applied sciences

MDPI

Article

Deep Fusion Prediction Method for Nonstationary Time Series Based on Feature Augmentation and Extraction

Yu-Lei Zhang [1,2,3], Yu-Ting Bai [1,2,3,*], Xue-Bo Jin [1,2,3,*], Ting-Li Su [1,2,3], Jian-Lei Kong [1,2,3] and Wei-Zhen Zheng [1,2,3]

1 School of Artificial Intelligence, Beijing Technology and Business University, Beijing 100048, China
2 State Environmental Protection Key Laboratory of Food Chain Pollution Control, Beijing Technology and Business University, Beijing 100048, China
3 China Light Industry Key Laboratory of Industrial Internet and Big Data, Beijing Technology and Business University, Beijing 100048, China
* Correspondence: baiyuting@btbu.edu.cn (Y.-T.B.); jinxuebo@btbu.edu.cn (X.-B.J.)

Abstract: Deep learning effectively identifies and predicts modes but faces performance reduction under few-shot learning conditions. In this paper, a time series prediction framework for small samples is proposed, including a data augmentation algorithm, time series trend decomposition, multi-model prediction, and error-based fusion. First, data samples are augmented by retaining and extracting time series features. Second, the expanded data are decomposed based on data trends, and then, multiple deep models are used for prediction. Third, the models' predictive outputs are combined with an error estimate from the intersection of covariances. Finally, the method is verified using natural systems and classic small-scale simulation datasets. The results show that the proposed method can improve the prediction accuracy of small sample sets with data augmentation and multi-model fusion.

Keywords: time series prediction; few-shot learning; data augmentation; feature decomposition; covariance intersection fusion

Citation: Zhang, Y.-L.; Bai, Y.-T.; Jin, X.-B.; Su, T.-L.; Kong, J.-L.; Zheng, W.-Z. Deep Fusion Prediction Method for Nonstationary Time Series Based on Feature Augmentation and Extraction. *Appl. Sci.* **2023**, *13*, 5088. https://doi.org/10.3390/app13085088

Academic Editor: Sangtae Ahn

Received: 2 March 2023
Revised: 11 April 2023
Accepted: 15 April 2023
Published: 19 April 2023

1. Introduction

Data are vital for various artificial and natural systems in the modern information era. The real-time data generated with equipment and sensors are critical to monitoring the controlled objects' operation statuses [1]. Moreover, advanced sensing and precaution are expected to be obtained for administrators. Then, the aforehand actions can be taken to adjust a system's operation and make effective decisions to avoid accidents and gain benefits. Therefore, close attention has been paid to prediction information in various applications, such as unmanned terminal control [2,3], environment monitoring [4–6], the stock market [7], agriculture [8], etc. In a prediction, the time series data are always of nonstationary and nonlinear features with complex noises [9]. The issue of time series prediction has been a research hotspot for a dozen years. The traditional statistical methods represented by the ARIMA (Auto-Regressive Integrated Moving Average) [10], ARCH (Auto-Regressive Conditional Heteroskedasticity) [11], and intelligent machine learning methods have been studied and applied widely. However, the methods may be invalid when the historical data are on a small scale.

The data size of a time series in practical applications is often not as large as we expect, although big data are always advocated in the internet world. The data in many fields are collected artificially in the traditional management style. For example, we investigated water environment monitoring in Beijing, China. The mainstream authoritative monitoring data are mainly collected by operators and assayed in the laboratory [12]. The frequency is one dataset of one day at most. There are only about 30 sets of monitoring data in a month on the official website. A similar phenomenon occurs in casual food inspections.

The government administration inspects different kinds of grain and foods in markets. The highest frequency is once a day based on human resources limitations and enormous market scope. There are only hundreds of inspection datasets in a year. Another typical example is demographic statistics. Population censuses have been conducted for about ten years in China. The latest censuses were in 1990, 2000, and 2010. The demographics in the highest frequency are only measured once a year, which means the number of recent population data is only in the dozens. The slow data collection makes the data size so small that it cannot meet the requirement of intelligent computing.

In many application states in the real world, the number of samples is small or minimal, and marking lots of unmarked samples will consume many human resources. Therefore, finding a way to learn with a small number of samples has become a current need. Existing prediction models mainly include shallow neural networks and deep neural networks. For existing methods, small-scale data hinder effective modeling and accurate calculations. Models fitted using statistical methods do not track new patterns because of limited information in the existing data. Building a model using machine learning methods may not be possible because the data size is too small to train the network. Deep networks often require a mass of data to train an available model. Suppose a small amount of data cannot reflect the natural characteristics of the data well. In that case, the deep network model will fail to achieve the expected accuracy, and it may lead to model overfitting.

Aiming at the problem of insufficient data volume, related research has begun to introduce some data augmentation methods to enhance the data volume of samples, such as Hermite interpolation, cubic spline interpolation, generative adversarial net, etc. A Hermite spline is a piecewise cubic polynomial with a given tangent at each value point. Cubic spline interpolation takes the curve as the primary element, connects the discrete point data into a curve, and can find the function value of any point on the curve. It solves the limitations of interpolation by fitting all discrete data points to find a continuous curve. Cubic spline interpolation is a standard method in numerical analysis that can assist in establishing the functional relationship between discrete variables. Compared with Hermite interpolation, the data curve obtained by cubic spline interpolation is smoother and more suitable for fundamental data changes. The generative adversarial net uses a GAN-based generator network to provide an effective regular representation for invisible data distributions. It uses a residual pairwise network as a discriminator to measure the similarity of paired samples. However, the extended samples obtained by these expansion methods can only guarantee the continuity of the small curves at the connection points. However, they cannot guarantee the authenticity of the sample data, which may cause cumulative errors and affect the experimental results. In addition, the complex data distribution is not captured, and the generated features are not interpretable. This cannot meet the prediction accuracy requirements of deep learning models. When expanding data samples, meeting the needs of model training and ensuring the authenticity of data sources as much as possible are still problems to be solved. Therefore, a small-sample learning method based on data enhancement is proposed to solve the problem. This data enhancement algorithm expands the characteristics of the samples to obtain more samples containing the original data's characteristics so that the model can extract features better.

This paper explores the prediction solution in small-scale time-series data. The limited data are expanded with a designed augmentation method. Then, the parallel augmentation datasets are predicted with the appropriate models corresponding to the data subsets that were decomposed [13]. The sub-prediction results are integrated with the error covariance intersection. The features of the small-scale time series are expanded and excavated in the proposed method. Moreover, the method is verified with a typical simulation of time series and practical systems.

This paper is organized as follows. In Section 2, related work on time series predictions is introduced, including the development of shallow neural networks and deep neural networks. Section 3 introduces the details of the proposed framework, including data augmentation, data decomposition based on time series trends, multi-model prediction,

and error estimation fusion with the intersection of covariances. Section 4 tests the proposed and comparative models on simulation data in MG and MSO systems and real data in a fraction ratio in a food safety check system and SO_2 concentration in an atmospheric quality monitoring system. In Section 5, the experimental results are discussed and analyzed, and the experimental results of the proposed model and the comparative model for the dataset are compared and analyzed. Section 6 summarizes the approach, and conclusions and outlooks are provided.

2. Related Works

The existing research on time series prediction mainly includes methods based on statistics and machine learning. Some studies have been discussed in the two categories of methods for small samples, but the amount of relevant research is relatively small. Classical prediction methods and studies on small samples are presented in this section. Moreover, the merits and demerits are also discussed briefly.

2.1. Statistical Prediction Methods

Prediction methods have been studied deeply based on statistical theory. The methods are established in the solution to the data rule description and relation fitting. The autocorrelation function and exponential decays are essential for studying time series variations [14]. In the generalized view, the traditional prediction methods include exponential smoothing [15], the state–space model represented by the Kalman filter [16], ARIMA [10], and ARCH [11]. The methods in the system of ARIMA are the most representative of the extensive studies and applications. The typical models include AR (Auto Regression), MA (moving average), and hybrid models. The AR model can fit the relationship of the regressor variable itself. The iteration relationship in the adjacent variables is expressed with a linear combination of the historical data. The MA model introduces the sliding window to extract the change features. The synthesis of AR and the MA can model the time series more accurately, in which ARIMA develops fast in the nonstationary regressive issue.

The primary statistical model is presented with the expression of ARIMA:

$$\left(1-\sum_{i=1}^{p}\alpha_i L^i\right)(1-L)^d x_t = \left(1+\sum_{i=1}^{q}\beta_i L^i\right)\varepsilon_t \tag{1}$$

where x_t is the value of time series at t. p is the autoregressive order, d is the differential order, and q is the moving average order. ε_t is the error term conforming to the normal distribution. L is the Lag operator. α_i and β_i are the autoregressive parameters and the shifted moving average parameter.

The primary support for the prediction methods above is the statistical theory, which does not require a strict data size. The usual data size of dozens and hundreds can meet the modeling demand. Some methods have been proposed for the small-scale data of time series. Rogoza [17] proposed the local extrapolation method to model a few samples that may be less than 10. The time series variables were derived based on the Kolmogorov–Gavor polynomial, in which the parameter identification and data sensibility were analyzed for short-term predictions. Shi et al. [18] built a preprocessing model for small samples to enhance their stationary time series. The simulated annealing algorithm and support vector machine were introduced to predict samples. Furthermore, the grey theory and method [19,20] are also studied in the traditional methods for small sample predictions.

The methods above solve the small sample issue based on mathematical descriptions in the statistical framework. The model can fit the changing trend in the available data, but one wonders if it can accord with new data outside of the existing scope. The methods can predict a small-scale time series that is stable in the short term. It may be invalid for the long-term prediction of complex nonstationary time series.

2.2. Shallow Machine Learning Methods

Machine learning provides novel access to data modeling with black-box thought. It focuses on the external representation of the data instead of the inner mathematical mechanism. BP (backpropagation) [21,22] is the classical network in time series analysis. Moreover, shallow machine learning methods are also studied widely, including the support vector machine [23], random forest [24], Bayes network [25], and the extreme learning machine [26]. The support vector machine transforms the input space into a high-dimensional space through nonlinear transformation and finds the optimal linear classification surface. This nonlinear transformation is defined as an appropriate inner product function. The random forest algorithm uses the Bootstrap sub-self-sampling method to obtain different sample sets for the construction of the model, thereby increasing the difference between the models and improving their extrapolation and prediction abilities. The extreme learning machine randomly selects the weight of the input layer and the bias of the hidden layer. The weight of the output layer minimizes the loss function, composed of the training error term, and the regular term of the weight norm of the output layer. It is calculated according to the Moore–Penrose (MP) generalized inverse matrix theory analysis.

The prediction model based on the shallow machine learning method has the advantages of a relatively mature theory, a simple algorithm structure, and certain self-learning and self-adaptive capabilities, and it can be applied to nonlinear data modeling. However, due to the substantial volatility of the currently collected time series, the prediction model based on the shallow machine learning method is ineffective in fitting nonlinear data tasks and cannot learn the long-term dependence of time series data.

2.3. Deep Learning Methods

With the promotion of techniques for data storage and computation, machine learning is replete with deep learning methods. The network structures are expanded, and more components are appended. In deep learning for time series, RNN (recurrent neural network) [27] is a typical network for the sequence features. In RNN, the hidden layer is endowed with a memory function by connecting the nodes in different components. Because RNN may diverge in the long term, LSTM (Long Short-Term Memory) [28] is proposed by introducing gate components. LSTM is improved with the redesign of the gate structure, including the bidirectional LSTM network [29] and the GRU (Gated Recurrent Unit) [30].

Deep learning develops along with big data. The mass of data provides the learning basis for deep networks. Meanwhile, the networks' complex components and deep structures need a lot of data to establish the function relationship. A test was conducted in our previous research, in which an LSTM network was trained with the classical datasets of an MG (Mackey–Glass) system and MSO (Multiple Superimposed Oscillator) systems [31]. The data size when the network converges is recorded in Table 1, in which the network is trained with different prediction steps.

Table 1. The minimum data size for training the LSTM network with simulation time series.

Prediction Steps	5	10	15
MG system dataset	1500	1700	1800
MSO system dataset	1800	1900	1950

It can be seen that the LSTM network needs at least 1500 sets of data to obtain the available model in the test. The number of prediction steps represents the step size of LSTM's backward predictions, including 5 steps, 10 steps, and 15 steps. The numbers corresponding to the datasets in Table 1 are the minimum data in the training set required to obtain an effective backward prediction in LSTM models with different step lengths. That is, 1500 sets of data mean that, in the MG system's dataset prediction task, the LSTM model trained by at least 1500 sets of training data can achieve adequate accuracy in the

test set, including data and labels. The LSTM-related parameter settings are shown in Table 2: *input-size* and *output-size* represent the time series dimension of the network's input and output, respectively, *layers* represent the number of layers in the network, *hidden unit* represents the number of hidden neurons, *batch size* represents the number of samples imported into the network simultaneously, and *leaking rate* represents the learning rate parameter. The settings of the parameters are based on the general situation, exploring how many sets of training data can be trained to achieve sufficient accuracy in the test set (including data and labels) of the LSTM model trained.

Table 2. Parameter settings related to the LSTM model.

Input-Size	*Output-Size*	*Layers*	*Hidden Unit*	*Batch Size*	*Leaking Rate*
1	1	2	32	60	0.001

The data size increases with the prediction steps. Furthermore, the classical data used in the test are relatively regular, without complex noises and uncertain mutations. A prediction model based on the deep learning method can automatically learn the hidden characteristic information in the data and capture nonlinear relationships. Although the deep learning method has a strong learning ability, when the data are more complex and unstable, this learning ability requires a large amount of data. The prediction accuracy is often unsatisfactory if the amount of data is insufficient. Few-shot learning cannot meet the data volume requirements of deep networks. Although the use of the width learning system can update the model training weight faster than a deep learning network, thus improving the training speed of the model, it is still limited by the amount of data imposed by few-shot learning.

Compared with the recurrent neural network and the convolutional neural network, the transformer model has the advantage of parallel computing, which improves the speed of model training and reasoning and is one of the main advantages of the model. However, the transformer model, as a deep learning algorithm, naturally has the defect of deep learning models in few-shot learning. Due to its large model size and high-dimensional input features, the transformer model requires a large amount of training data to avoid overfitting and improve its generalization ability. Therefore, the transformer model does not have an advantage in the field of few-shot learning.

In addition, in addition to using a single model to solve nonlinear time series forecasting problems, the technique of applying a combination of models based on machine learning has also begun to appear. For example, Xiong et al. [32] propose a novel hybrid method combining seasonal–trend decomposition procedures based on loess (STL) and extreme learning machines (ELMs) for the short-, medium-, and long-term forecasting of seasonal vegetable prices. Liu et al. [33] propose a hybrid model combining inputs selected using deep quantitative analysis, wavelet transform (WT), the genetic algorithm (GA), and support vector machines (SVM).

This discussion can be concluded with a brief survey of traditional statistical methods and modern deep-learning networks. Existing methods have limitations in the field of few-shot learning predictions. Statistical methods require a relatively small amount of data. However, feature extraction is weak for unknown potential trends outside the range of the existing data. The existing methods of shallow machine learning networks have a single structure, a weak generalization ability, and cannot learn long-term data dependencies. A deep network needs abundant data to train a model with high prediction accuracy, and few-shot learning limits the data volume demand of a deep network. Furthermore, different methods specialize in handling different data, such as ARIMA for stationary time series and LSTM for strongly nonlinear series with noise. For the ideal prediction of small samples, two aspects should be considered: first, the sufficient expansion and feature extraction of small samples and, second, the matching of models with different data trends.

3. Fusion Prediction Based on Data Augmentation

3.1. Framework of Fusion Prediction in Data Augmentation

Deep networks are expected to excavate the features of small-scale time series. To apply deep networks, the issue of the demand for a large data size should be solved first. In addition, a deep network cannot be regarded as an almighty tool because it can be improved in a complex environment. The novel prediction method is designed from two perspectives: the augmentation of small samples and a distributed prediction based on GRU. The framework of the proposed prediction method is shown in Figure 1.

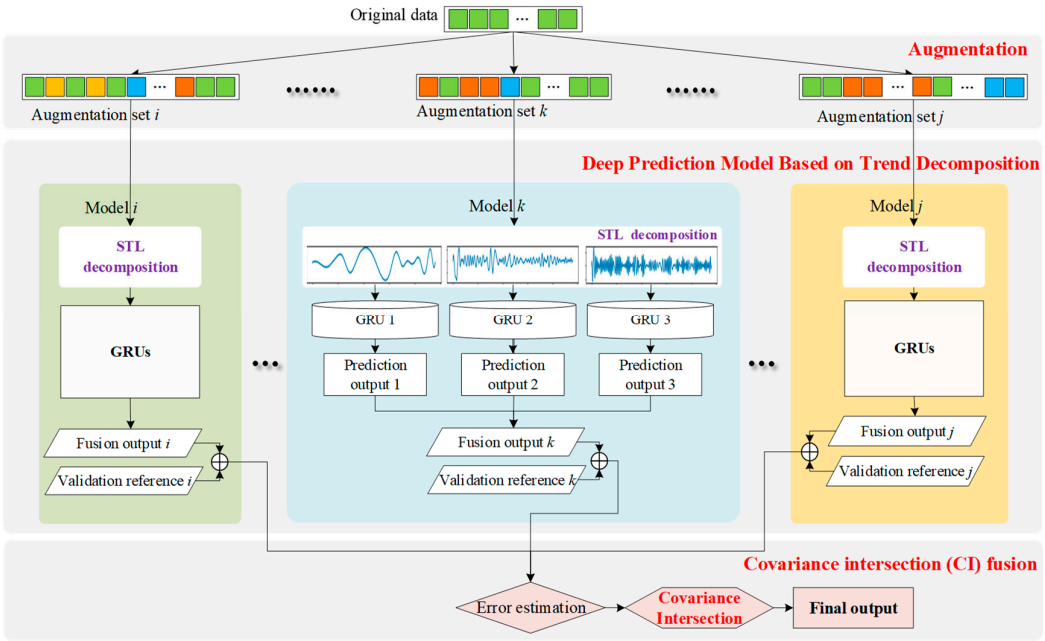


Figure 1. The framework of the fusion prediction model based on the augmentation of the original data.

First, the augmentation algorithm should expand the original data based on random probability because small samples cannot satisfy the large amount of training data required by the deep learning model. Multiple augmentation sets are generated to meet the data amount requirements. Each augmentation set will be input for the multi-model prediction in the next step. The augmentation of small-scale time series will be elaborated in Section 3.2.

Second, a deep prediction model based on trend decomposition is designed for the augmentation sets. For each augmentation set, the STL (seasonal and trend decomposition using loess) decomposition is applied to the set to generate three components: the trend component, the seasonal component, and the remainder component subsequence. The decomposed features are predicted with GRU models. Then, the prediction results of each GRU are fused as the prediction output of the augmentation set. The error indexes of each model are output together with the prediction results, such as with RMSE (Root Mean Square Error) and MAE (Mean Absolute Error), which provide the basis of the subsequent fusing of the outputs of multiple augmentation sets. The proposed deep prediction model based on trend decomposition will be elaborated in Section 3.3.

Third, the output results of each augmentation set are fused to obtain the final output. The output of each augmentation set includes two parts: the prediction result of the deep networks based on trend decomposition and the error indexes. The covariance intersection (CI) fusion algorithm is used based on the error indexes to obtain the final output. The covariance intersection algorithm based on the prediction error index will be elaborated in Section 3.4.

As mentioned above, the method consists of three major parts.

The first part is the augmentation part, as shown in the augmentation part in Figure 1. An algorithm of augmentation based on random probability expands the original data of small-scale time series to generate augmentation sets of different sizes to train deep learning models, as shown in augmentation sets i, j, and k. The augmentation results are shown in the colored blocks in the augmentation section. The green blocks represent the original data, and the blocks of other colors represent the sample data obtained via resampling based on the random probability algorithm. Each augmentation set will be used as the input of the corresponding model to learn the trend characteristics of the time series.

The second major part is a deep prediction model based on trend decomposition for parallel datasets. In each augmentation set, the augmented data from the original samples obtained using a deep prediction model based on trend decomposition are nonlinear and nonstationary. Therefore, to reduce the complexity of the trend in the augmentation set, the STL algorithm is used to decompose it, and three subsequences of the trend, cycle, and fluctuation of the augmentation set are obtained. Then, the GRU is used to predict each subsequence and fuse the prediction results of the three subsequences to obtain the prediction output of a single augmentation set. In addition, to fuse the prediction results of the multiple deep prediction models in the next step, the prediction error index of each augmentation set corresponding to the model is also output as a verification reference for model performance.

The third part is the fusion part. Based on the prediction error index of each model, the covariance intersection fusion algorithm is used to fuse the prediction results output by the multiple models to obtain the final output. In the fusion of the prediction results of the multiple models, the fusion is based on the prediction error index of each model, which not only considers the prediction performance of each model but also makes full use of the advantages of different models. It avoids the impact on the proposed framework's final prediction result due to the model's poor prediction result corresponding to a single augmentation set.

In the three major parts of the framework above, it can be seen that the main issues include the augmentation of the original small samples, the distributed deep prediction model, and the integration method based on the covariate intersection. The three parts are introduced as follows.

3.2. Augmentation of Small-Scale Time Series

In the framework proposed, the precondition for deep network prediction is the augmentation of the original data. Small samples should be expanded to meet the data size demand of the deep networks. Therefore, an augmentation algorithm based on random probability is proposed first.

The original dataset is expressed as $\{x_1, x_2, \cdots, x_n\}$, where n is the number of samples. The expansion of the small-scale time series is achieved through the following steps. First, the parameters related to resampling are set. The minimum data size of the network according to the needs of the augmented sample is set. The number of resamplings, m, can be ascertained with the limitation of $m \cdot n \geq \varepsilon$. The resampling is conducted following the Bootstrap method [34,35]. If samples are unknown, multiple samplings in the existing partial samples construct the estimated confidence interval. In an abstract sense, the usual estimation from the samples does not extract all the information, while Bootstrap builds the confidence interval by exerting the residual value with the resampling. Third, for the original set, the sample is extracted with playback n times, and the new set is $X_j (1 \leq j \leq m)$.

Meanwhile, the probability distribution parameter, θ_j, is calculated for the new set. After the resampling m times, the new sets are obtained and joined as $Y_k{}' = \{X_1, X_2, \cdots, X_m\}^k$ and $1 \leq k \leq \alpha$, in which X_j consists of n samples. The corresponding parameter set is

$\{\theta_1, \theta_2, \cdots, \theta_m\}$ of which the a-th and b-th parameters are the lowest and highest points of the confidence interval:

$$\begin{cases} a = \text{intfloor}\left(m \cdot {}^{\beta}\Big/_{2} \cdot 100\%\right) \\ b = \text{intfloor}(m \cdot (1-\beta) \cdot 100\%) \end{cases} \tag{2}$$

where β is the quantile parameter. Finally, the samples and the resampled new set are fused. b samples are selected from the original data and inserted evenly into the newly joined set. Small sample augmentation is performed through the process of resampling and union. First, the relevant resampling parameters are set, and the number of resampling times is determined. Second, confidence intervals for the estimates are constructed by taking multiple samples across the existing partial sample. Third, for the original set, samples are extracted to obtain a new set after n playbacks. At the same time, the probability distribution parameters for the new set are calculated. Finally, the samples and the resampled new set are fused. The augmentation can be conducted α times considering the multiple models trained later. The augmentation Algorithm 1 is concluded as follows for a clear and logical presentation.

Algorithm 1: Augmentation of a small sample.

Input: $\{x_1, x_2, \cdots, x_n\}$, m, α, β
Output: Y_k
Procedure
For k = 1, k <= α, k++
    For j = 1, j < = m, j++
        resampling of $\{x_1, x_2, \cdots, x_n\} \rightarrow X_j$;
        calculate probability distribution parameter θ_j;
    For end
    join $X_j \rightarrow {Y_k}' = \{X_1, X_2, \cdots, X_m\}^k$;
    join $\theta_j \rightarrow \{\theta_1, \theta_2, \cdots, \theta_m\}$;
    calculate confidence interval parameters a and b following Formula (2);
    select b samples from $\{x_1, x_2, \cdots, x_n\}$ evenly;
    insert b samples selected above into ${Y_k}'$ evenly $\rightarrow Y_k$;
For end
Procedure end

3.3. Deep Prediction Model Based on Trend Decomposition

Obviously, the simpler the data trend, the better the prediction result. In this method, the augmentation sets are decomposed into subsets for specific analysis. In the decomposition, STL (seasonal–trend decomposition procedure based on loess) [36] is introduced to obtain the subsets of trend, period, and fluctuation. STL is formulated in independent components, as shown in Formula (3).

$$Y_t = T_t + S_t + R_t \; t = 1, 2, \ldots, N \tag{3}$$

where T_t, S_t, and R_t are the trend, period, and fluctuation components, respectively. The loess (locally weighted scatterplot smoothing) [37] smoother is based on fitting a weighted polynomial regression for a given observation time, where weights decrease with their distance from the nearest neighbor. The calculation process of STL is presented with the loop mode in Figure 2. The time series is fitted iteratively until the trend and seasonality stabilize. In a multi-step process, moving averages alternate with loess smoothing.

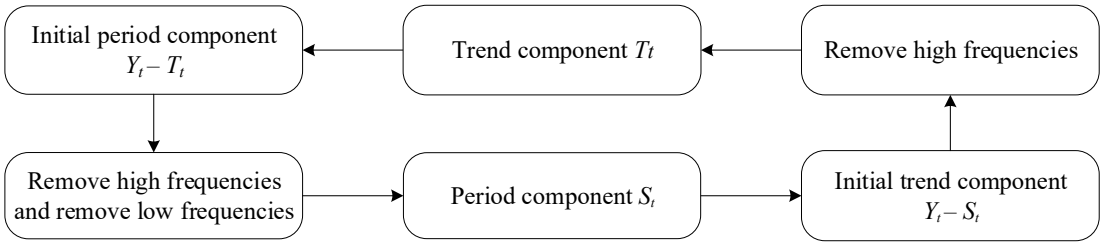


Figure 2. Flowchart of the data decomposition based on STL.

The STL algorithm includes a series of loess smoothing, using loess smoothing periodic subsequences on the original data and extending a period before and after each. The window length is $n_{(p)}$. The resulting periodic subsequence is denoted as $C_v^{(k+1)}, v = -n_{(p)} + 1, \ldots, -N + n_{(p)}$. A moving average is performed on the obtained periodic subsequences, $C_v^{(k+1)}$. Then, loess is used to smooth the sequence. Finally, the low-throughput sequence of the periodic subsequence is obtained, denoted as $L_v^{(k+1)}, v = 1, \ldots, -N$. The trend component is subtracted from the smoothed periodic subsequence to obtain the periodic component, denoted as $S_v^{(k+1)} = C_v^{(k+1)} - L_v^{(k+1)}$.

Three subsequences of trend, period, and fluctuation are obtained through the STL algorithm for the augmentation set. The GRU is used for fitting and feature extraction for each subsequence separately. The basic structure of the GRU is shown in Figure 3, in which two types of gates are the main components. The update gate (block 2 in Figure 3) controls the degree to which the state information at a previous time is brought into the current state. The larger the value of the update gate, the more status information from a previous moment is brought in. The reset gate (block 1 in Figure 3) decides how much information is written to the current candidate activation, $\widetilde{h}_t$, in the previous state. The smaller the reset gate, the less information about the previous state is written.

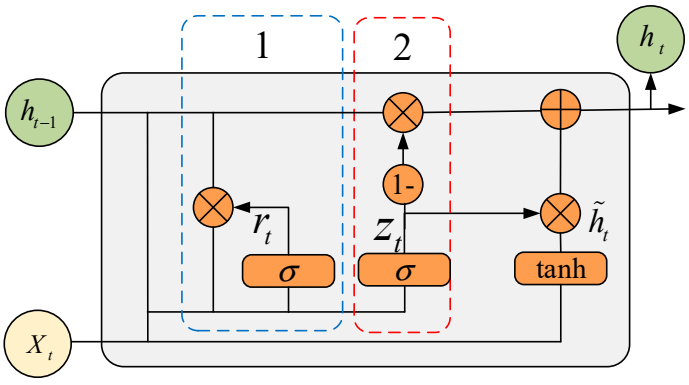


Figure 3. Structure of a GRU cell.

The output, h_t, from each GRU cell in Figure 3 is obtained as

$$\begin{aligned} z_t &= \sigma(x_t U^z + h_{t-1} W^z + b^z) \\ r_t &= \sigma(x_t U^r + h_{t-1} W^r + b^r) \\ \widetilde{h}_t &= \tanh(x_t U^h + (h_{t-1} \circ r_t) W^h + b^h) \\ h_t &= (1 - z_t) \circ \widetilde{h}_t + z_t \circ h_{t-1} \end{aligned} \tag{4}$$

where z_t, r_t, b^z, and b^r are the update gate, reset gate, current candidate activation, and activation of the GRU at time t, respectively. $\widetilde{h}_t$ represents the hidden state of the previous time step. U^z, U^r, U^h, W^z, W^r, and W^h are weight matrices to be learned during model training. $\circ$ is an element-wise multiplication. σ and tanh are activation functions.

The decomposed subsets are predicted with three GRU networks correspondingly. The sub-outputs are added to obtain the result of the augmentation subset.

3.4. Fusion Method Based on Error Covariance Intersection

Multiple models are trained in correspondence with the augmentation sets. The models can extract the different features of the original data by expanding with the random probability mechanism. In the practical prediction, multiple models are applied simultaneously, and the outputs should be fused for the final output. In the fusion, the prediction ability and performance of the models should be regarded as the pivotal factors. Then, the fusion method is introduced based on the covariance intersection of the network errors.

The prediction errors in the networks are set as the essential information to evaluate the network performance. In the augmentation set, parts of the data train the network, while others are set as the validation reference. The prediction error can be calculated with the validation data, providing the basis for the fusion weight. The validation data are y_t^i, and the predicted results are $\hat{y}_t^i$, where t is the time step, n is n time steps, and i is the serial number of the augmentation and network. For each network, the estimation variance, p_i, is calculated as

$$p_i = \sqrt{\sum_{t=1}^{n} \left(y_t^i - \hat{y}_t^i\right)^2} \Big/ n \tag{5}$$

and the covariance of all networks, p, is

$$p^{-1} = \sum_{i=1}^{m} \omega_i p_i^{-1} \tag{6}$$

where ω is the weight of the network, and m is the number of networks. The covariance is expected to be minimal, which means the estimation is balanced with the steady effect. Then, the calculation of the covariance can be converted to the optimization problem, and the model is expressed as

$$\begin{aligned} &\min_{\omega_i} p^{-1} = \sum_{i=1}^{m} \omega_i p_i^{-1} \\ &s.t. \begin{cases} \omega_i \in [0,1] \\ \sum_i^m \omega_i = 1 \end{cases} \end{aligned} \tag{7}$$

Various methods can be selected for the optimization problem with the existing research [38]. Sequential least square is used to optimize p^{-1} in this paper. Then, the fusion result, $\hat{y}$, can be obtained with the optimization resolution:

$$p^{-1}\hat{y}_t = \sum_{i=1}^{m} \omega_i p_i^{-1} \hat{y}_t^i \tag{8}$$

The final prediction result can be achieved with the framework and algorithms above. The original small samples are augmented to generate the long-term time series. The augmentation should be conducted with the algorithm in Section 3.2. Then, multiple networks are trained with the decomposition of the augmentation sets and GRU models. Finally, the fusion solution is presented by assessing the prediction error with the covariance intersection optimization.

4. Simulation and Experiment

Experiments are designed and carried out with the simulation and practical data, of which the number is relatively small for the usual deep models. The proposed fusion prediction method is tested using data augmentation and multiple prediction models. The experiments and results presented in this section are divided into the simulation and practical parts.

The simulation and experiment were conducted in the open-source deep learning library Keras (https://keras.io/zh/ (accessed on 17 October 2022)) based on Tensorflow

(http://www.tensorflow.cn/ (accessed on 19 October 2022)). All experiments were performed on a PC server with an Intel CORE CPU i5-4200U at 1.60 GHz with 4 GB of memory.

4.1. Test on Typical Simulated Time Series

4.1.1. Simulation Data

For time series predictions, typical data trends are usually studied. The following part selects two time series benchmarks, the Mackey–Glass (MG) system and the multiple superimposed oscillator (MSO) problems. For each dataset, 1350 sets of data are selected for experiments, and the first 1000 sets are the training set of the model, which is used to train the proposed model. The following 350 sets of data are divided into the first 150 sets to verify the model's accuracy, and the last 200 sets of data are used to test the model's predictive performance.

The MG system is a typical series model with a chaotic attractor. It is derived from a time-delay differential system:

$$\frac{dy(t)}{dt} = \frac{ay(t-\tau)}{1+y^n(t-\tau)} + by(t) \tag{9}$$

where n is the order of the time series variable, and a and b are the adjusting parameters. The MG system will be chaotic when $\tau < 16.8$. In the experiment, the parameters are set as $\tau = 16$, $n = 4$, $a = 0.1$, and $b = 0.1$. Then, 1350 sets of data are generated. The original data of the MG system is shown in Figure 4a.

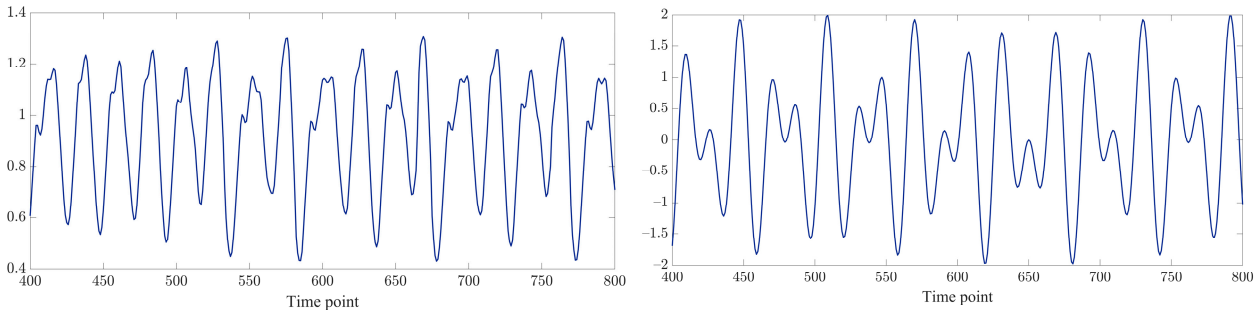


Figure 4. Original simulation data used in the experiments. (**a**) Data from MG system. (**b**) Data from MSO system.

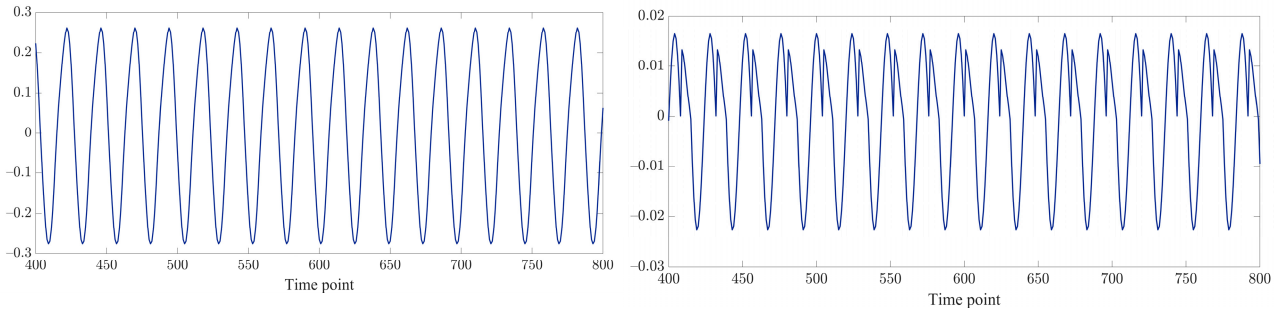
Data in the MSO system are generated by overlaying several simple sine wave functions, expressed as follows:

$$y(n) = \sum_{i=1}^{k} \sin(\alpha_i n) \tag{10}$$

where n is the time step order, k is the number of sine waves, and α_i is the frequency of the sine waves. In the experiment, $k = 6$, and α_i is set as a random value in $[0, 1]$. The data size is the same as that of the MG system. The original data are presented in Figure 4b. The simulation data in MG and MSO systems are relatively periodic. Parts of the data are shown in Figure 4 for an intuitional observation.

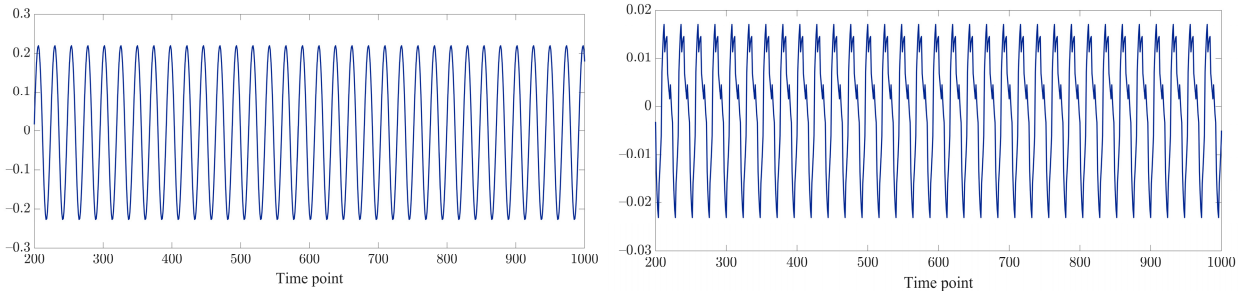
Based on the state trend characteristics of the time series data, the STL algorithm can decompose the data into three parts: seasonal, trend, and random items. The simulation data in the MG and MSO systems are decomposed into time series using the STL algorithm, and the seasonal subsequences are obtained, as shown in Figure 5. In the proposed time series prediction framework, input data are subjected to a sample enhancement algorithm to obtain multiple enhancement sets. Similarly, the enhancement set obtained through the sample enhancement algorithm is decomposed using STL to obtain seasonal subsequences, as shown in Figure 6. The overall trend of the data fluctuates significantly, with a certain

degree of seasonality. The seasonal trend of the enhancement set after the enhancement algorithm is similar to the seasonal trend of the original simulation data in Figure 6. In addition, each enhancement set will also be used as input for the corresponding multi-model prediction in the next step of training.



(**a**) (**b**)

Figure 5. Seasonal subsequences of original simulation data. (**a**) Seasonal subsequences of the MG system. (**b**) Seasonal subsequences of the MSO system.



(**a**) (**b**)

Figure 6. Seasonal subsequences of the augmentation set from the original simulation data. (**a**) Seasonal subsequences of the augmentation set from the MG system. (**b**) Seasonal subsequences of the augmentation set from the MSO system.

Concrete model building is conducted following the method in Section 3, including training data augmentation, data trend decomposition, parallel prediction with multiple GRUs, and a result fusion based on the error covariance intersection. Because the proposed method mainly consists of the sample augmentation and covariance intersection fusion of multiple GRUs, it is abbreviated as A-CI-GRUs. Other typical prediction methods are set as the contrast, as shown in Table 3. In order to exclude other interferences, the same model parameters in the experiments are set consistently.

Table 3. Related parameter settings for different models.

	Model	Parameter Settings
Statistical model	ARIMA	$p = 2$, $d = 1$, $q = 2$
Deep learning model	LSTM	*layers* = 2, *nodes* = 16, *batch size* = 60, $\eta = 0.03$
	GRU	*layers* = 2, *nodes* = 16, *batch size* = 60, $\eta = 0.03$
	Transformer	*seq_len* = 24, *label_len* = 24, *n_heads* = 8 *layers* = 2, *d_layers* = 1, *d_ff* = 2048, *factor* = 5 *dropout* = 0.05, *batch_size* = 60, $\eta = 0.03$
Multi-model Fusion	CI-GRUs	*layers* = 2, *nodes* = 16, *batch size* = 60, $\eta = 0.03$
	A-CI-GRUs	*layers* = 2, *nodes* = 16, *batch size* = 60, $\eta = 0.03$

(1) ARIMA: The typical model in the statistical prediction methods. Its main parameters, p and q, are determined according to the training data.

(2) LSTM: The efficient recurrent neural network in deep learning.

(3) GRU: The development of LSTM, which is more efficient in a relatively simple structure. The parameters of LSTM and GRU are adjusted to obtain the optimal training result.

(4) Transformer relies on the attention mechanism to represent the global dependencies between the input and output of the model. The transformer's core is the self-attention module, which can be viewed as a fully connected layer whose weights are dynamically generated based on the pairwise similarity of the input patterns.

(5) Fusion model of multiple GRUs: Considering our method's model framework, the part of the proposed model is built, namely, the multiple GRUs without augmented samples. The GRUs are integrated with the covariance intersection, the same as the proposed method. The contrast method is abbreviated as CI-GRUs.

In Table 3, *layers* are the number of network layers, *nodes* are the number of hidden nodes, *batch size* is the number of training steps, and η is the learning rate. *Seq_len* represents the input sequence length of the transformer encoder, and *label_len* represents the starting token length of the transformer decoder. *n_heads* represents the number of heads, *e_layers* represents the number of encoder layers, *d_layers* represents the number of decoder layers, *d_ff* represents the dimension of function, and *factor* represents the probsparse attention factor. *dropout* means the probability of dropout, and *batch_size* represents the batch size of the training input data.

Prediction methods perform differently in various step lengths. The proposed and comparison methods were tested in experiments with backward predictions of 5, 10, and 15 time steps. RMSE and MAE are introduced as the error evaluation criteria.

4.1.2. Test Results

In the experiment, 200 sets of data were tested using the proposed and contrast methods. The general prediction outputs are shown with curve graphs, for which the results of the MG system are shown in Figure 7, and the MSO system's results are shown in Figure 8.

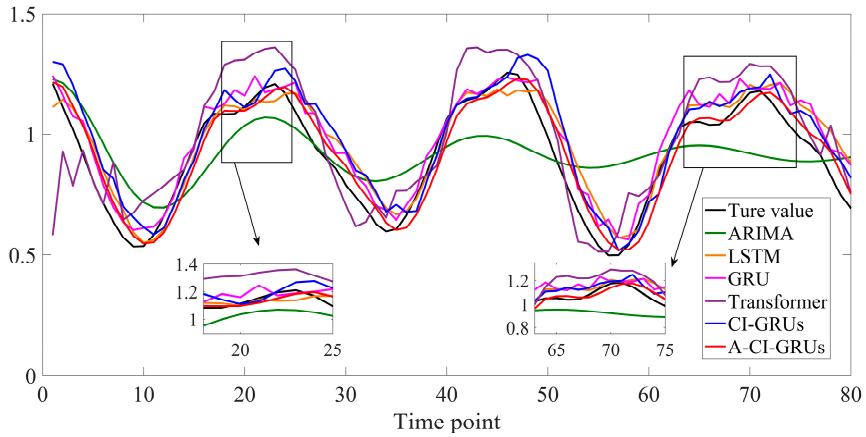


Figure 7. Prediction results of data in the MG system.

Figures 7 and 8 show the performance of different methods. In total, 200 sets of data in the MS and MSO systems were tested. ARIMA performs poorly for the losing track in the MS system and shifting in the MSO system. Other methods can trace the general trend, but their approximate degree varies. The LSTM network fluctuates more acutely, and the GRU develops a little. The fluctuation of the transformer is relatively more severe. CI-GRUs and A-CI-GRUs are closer to the true value, and their deviations are smaller than others. For an intuitional comparison, the absolute errors in all methods are drawn in Figures 9 and 10.

For the prediction length performance, the error criteria are calculated for different steps, listed in Table 4.

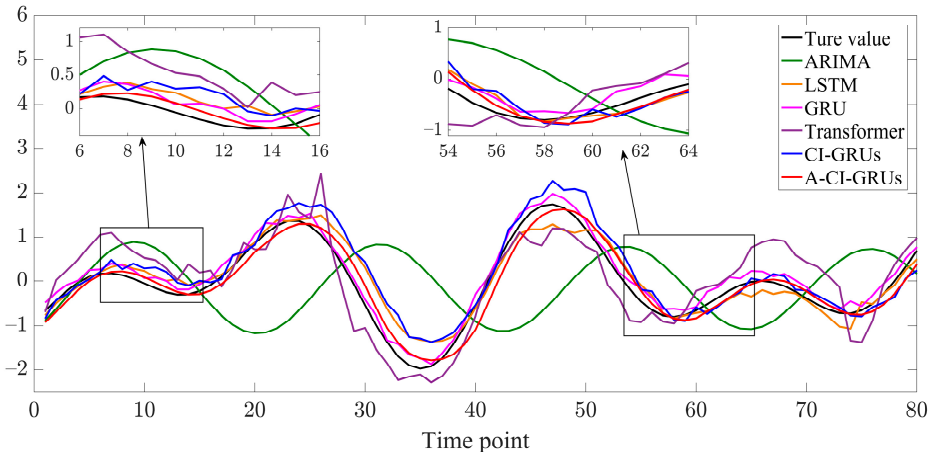


Figure 8. Prediction results of data in the MSO system.

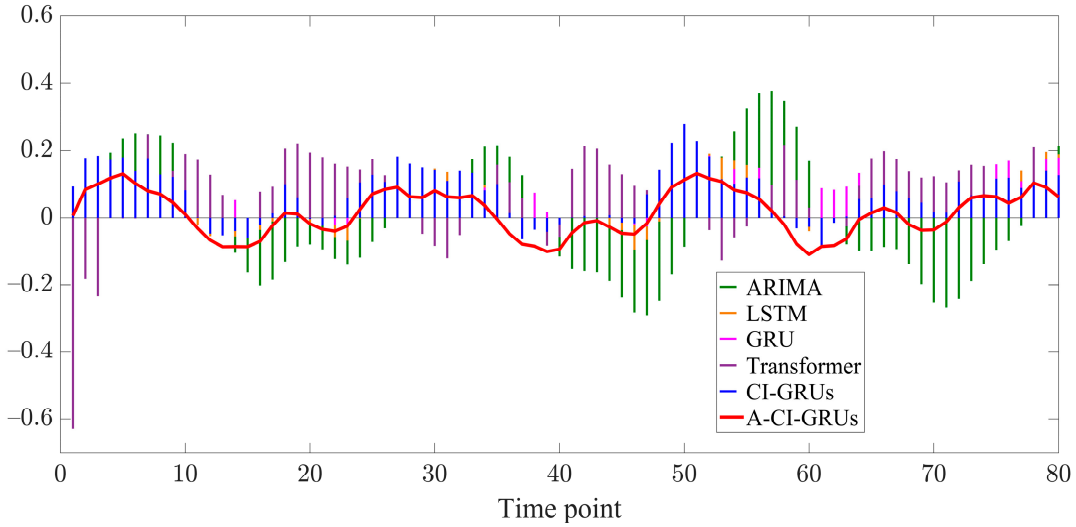


Figure 9. Errors in the prediction methods in the MG system.

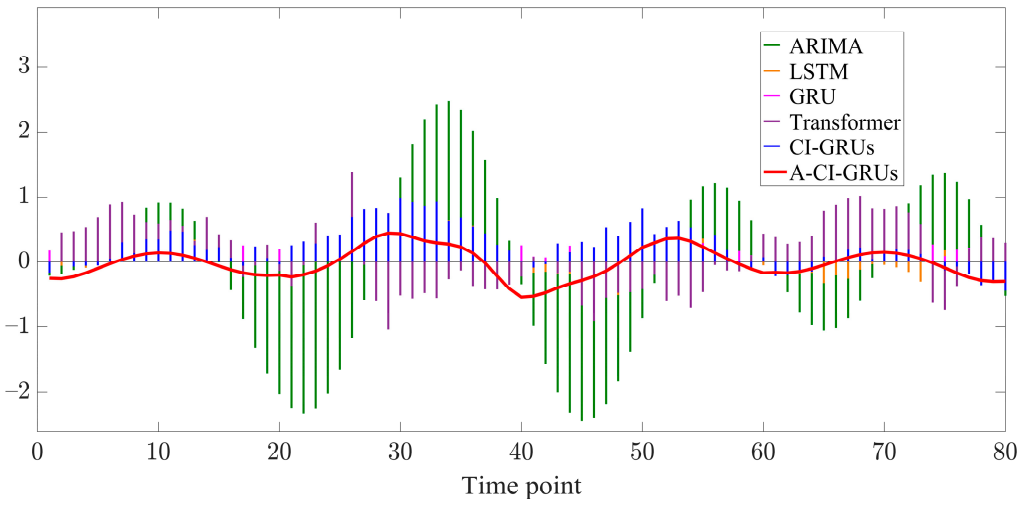


Figure 10. Errors in the prediction methods in the MSO system.

Table 4. Errors in the prediction result in different steps in the MG system and MSO system.

		MG System			MSO System		
		5	**10**	**15**	**5**	**10**	**15**
RMSE	ARIMA	0.0860	0.1210	0.1109	1.2327	1.3584	1.5893
	LSTM	0.0577	0.0712	0.0832	0.6205	0.8289	0.8938
	GRU	0.0472	0.0689	0.0789	0.2444	0.3589	0.3623
	Transformer	0.1379	0.1590	0.1642	0.4485	0.6149	0.7211
	CI-GRUs	0.0356	0.0412	0.0634	0.3623	0.4012	0.4265
	A-CI-GRUs	0.0269	0.0313	0.0529	0.2343	0.2783	0.3074
MAE	ARIMA	0.1889	0.2034	0.2101	1.0061	1.3342	1.3983
	LSTM	0.1223	0.1324	0.1423	0.4955	0.5253	0.5321
	GRU	0.1070	0.1132	0.1198	0.2160	0.3129	0.3983
	Transformer	0.1093	0.1268	0.1286	0.3600	0.4863	0.5880
	CI-GRUs	0.0729	0.0805	0.0894	0.2817	0.3029	0.3182
	A-CI-GRUs	0.0583	0.0612	0.0669	0.2054	0.2983	0.3056

The error in each point in Figures 9 and 10 shows the same performance corresponding to Figures 7 and 8. The general descending sort of all methods is A-CI-GRUs, CI-GRUs, GRU, transformer, LSTM, and ARIMA. A conclusion can be drawn following the error criteria in Table 4. From the perspective of the different methods, their errors decrease progressively from ARIMA to the A-CI-GRUs, which is similar to the graph trend in Figures 7–10. From the perspective of step length, the error rises when the step increases. The increment is dramatic in ARIMA, LSTM, and transformer. Multiple models are affected little by the step length, and errors in the A-CI-GRUs are relatively stable.

4.2. Test on Time Series in Practical Systems

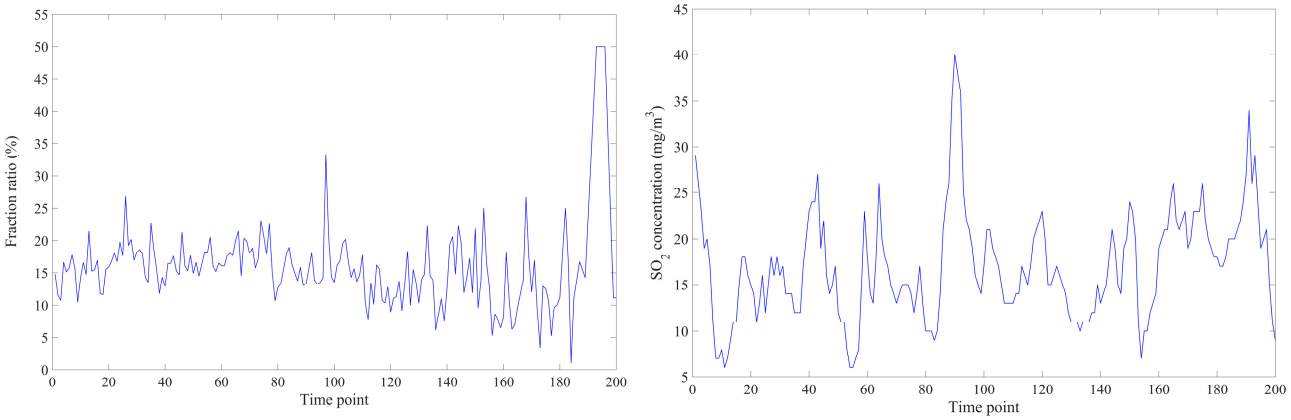
4.2.1. Dataset

The food safety check data were collected from the websites of the Chinese General Administration of Quality Supervision, Inspection, and Quarantine. The data are from 26 provinces of China from 2013 to 2018. The sampling area and scope consider the production and consumption conditions. The food safety attributes of rice are measured and integrated as a comprehensive index, namely, the fraction ratio, to present its safety degree. The data are organized as a time series by value per day.

For the atmospheric quality data, the concentration of SO_2 is set as the main index. It was measured at an industrial park in Hebei Province, China. A segment of data from December 2017 is used in the experiment.

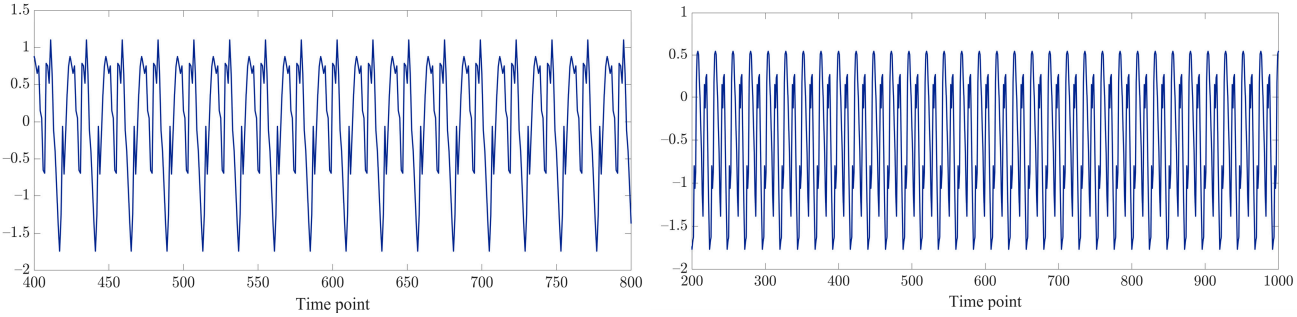
As mentioned in the introductory section, many practical systems generate data at a low frequency according to monitoring demands and the manual sampling approach. Therefore, practical data are tested except for the simulation systems above. The data from the spot check system of food safety are recorded once a day. The data from the atmospheric quality monitoring system are the typical time series. Then, parts of the data are shown in Figure 11 for an intuitional observation. The two categories of data are tested with our proposed and contrast methods in the same mode as a simulation experiment. The data size is set as 1000 sets in training, 300 in the validation, and 200 in the test.

Similarly, the original practical data used in the experiments is decomposed using the STL algorithm to obtain seasonal subsequences, and the enhancement set obtained through the sample enhancement algorithm is decomposed using STL to obtain seasonal subsequences, as shown in Figures 12 and 13. Although practical data have more complex characteristic trends and more severe volatility, the seasonal subsequences decomposed from the augmented practical data in Figures 12 and 13 still have a high similarity in the same trend as the original practical data.



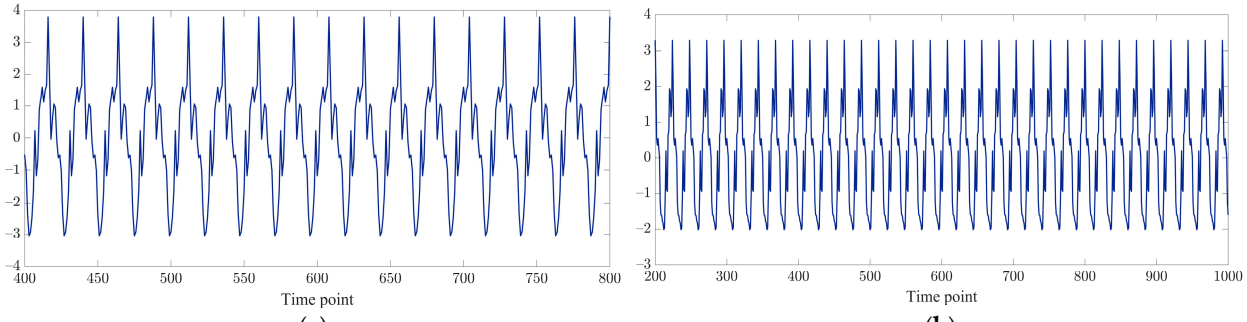
(**a**) (**b**)

Figure 11. Original practical data used in the experiments. (**a**) Data from food safety fraction ratio. (**b**) Data from SO_2 concentration.



(**a**) (**b**)

Figure 12. Seasonal subsequences of the food safety fraction ratio. (**a**) Seasonal subsequences of original practical data. (**b**) Seasonal subsequences of the augmentation set from the food safety fraction ratio.



(**a**) (**b**)

Figure 13. Seasonal subsequences of the SO_2 concentration. (**a**) Seasonal subsequences of original practical data. (**b**) Seasonal subsequences of the augmentation set from the SO_2 concentration.

4.2.2. Test Results

The experiment is similar to the simulation; the prediction results are obtained from the proposed and contrast methods with different step lengths. The general results are shown in Figures 14 and 15.

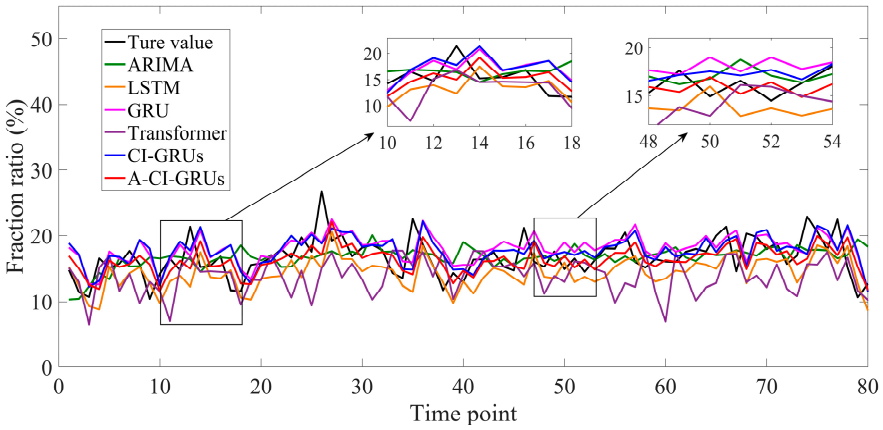


Figure 14. Prediction results of the fraction ratio in the food safety spot check system.

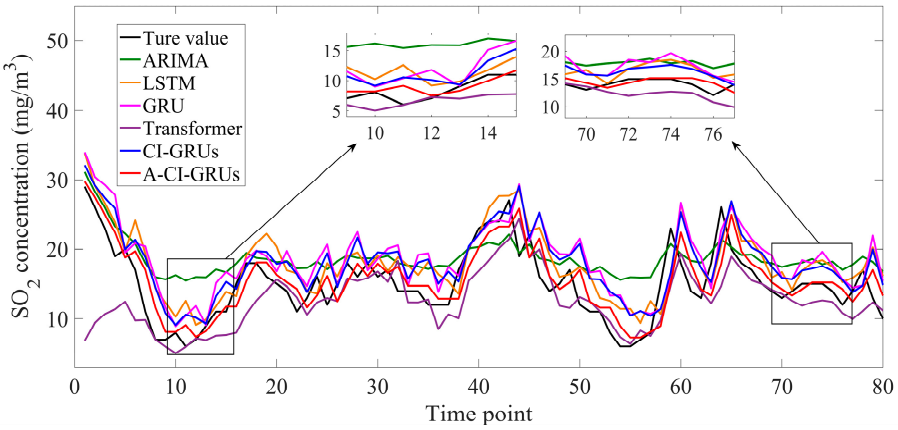


Figure 15. Prediction results of the SO_2 concentration in the atmospheric quality monitoring system.

From Figures 14 and 15, the practical data show pronounced fluctuations without evident periodicity. ARIMA can hardly trace dramatic data changes, especially for the large saltation. LSTM, transformer, and the GRU can only present the general trend, of which the compactness to the true value is low. Compared with the virtual dataset with a relatively simple fitting data trend, transformer's performance is not ideal when fitting the real dataset with a more complex trend. The results of the CI-GRUs and A-CI-GRUs are closer to the benchmark data, although there is a minor backward shifting in the prediction results for SO_2. The corresponding absolute errors are presented in Figures 16 and 17. The factual errors are calculated according to the types of methods and prediction steps, which are listed in Table 5.

Figures 16 and 17 show a noticeable distinction in various methods. For the error in the five-step method, the A-CI-GRUs reduce the RMSE by 0.54 in the CI-GRUs, 0.24 in transformer, 1.22 in the GRU, 2.68 in LSTM, and 1.99 in ARIMA in the prediction of the food safety fraction ratio. Similarly, the RMSE decreases from 8.69 to 2.87, and MAE reduces from 5.74 to 2.17 in the prediction of the SO_2 concentration. There is also a performance reduction with the prediction step length increasing. The average increment of the RMSE in the A-CI-GRUs is 0.49, whereas it is 1.16, 0.40, 0.66, 1.14, and 0.72 using the other contrast methods. Tables 4 and 5 show that the A-CI-GRUs perform best in the five datasets derived from the RMSE and MAE indicators. This verifies that the proposed prediction framework is practical. In addition, when the backward prediction steps of the model increase, the prediction error increment of the A-CI-GRUs is also smaller than that of the comparison model.

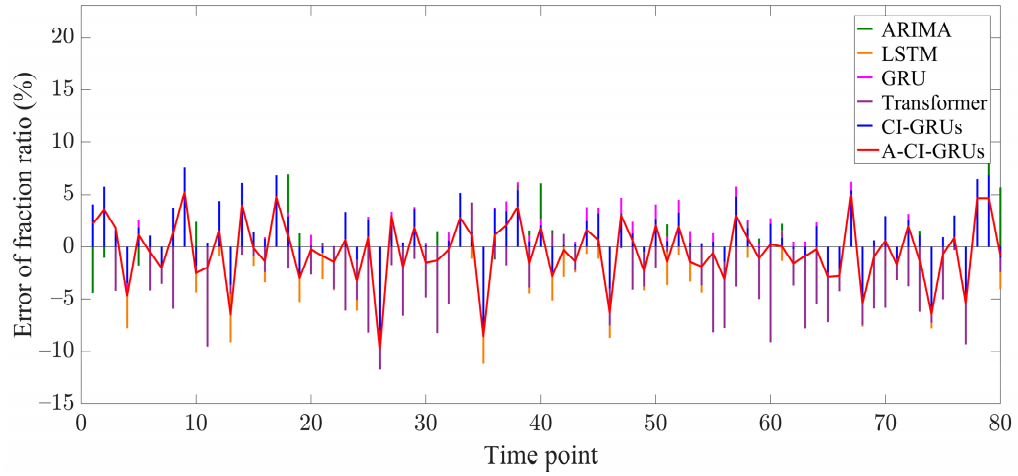


Figure 16. Errors in prediction methods for the food fraction ratio.

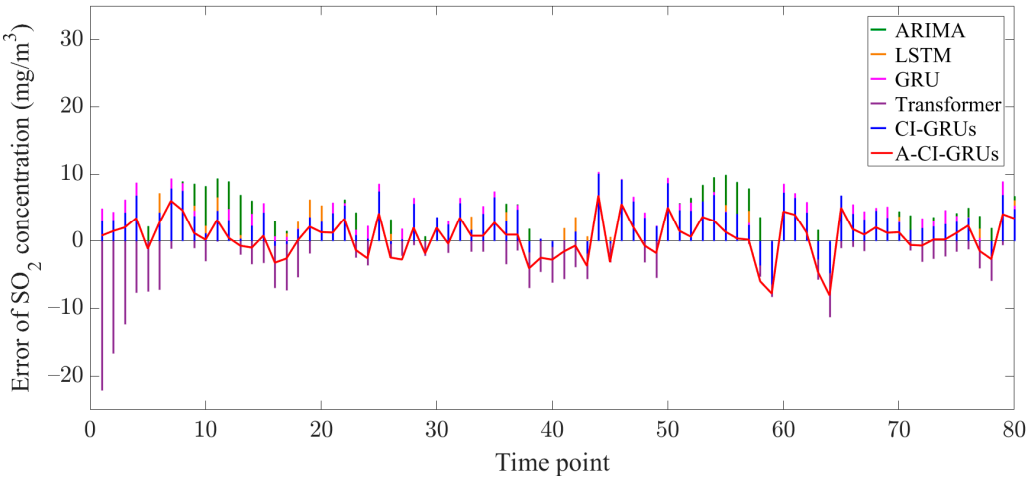


Figure 17. Errors in the prediction methods for the SO_2 concentration.

Table 5. Errors in the prediction result in different steps of the food safety fraction ratio and SO_2 concentration.

		Food Safety Fraction Ratio			SO_2 Concentration		
		5	**10**	**15**	**5**	**10**	**15**
RMSE	ARIMA	7.0300	9.3423	10.3224	4.7846	5.8321	6.1232
	LSTM	8.0841	8.4562	8.9342	5.3669	5.9832	6.1233
	GRU	6.6246	7.3452	7.8923	4.5867	5.3422	5.9653
	Transformer	5.6369	7.0029	8.3610	8.6905	10.2021	10.5355
	CI-GRUs	5.9455	6.2345	7.1234	4.0821	5.3983	5.7834
	A-CI-GRUs	5.4012	5.9839	6.2532	2.8743	3.5683	3.9834
MAE	ARIMA	4.5877	5.8965	6.2312	3.7824	4.5834	6.0123
	LSTM	4.8356	4.9342	4.9834	4.0112	4.2342	5.2736
	GRU	4.2812	5.0745	4.8912	3.7135	4.2453	4.9832
	Transformer	4.4159	5.4351	6.2418	5.7427	7.1292	7.4242
	CI-GRU	3.9939	4.0234	4.1235	3.1694	4.2341	4.0231
	A-CI-GRUs	3.5680	3.9834	4.2371	2.1708	2.9843	3.7532

5. Discussion

A fusion prediction model is proposed considering small samples in some practical systems. The fusion model extracts the features by augmenting the sample and utilizing the

multiple models parallelly. It was tested with simulation and practical data and compared with other relative methods. This can be discussed in the context of the experimental results to analyze the characteristics of the proposed method.

On the one hand, the situation mainly considers that many systems and activities generate data at a low frequency, and historical datasets are often small. Thus, an augmentation method is designed in this paper for the training demands of deep models. The proposed method is compared with CI-GRUs, which lack the augmentation part. For MG and MSO simulation data, the RMSE values of our method are 75.56% and 64.67% of those in the CI-GRUs. Similarly, in the practical prediction of the food fraction ratio and SO_2 concentration, the RMSE decreases by 9.15% and 31.51% compared with the CI-GRUs. The other methods, LSTM, GRU, and transformer, which are trained on the original small sample, perform poorly in accurately fitting the true value. This proves that augmentation can enlarge the sample size with the inherent data feature and can help improve the model's prediction performance.

On the other hand, the proposed method realizes the fusion of multiple methods. Finite samples limit single models such as ARIMA, LSTM, GRU, and transformer. Although the deep models of LSTM, GRU, and transformer can trace the general trend in the experiment, their precision is low in predicting the concrete values of each point. When facing data prediction tasks with more complex trend characteristics, the transformer cannot exert its ability to fit data trends when the number of samples is small. Furthermore, a covariance intersection is introduced to fuse the models instead of using the traditional addition method. In the covariance intersection fusion, the performance of each model is evaluated with the error degree in the validation set. Then, the fusion weight can be obtained quantificationally rather than using the weight of the artificial experience or a random value.

The proposed method guarantees the sample size for deep models. It also realizes reliable predictions by integrating multiple models. In this paper, the type and number of models used in the fusion framework are determined with common sense. Future work can deeply explore how to choose concrete models considering the calculation speed and computing resources. It is expected that the selected model will be as simple as possible, and the number of multiple models should be restricted.

6. Conclusions

An integrated method is studied in this paper to solve the prediction issue in small-scale time series. The overall structure is designed to organize the associated calculation modules, in which the small sample is augmented and predicted with parallel GRUs. The sub-results are fused with the covariance intersection method. The three key components, augmentation, multiple models, and error fusion, realize feature extraction using a limited data size. The method is verified with a classical time series and practical system data. In the future, under the framework of the proposed model, the model type and quantity selection principles can be explored, including using more deep learning models to complete the prediction tasks of the decomposed data to improve prediction accuracy. During data expansion, better retaining the original data characteristics is still a problem worth exploring while also ensuring the data scale required for deep learning model training. In addition, new migration learning strategies can be used to solve the problem of insufficient data for few-shot learning.

Author Contributions: Conceptualization, Y.-L.Z. and Y.-T.B.; methodology, Y.-L.Z. and Y.-T.B.; writing—original draft preparation, Y.-L.Z., Y.-T.B. and W.-Z.Z.; funding acquisition, Y.-T.B., X.-B.J., J.-L.K. and T.-L.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partly supported by the National Natural Science Foundation of China, Nos. 62203020, 62173007, 62006008.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data in the experiment used to support the findings of this study are available from the corresponding authors upon request.

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

References

1. Motroni, A.; Buffi, A.; Nepa, P.; Tellini, B. Sensor-fusion and tracking method for indoor vehicles with low-density UHF-RFID tags. *IEEE Trans. Instrum. Meas.* **2020**, *70*, 8001314. [CrossRef]
2. Lian, S.K.; Meng, W.; Lin, Z.M. Adaptive attitude control of a quadrotor using fast nonsingular terminal sliding mode. *IEEE Trans. Ind. Electron.* **2022**, *69*, 1597–1607. [CrossRef]
3. Mofid, O.; Mobayen, S.; Zhang, C. Desired tracking of delayed quadrotor UAV under model uncertainty and wind disturbance using adaptive super-twisting terminal sliding mode control. *ISA Trans.* **2021**, *123*, 455–471. [CrossRef]
4. Jin, X.B.; Wang, Z.Y.; Kong, J.L.; Bai, Y.T.; Su, T.L.; Ma, H.J.; Chakrabarti, P. Deep spatio-temporal graph network with self-optimization for air quality prediction. *Entropy* **2023**, *25*, 247. [CrossRef]
5. Zhang, H.; Hu, B.; Wang, X. An action dependent heuristic dynamic programming approach for algal bloom prediction with time-varying parameters. *IEEE Access* **2020**, *8*, 26235–26246. [CrossRef]
6. Jin, X.B.; Wang, Z.Y.; Gong, W.T.; Kong, J.L.; Bai, Y.T.; Su, T.L.; Ma, H.J.; Chakrabarti, P. Variational bayesian network with information interpretability filtering for air quality forecasting. *Mathematics* **2023**, *11*, 837. [CrossRef]
7. Hu, H.; Tang, L.; Zhang, S. Predicting the direction of stock markets using optimized neural networks with google trends. *Neurocomputing* **2018**, *285*, 188–195. [CrossRef]
8. Kong, J.L.; Fan, X.M.; Jin, X.B.; Su, T.L.; Bai, Y.T.; Ma, H.J.; Zuo, M. Bmae-net: A data-driven weather prediction network for smart agriculture. *Agronomy* **2023**, *13*, 625. [CrossRef]
9. Lim, B.; Zohren, S. Time-series forecasting with deep learning: A survey. *Philos. Trans. R. Soc. A* **2021**, *379*. [CrossRef]
10. Box, G.; Jenkins, G.M.; MacGregor, J.F. Some recent advances in forecasting and control. *J. R. Stat. Soc. C Appl.* **1974**, *23*, 158–179. [CrossRef]
11. Engle, R.F. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econom. J. Econom. Soc.* **1982**, *50*, 987–1007. [CrossRef]
12. Yang, Y.; Bai, Y.; Wang, X.; Wang, L.; Jin, X.; Sun, Q. Group decision-making support for sustainable governance of algal bloom in urban lakes. *Sustainability* **2020**, *12*, 1494. [CrossRef]
13. Lee, T. Emd and lstm hybrid deep learning model for predicting sunspot number time series with a cyclic pattern. *Sol. Phys.* **2020**, *295*, 6. [CrossRef]
14. Schaffer, A.L.; Dobbins, T.A.; Pearson, S.A. Interrupted time series analysis using autoregressive integrated moving average (ARIMA) models: A guide for evaluating large-scale health interventions. *BMC Med. Res. Methodol.* **2021**, *21*, 58. [CrossRef]
15. Seong, B. Smoothing and forecasting mixed-frequency time series with vector exponential smoothing models. *Econ. Model.* **2020**, *91*, 463–468. [CrossRef]
16. Durbin, J.; Koopman, S.J. *Time Series Analysis by State Space Methods*; Oxford University Press: Oxford, UK, 2010; Volume 47, p. 373.
17. Rogoza, W. Method for the prediction of time series using small sets of experimental samples. *Appl. Math. Comput.* **2019**, *355*, 108–122. [CrossRef]
18. Shi, W.; Wang, Y.; Tang, Y. Forecasting method for water quality time series of few and abnormal data. *J. Comput. Appl.* **2010**, *2*, 486–489. [CrossRef]
19. Rajesh, R. Forecasting supply chain resilience performance using grey prediction. *Electron. Commer. Res. A* **2016**, *20*, 42–58.
20. Deng, Y.; Zhou, X.L.; Shen, J.; Xiao, G.; Hong, H.C.; Lin, H.J.; Wu, F.Y.; Liao, B.Q. New methods based on back propagation (BP) and radial basis function (RBF) artificial neural networks (ANNs) for predicting the occurrence of haloketones in tap water. *Sci. Total Environ.* **2021**, *772*, 145534. [CrossRef]
21. Tsai, C.P.; Lee, T.L. Back-propagation neural network in tidal-level forecasting. *J. Waterw. Port Coast.* **1999**, *125*, 195–202. [CrossRef]
22. Mandal, S. Discussion of back-propagation neural network in tidal-level forecasting. *J. Waterw. Port Coast.* **2001**, *127*, 55. [CrossRef]
23. Cui, M.; Wang, Y.; Lin, X. Fault diagnosis of rolling bearings based on an improved stack autoencoder and support vector machine. *IEEE Sens. J.* **2021**, *21*, 4927–4937. [CrossRef]
24. Xue, L.; Liu, Y.T.; Xiong, Y.F.; Liu, Y.L.; Cui, X.H.; Lei, G. A data-driven shale gas production forecasting method based on the multi-objective random forest regression. *J. Petrol. Sci. Eng.* **2021**, *196*, 107801. [CrossRef]
25. Xu, X.; Yang, C.-C.; Xiao, Y.; Kong, J.-L. A fine-grained recognition neural network with high-order feature maps via graph-based embedding for natural bird diversity conservation. *Int. J. Environ. Res. Public Health* **2023**, *20*, 4924. [CrossRef]
26. Wang, Y.; Tang, H.; Huang, J. A comparative study of different machine learning methods for reservoir landslide displacement prediction. *Eng. Geol.* **2022**, *298*, 106544. [CrossRef]
27. Shu, X.B.; Zhang, L.Y.; Qi, G.J. Spatiotemporal co-attention recurrent neural networks for human-skeleton motion prediction. *IEEE Trans. Pattren Anal.* **2022**, *44*, 3300–3315. [CrossRef]
28. Zheng, H.F.; Lin, F.; Feng, X.X. A hybrid deep learning model with attention-based conv-lstm networks for short-term traffic flow prediction. *IEEE Trans. Intell. Transp.* **2021**, *22*, 6910–6920. [CrossRef]

29. Onan, A. Bidirectional convolutional recurrent neural network architecture with group-wise enhancement mechanism for text sentiment classification. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 2098–2117. [CrossRef]
30. Wei, X.; Zhang, L.L.; Yang, H.Q. Machine learning for pore-water pressure time-series prediction: Application of recurrent neural networks. *Geosci. Front.* **2021**, *12*, 453–467. [CrossRef]
31. Sun, X.; Li, T.; Li, Q.; Huang, Y.; Li, Y. Deep belief echo-state network and its application to time series prediction. *Knowl.-Based Syst.* **2017**, *130*, 17–29. [CrossRef]
32. Xiong, T.; Li, C.; Bao, Y. Seasonal forecasting of agricultural commodity price using a hybrid STL and ELM method: Evidence from the vegetable market in China. *Neurocomputing* **2018**, *275*, 2831–2844. [CrossRef]
33. Liu, D.; Niu, D.; Wang, H.; Fan, L. Short-term wind speed forecasting using wavelet transform and support vector machines optimized by genetic algorithm. *Renew. Energy* **2014**, *62*, 592–597. [CrossRef]
34. Mohammad, Z.K.; Batelaan, O.; Fadaee, M.; Hinkelmann, R. Ensemble machine learning paradigms in hydrology: A review. *J. Hydrol.* **2021**, *598*, 126266.
35. Wang, C.N. A two-stage dea approach to measure operational efficiency in vietnam's port industry. *Mathematics* **2022**, *10*, 1385. [CrossRef]
36. Cleveland, R.B.; Cleveland, W.S.; McRae, J.E. Stl: A seasonal-trend decomposition. *J. Off. Stat.* **1990**, *6*, 3–73.
37. Dagum, B.E.; Luati, A. Global and local statistical properties of fixed-length nonparametric smoothers. *Stat. Method Appl. Ger.* **2002**, *11*, 313–333. [CrossRef]
38. Gupta, M.; Gupta, B. An ensemble model for breast cancer prediction using sequential least squares programming method (SLSQP). In Proceedings of the IEEE Eleventh International Conference on Contemporary Computing, Noida, India, 2–4 August 2018; Volume 1, pp. 1–3.