, which enables the deep-learning model to show more accurate predictions across a wide range of predictive metrics.

Finally, Table 5 compares the effect of whether or not macro-variables are input on the prediction accuracy of the model. Most of the data in the table are negative, which indicates that the model with macro-features has better prediction performance. RM-CNN-LSTM-P reduces the RMSE, MAE, MSLE, MAPE, SMAPE, and QLIKE by 18.25%, 14.91%, 30.00%, 12.85%, 13.74%, and 23.42%, respectively, when compared to the model without inputting macro-variables, confirming the importance of macro-features in improving model performance. Macro features provide more comprehensive information about the external environment, and these macroeconomic indicators can reflect the overall market trend and potential economic cycle changes, which allows the model to take into account more factors that may affect the prediction, thus improving the prediction accuracy and robustness of the model.

Combining the advantages of CNN and LSTM for processing sequence data and parallel LSTM for capturing temporal characteristics, together with the introduction of macro-features, the structural design of RM-CNN-LSTM-P as well as the integration of macro-features enable the model to more accurately capture the key information driving the prediction when dealing with complex data, and thus its prediction performance is significantly better than that of the other models in many performance indicators.

Table 5. Assessment of the impact of fusing RR-MIDAS on the performance of each model. Displaying the percentage change in predictive performance metrics when macroeconomic variables are included in the model.

Model	RMSE	MAE	MSLE	MAPE	SMAPE	QLIKE
RM-CNN-LSTM-P	−18.25%	−14.91%	−30.00%	−12.85%	−13.74%	−23.42%
RM-CNN-LSTM	−0.29%	−2.87%	−1.54%	−16.75%	−4.95%	9.00%
RM-LSTM	−2.62%	−0.19%	−2.70%	−6.37%	0.72%	0.56%
RM-CNN-GRU	−0.99%	0.58%	−2.36%	4.55%	1.12%	−11.23%
RM-CNN	−0.80%	−3.53%	−9.55%	−5.14%	−3.00%	−84.55%
RM-SVR	−3.81%	−2.03%	−5.38%	−1.20%	1.25%	−29.14%
RM-RF	−0.97%	0.16%	−0.49%	0.07%	1.04%	−4.10%
RM-RIDGE	5.43%	8.04%	9.30%	9.37%	2.30%	−97.86%
RM-LINEAR	5.40%	7.76%	14.97%	13.03%	−0.84%	−96.98%

3.5. The Degree of Influence of the Three Macro-Variables on the Predictive Performance of the Model

The purpose of this experiment is to remove any one of the three macro-variables in order to study the effect of a macro-variable on the model as a whole; if the predictive performance of the model does not decrease significantly after the removal of a macro-variable, it means that this part of the model has a small effect on the performance of the model, and if the predictive performance of the model decreases significantly after the removal of a macro-variable, it means that this macro-variable has an important predictive value.

In order to explore the degree of influence of the three macro-variables CPI, GDP, and PMI on the model prediction performance, this paper, based on the tuning results, fixes the optimal hyperparameters (as Table 2) to conduct the experiments; removes CPI, GDP, and PMI successively in order to test the degree of importance of the variables in the model prediction; calculates the prediction value; and calculates the six evaluation indexes proposed in Section 2.4.1. In this paper, the following formula is used to assess the importance of variables:

$$\mathrm{ER} = \frac{loss_i - loss_{none}}{loss_{none}}, \tag{24}$$

where $loss_i$ denotes the loss value with the removal of macro-variable i and $loss_{none}$ denotes the loss value when no macro-variable is removed.

Table 6 demonstrates the percentage change in the loss values of the corresponding error indicators after removing each macro variable.

Table 6. Percentage change in value of losses. The table shows the impact of removing individual macroeconomic variables on the predictive performance of the model, as measured by various loss indicators.

	CPI	GDP	PMI
RMSE	10.44%	14.93%	10.33%
MAE	11.67%	12.59%	8.86%
MSLE	20.50%	22.96%	14.81%
MAPE	6.69%	9.75%	0.96%
SMAPE	12.66%	8.70%	6.78%
QLIKE	187.39%	1.39%	21.93%

A larger percentage change in the loss values implies that the corresponding macro-variables have a more significant effect on the improvement of the model's forecasting accuracy. It is not difficult to see that the changes in the loss indicators after removing the GDP variable are generally higher than the other macro-variables, especially in RMSE and MSLE. This indicates that GDP is one of the most important factors influencing the model forecasts.

For the different loss indicators, the magnitude of change in QLIKE is particularly significant. the percentage change in QLIKE for CPI is as high as 187.39%, which is much higher than the range of change in GDP and PMI. This indicates that QLIKE is very sensitive to changes in CPI.

It can be seen that for the RM-CNN-LSTM-P model that the effects of the three macro-variables are positive for RMSE, MAE, MSLE, MAPE, SMAPE, and QLIKE. The changes in these quantitative indicators not only allow us to assess the importance of macro-variables, but also to find out the degree of fluctuation in the performance of the predictive model when different macro-variables are missing.

3.6. DM Test

In order to verify whether the RM-CNN-LSTM-P model is significantly better than the other models, this paper adopts the DM test, which is used to compare whether there is a significant difference in the prediction performance of the two prediction models.The results of the DM test are shown in Table 7.

Table 7. DM test results. This table presents the results of the DM test, indicating the statistical significance of the differences in predictive performance between the RM-CNN-LSTM-P model and other models.

	LINEAR	**RIDGE**	**RF**	**HAR**	**SVR**	**CNN**	**CNN-GRU**	**CNN-LSTM**	**LSTM**	**CNN-LSTM-P**
LINEAR	/	4.27 ***	7.85 ***	0.29	8.76 ***	9.08 ***	8.99 ***	8.89 ***	9.51 ***	9.7 ***
RIDGE	−4.27 ***	/	6.94 ***	−0.27	8.12 ***	8.53 ***	8.38 ***	8.25 ***	9.15 ***	9.42 ***
RF	−7.85 ***	−6.94 ***	/	−1.79 *	1.59	5.14 ***	10.08 ***	10.34 ***	7.12 ***	6.96 ***
HAR	−0.29	0.27	1.79 *	/	1.89 *	2.03 **	2.03 **	2.02 **	2.13 **	2.18 **
SVR	−8.76 ***	−8.12 ***	−1.59	−1.89 *	/	2.68 ***	2.41 **	2.05 **	5.57 ***	6.33 ***
CNN	−9.08 ***	−8.53 ***	−5.14 ***	−2.03 **	−2.68 ***	/	−0.08	−0.5	2.57 **	2.95 ***
CNN-GRU	−8.99 ***	−8.38 ***	−10.08 ***	−2.03 **	−2.41 **	0.08	/	−3.27 ***	2.53 **	2.97 ***
CNN-LSTM	−8.89 ***	−8.25 ***	−10.34 ***	−2.02 **	−2.05 **	0.5	3.27 ***	/	2.74 ***	3.14 ***
LSTM	−9.51 ***	−9.15 ***	−7.12 ***	−2.13 **	−5.57 ***	−2.57 **	−2.53 **	−2.74 ***	/	2.3 **
CNN-LSTM-P	−9.7 ***	−9.42 ***	−6.96 ***	−2.18 **	−6.33 ***	−2.95 ***	−2.97 ***	−3.14 ***	−2.3 **	/

Note: *** denotes $p < 0.01$, ** denotes $p < 0.05$, * denotes $p < 0.1$. Except for the HAR model, all models omit the 'RM-' labeling.

The null hypothesis of the DM test is that the two models have the same prediction performance, and if the value of the DM test statistic is negative it indicates that there is a significant difference between the prediction performance of the model and the prediction performance of a certain column model, and the larger the absolute value of the DM statistic, the larger the difference in the prediction performance of the model.

First of all, the DM statistics of the prediction results of the RM-CNN-LSTM-P model and other models are all significant at the 1% or 5% level, and the DM statistics are all negative, which indicates that the prediction results of the RM-CNN-LSTM-P model are more accurate and robust. Among them, the absolute values of the DM statistics of the RM-CNN-LSTM-P and RM-LINEAR, RM-RIDGE, RM-RF, and RM-SVR models are all larger, which indicates that the predictive ability of the RM-CNN-LSTM-P model is significantly better than the RM-LINEAR, RM-RIDGE, RM-RF, and RM-SVR models. Secondly, the DM test results show that the deep-learning model is better than the machine-learning model, and the traditional econometric model has the worst prediction accuracy, and this conclusion is robust. The results of the DM test again prove the conclusions in Table 3.

4. Discussion

4.1. Linear Interpolation

Some scholars use interpolation to adjust the frequency of variables; for example, Ding et al. [34] used linear interpolation to adjust the frequency of GDP variables. Table 8 demonstrates the advantages of the RR-MIDAS model over traditional interpolation methods in terms of prediction accuracy by comparing the prediction accuracies of the different models after processing the mixed-frequency data.The accuracy of the results of the RR-

MIDAS model is generally higher than that of the predictions derived from the use of interpolation when processing the mixed-frequency data and making the predictions. The main reason for this difference is that interpolation, especially linear interpolation, may distort the structure of the data due to the limitations of the method itself, which not only fails to accurately capture the latest dynamics of the data but may also compromise the timeliness of the predictions as a result.

The RR-MIDAS model employs a frequency alignment technique, which ensures the consistency of the data across time scales by frequency-adjusting the low-frequency explanatory variables. In addition, the model imposes additional controls on these variables by introducing polynomial weight constraints, which not only reduces the number of parameters in the model but also enhances its adaptability and flexibility.

Table 8. Analysis of model prediction performance based on multiple error metrics comparing the RR-MIDAS model with the linear interpolation model. This table compares and ranks the predictive performance of different models using different loss functions.

Model	RMSE	MAE	MSLE	MAPE	SMAPE	QLIKE	Rank
RM-CNN-LSTM-P	0.833 (1)	0.445 (1)	0.091 (1)	0.536 (3)	0.471 (1)	3.627 (6)	1
LI-CNN-LSTM-P	0.921 (4)	0.466 (2)	0.101 (2)	0.479 (1)	0.474 (2)	4.363 (9)	2
RM-LSTM	0.893 (2)	0.535 (7)	0.108 (3)	0.779 (7)	0.563 (7)	2.524 (5)	3
RM-CNN-GRU	1.000 (6)	0.520 (4)	0.124 (5)	0.644 (5)	0.543 (4)	4.014 (8)	4
RM-CNN-LSTM	1.016 (8)	0.508 (3)	0.128 (6)	0.512 (2)	0.519 (3)	5.162 (12)	5
LI-LSTM	0.901 (3)	0.563 (8)	0.111 (4)	0.860 (8)	0.579 (8)	2.341 (4)	6
LI-CNN-LSTM	1.018 (9)	0.526 (5)	0.131 (8)	0.638 (4)	0.551 (5)	4.613 (11)	7
LI-CNN-GRU	1.014 (7)	0.531 (6)	0.130 (7)	0.653 (6)	0.558 (6)	4.504 (10)	7
RM-CNN	0.997 (5)	0.602 (9)	0.142 (9)	0.905 (9)	0.711 (9)	42.410 (13)	8
RM-SVR	1.136 (11)	0.822 (11)	0.211 (11)	1.734 (11)	0.732 (10)	1.218 (2)	9
LI-CNN	1.022 (10)	0.652 (10)	0.156 (10)	1.080 (10)	0.735 (11)	671.651 (14)	10
RM-RF	1.641 (13)	1.282 (13)	0.405 (13)	2.896 (13)	0.871 (13)	1.216 (1)	11
LI-SVR	1.401 (12)	1.041 (12)	0.305 (12)	2.287 (12)	0.817 (12)	3.689 (7)	12
LI-RF	2.086 (14)	1.541 (14)	0.523 (14)	3.576 (14)	0.904 (14)	1.387 (3)	13

Note: "LI-" is the linear interpolation model. The values in brackets indicate the ranking of each model based on the respective error metric. Rank is the result of the summation of the rankings of each indicator and then sorted.

4.2. Parallel Model

Table 9 demonstrates the superiority in performance of the RM-LSTM model compared to the RM-CNN-LSTM model. This study speculates that this result may stem from the critical role of spatial information in the prediction process, whereas CNNs may cause some degree of information loss in extracting this spatial information.

By introducing LSTM in parallel to the RM-LSTM model, we observed a significant improvement in prediction accuracy. This finding highlights the importance of temporal information on realized volatility in stock price volatility forecasting to improve forecasting accuracy.

Considering that market indicators, fundamental indicators, and macro-variables all contain rich temporal and spatial information, this study argues that the introduction of CNNs to extract spatial features of these data is crucial.

Table 9. Analysis of model prediction performance based on multiple error metrics comparing parallel models. This table compares and ranks the predictive performance of different models using different loss functions.

Model	RMSE	MAE	MSLE	MAPE	SMAPE	QLIKE	Rank
RM-LSTM-P	0.917 (2)	0.485 (1)	0.099 (1)	0.629 (2)	0.507 (1)	2.716 (2)	1
RM-LSTM	0.893 (1)	0.535 (3)	0.108 (2)	0.779 (3)	0.563 (3)	2.524 (1)	2
RM-CNN-LSTM	1.016 (3)	0.508 (2)	0.128 (3)	0.512 (1)	0.519 (2)	5.162 (3)	3

Note: The values in brackets indicate the ranking of each model based on the respective error metric. Rank is the result of the summation of the rankings of each indicator and then sorted.

5. Conclusions and Extension

In order to accurately predict the realized volatility of stocks, this paper introduces macro-indicators on the basis of market indicators and fundamental indicators of stocks. When dealing with mixed-frequency data, this paper integrates the RR-MIDAS model into the CNN-LSTM architecture, which effectively solves the problem of low-frequency macroeconomic data processing and significantly advances the volatility prediction technique. In addition, in terms of forecasting model, this paper extracts the temporal information of realized volatility by parallelizing an LSTM, which improves the forecasting accuracy of the model. Finally, the hyperparameters of the prediction model are automatically optimized through the Optuna tuning framework, which provides a new model framework for volatility prediction.

Based on the above study, this paper proposes five conclusions: first, from the perspective of introduced variables, among all models, those with macro-features have better forecasting performance. Second, in terms of prediction models, the RM-CNN-LSTM-P model proposed in this paper has the best prediction performance, and the deep-learning model has better prediction performance than the machine-learning model and the traditional econometric model. Third, from the point of view of whether parallel LSTM, the prediction performance of RM-CNN-LSTM model and RM-LSTM after parallel LSTM is improved, and LSTM can effectively extract the temporal features of the realized volatility, which improves the prediction performance of the model. Fourth, in terms of the mixing data processing method, the RR-MIDAS-based prediction model has better prediction results compared with the interpolation method. Fifth, in terms of the robustness of the results, the RM-CNN-LSTM-P model proposed in this paper has a robust performance. The results of impact experiments show that all three macro-variables, CPI, GDP, and PMI, have significant positive effects on improving the prediction accuracy of the model. The DM test results show that the prediction performance of the RM-CNN-LSTM-P model proposed in this paper is significantly better than that of other machine-learning or econometrics models.

The research in this paper has certain practical value. Firstly, for regulators, they can adjust regulatory policies and measures according to the predicted volatility situation, intervene in the market in time, and prevent the occurrence of financial risks. Secondly, for financial institutions, according to the predicted volatility situation, they can adjust the weights of different assets, choose appropriate investment products, and reduce the risk exposure of their investment portfolios. Finally, for individual investors, they can choose investment products with lower volatility according to the predicted volatility situation, so as to realize sound value-added worth of their personal investment portfolios.

The research in this paper provides innovative ideas and empirical support for further development in the field of volatility forecasting. In future volatility prediction research, on the one hand, we can further enrich the index system of volatility prediction by introducing multifactors such as market sentiment indicators to further prove the accuracy of the model; on the other hand, we can consider further improving the CNN-LSTM framework and introducing the attention mechanism to continuously improve the learning ability and prediction performance of the model.

Author Contributions: Conceptualization, Y.S.; methodology, Y.S.; software, Y.H. and W.M.; validation, W.M., Y.H. and Y.S.; formal analysis, W.M., Y.S. and Y.H.; resources, Y.S.; data curation, Y.H. and W.M.; writing—original draft preparation, W.M. and Y.H.; writing—review and editing, W.M., Y.H. and Y.S.; visualization, Y.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Shanghai Planning Project of Philosophy and Social Science (2023BJB009).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset for the empirical analysis can be derived from the following resource available in the public domain: https://www.joinquant.com/data/ and Sina Finance (https://finance.sina.com.cn) (accessed on 13 May 2024).

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

References

1. Alaali, F. The effect of oil and stock price volatility on firm level investment: The case of UK firms. *Energy Econ.* **2020**, *87*, 104731. [CrossRef]
2. Antonakakis, N.; Cunado, J.; Filis, G.; Gabauer, D.; de Gracia, F.P. Dynamic connectedness among the implied volatilities of oil prices and financial assets: New evidence of the COVID-19 pandemic. *Int. Rev. Econ. Financ.* **2023**, *83*, 114–123. [CrossRef]
3. Shahid, M.N.; Azmi, W.; Ali, M.; Islam, M.U.; Rizvi, S.A.R. Uncovering risk transmission between socially responsible investments, alternative energy investments and the implied volatility of major commodities. *Energy Econ.* **2023**, *120*, 106634. [CrossRef]
4. Engle, R.F. Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica* **1982**, *50*, 987–1007. [CrossRef]
5. Bollerslev, T. Generalized autoregressive conditional heteroskedasticity. *J. Econom.* **1986**, *31*, 307–327. [CrossRef]
6. Taylor, S.J. Modeling Stochastic Volatility: A Review and Comparative Study. *Math. Financ.* **1994**, *4*, 183–204. [CrossRef]
7. Andersen, T.G.; Bollerslev, T.; Diebold, F.X.; Labys, P. Modeling and Forecasting Realized Volatility. *Econometrica* **2003**, *71*, 579–625. [CrossRef]
8. Corsi, F. A Simple Approximate Long-Memory Model of Realized Volatility. *J. Financ. Econom.* **2009**, *7*, 174–196. [CrossRef]
9. Qiu, Y.; Song, Z.; Chen, Z. Short-term stock trends prediction based on sentiment analysis and machine learning. *Soft Comput.* **2022**, *26*, 2209–2224. [CrossRef]
10. Lei, B.; Zhang, B.; Song, Y. Volatility Forecasting for High-Frequency Financial Data Based on Web Search Index and Deep Learning Model. *Mathematics* **2021**, *9*, 320. [CrossRef]
11. Liu, J.; Li, Z.; Sun, H.; Yu, L.; Gao, W. Volatility forecasting for the shipping market indexes: An AR-SVR-GARCH approach. *Marit. Policy Manag.* **2022**, *49*, 864–881. [CrossRef]
12. Li, H.; Qiao, G. Realized volatility forecasting based on rolling SW-SVR method: Evidence from CSI 300 index. *Appl. Econ. Lett.* **2023**, *30*, 975–980. [CrossRef]
13. Sadorsky, P.; Mcaleer, M. A Random Forests Approach to Predicting Clean Energy Stock Prices. *J. Risk Financ. Manag.* **2021**, *14*, 48. [CrossRef]
14. Zhuo, Y.; Morimoto, T. A Hybrid Model for Forecasting Realized Volatility Based on Heterogeneous Autoregressive Model and Support Vector Regression. *Risks* **2024**, *12*, 12. [CrossRef]
15. Hoseinzade, E.; Haratizadeh, S. CNNpred: CNN-based stock market prediction using a diverse set of variables. *Expert Syst. Appl.* **2019**, *129*, 273–285. [CrossRef]
16. Zhou, X. Stock Price Prediction using Combined LSTM-CNN Model. In Proceedings of the 2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Taiyuan, China, 3–5 December 2021; pp. 67–71. [CrossRef]
17. Lu, W.; Li, J.; Li, Y.; Sun, A.; Wang, J. A CNN-LSTM-Based Model to Forecast Stock Prices. *Complexity* **2020**, *2020*, 6622927. [CrossRef]
18. Chen, N. Visual recognition and prediction analysis of China's real estate index and stock trend based on CNN-LSTM algorithm optimized by neural networks. *PLoS ONE* **2023**, *18*, e0282159. [CrossRef] [PubMed]
19. Amendola, A.; Candila, V.; Gallo, G.M. On the asymmetric impact of macro–variables on volatility. *Econ. Model.* **2019**, *76*, 135–152. [CrossRef]
20. Shang, Y.; Zheng, T. Mixed-frequency SV model for stock volatility and macroeconomics. *Econ. Model.* **2021**, *95*, 462–472. [CrossRef]
21. Li, D.; Zhang, L.; Li, L. Forecasting stock volatility with economic policy uncertainty: A smooth transition GARCH-MIDAS model. *Int. Rev. Financ. Anal.* **2023**, *88*, 102708. [CrossRef]
22. Ghysels, E.; Sinko, A.; Valkanov, R. MIDAS Regressions: Further Results and New Directions. *Econom. Rev.* **2007**, *26*, 53–90. [CrossRef]
23. Foroni, C.; Guérin, P.; Marcellino, M. Using low frequency information for predicting high frequency variables. *Int. J. Forecast.* **2018**, *34*, 774–787. [CrossRef]
24. Xu, Q.; Zhuo, X.; Jiang, C. Predicting market interest rates via reverse restricted MIDAS model. *J. Manag. Sci. China* **2019**, *22*, 55–71.
25. Wu, X.; Zhao, A.; Cheng, T. A Real-Time GARCH-MIDAS model. *Financ. Res. Lett.* **2023**, *56*, 104103. [CrossRef]
26. Breitung, J.; Roling, C. Forecasting Inflation Rates Using Daily Data: A Nonparametric MIDAS Approach. *J. Forecast.* **2015**, *34*, 588–603. [CrossRef]
27. Mishra, P.; Alakkari, K.; Abotaleb, M.; Singh, P.K.; Singh, S.; Ray, M.; Das, S.S.; Rahman, U.H.; Othman, A.J.; Ibragimova, N.A.; et al. Nowcasting India Economic Growth Using a Mixed-Data Sampling (MIDAS) Model (Empirical Study with Economic Policy Uncertainty–Consumer Prices Index). *Data* **2021**, *6*, 113. [CrossRef]

28. Xu, Q.; Wang, L.; Jiang, C.; Liu, Y. A novel (U)MIDAS-SVR model with multi-source market sentiment for forecasting stock returns. *Neural Comput. Appl.* **2020**, *32*, 5875–5888. [CrossRef]
29. Li, C.; Zhang, X.; Qaosar, M.; Ahmed, S.; Alam, K.M.R.; Morimoto, Y. Multi-factor Based Stock Price Prediction Using Hybrid Neural Networks with Attention Mechanism. In Proceedings of the 2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), Fukuoka, Japan, 5–8 August 2019; pp. 961–966. [CrossRef]
30. Diebold, F.X.; Mariano, R.S. Comparing Predictive Accuracy. *J. Bus. Econ. Stat.* **2002**, *20*, 134–144. [CrossRef]
31. Song, Y.; Tang, X.; Wang, H.; Ma, Z. Volatility forecasting for stock market incorporating macroeconomic variables based on GARCH-MIDAS and deep learning models. *J. Forecast.* **2023**, *42*, 51–59. [CrossRef]
32. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A Next-generation Hyperparameter Optimization Framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2623–2631. [CrossRef]
33. Bergstra, J.; Bardenet, R.; Bengio, Y.; Kégl, B. Algorithms for Hyper-Parameter Optimization. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: San Francisco, CA, USA, 2011; Volume 24.
34. Ding, Z.; Xu, D.; Li, W. Identification of Stability for the Influences of Macroeconomic Variables to the Term Structure of Interest Rate. *J. Quant. Technol. Econ.* **2014**, *31*, 56–74. [CrossRef]