sustainability

MDPI

Article

A Novel Model Based on DA-RNN Network and Skip Gated Recurrent Neural Network for Periodic Time Series Forecasting

Bingqing Huang [1], Haonan Zheng [2,*], Xinbo Guo [2], Yi Yang [2] and Ximing Liu [3]

[1] School of Science, Rensselaer Polytechnic Institute, New York, NY 12180, USA; huangb5@rpi.edu
[2] School of Information Science & Engineering, Lanzhou University, Lanzhou 730000, China; guoxb18@lzu.edu.cn (X.G.); yy@lzu.edu.cn (Y.Y.)
[3] School of Management, Hefei University of Technology, Hefei 230002, China; ximing_hfut@outlook.com
* Correspondence: zhenghn20@lzu.edu.cn

Abstract: Deep learning models are playing an increasingly important role in time series forecasting with their excellent predictive ability and the convenience of not requiring complex feature engineering. However, the existing deep learning models still have shortcomings in dealing with periodic and long-distance dependent sequences, which lead to unsatisfactory forecasting performance on this type of dataset. To handle these two issues better, this paper proposes a novel periodic time series forecasting model based on DA-RNN, called DA-SKIP. Using the idea of task decomposition, the novel model, based on DA-RNN, GRU-SKIP and autoregressive component, breaks down the prediction of periodic time series into three parts: linear forecasting, nonlinear forecasting and periodic forecasting. The results of the experiments on Solar Energy, Electricity Consumption and Air Quality datasets show that the proposed model outperforms the three comparison models in capturing periodicity and long-distance dependence features of sequences.

Keywords: time series; prediction; deep learning; attention; recurrent neural network

Citation: Huang, B.; Zheng, H.; Guo, X.; Yang, Y.; Liu, X. A Novel Model Based on DA-RNN Network and Skip Gated Recurrent Neural Network for Periodic Time Series Forecasting. *Sustainability* **2022**, *14*, 326. https://doi.org/10.3390/su14010326

Academic Editors: Zhengxin Wang, Song Ding, Xin Ma and Wendong Yang

Received: 5 December 2021
Accepted: 24 December 2021
Published: 29 December 2021

1. Introduction

Time series forecasting can be summarized as a process of extracting useful information from historical records and then forecasting the future value [1]. It has shown great application value in stock trend forecasting [2], traffic flow forecasting [3], power generation [4], electricity consumption forecasting [5], tourism passenger flow forecasting [6], weather forecasting [7] and other fields. Among the problems in time series forecasting, the biggest problem faced by existing models is capturing the long-distance dependence in sequences. From autoregressive models [8] to recurrent neural networks [9], researchers have been trying to improve the model's prediction performance of long-distance dependent sequences. Furthermore, the periodicity of time series is also an important factor that is worth considering. Traditional time series forecasting models often are not able to achieve the best results on periodic time series datasets [10]. If the periodicity of time series is taken into consideration in optimizing the model, the applicability of the model can be improved such that it can achieve better performance on this type of dataset.

The research methods of time series forecasting have been continuously improving and innovating since the 1970s. Time series forecasting models can be roughly divided into three categories. The first category is time series forecasting methods based on statistical models, such as the Markov model [11] and the autoregressive moving average model (ARIMA) [12]; the second category is time series forecasting methods based on machine learning, such as many methods based on Bayesian network or support vector machine method [13,14]; the third category is time prediction method based on deep learning, such as artificial neural network (ANN) [15], Long Short-Term Memory (LSTM) [16] and Gate Recurrent Unit (GRU) [17], etc.

With the breakthrough in the research of deep learning, deep learning has been playing an increasingly important role in time series forecasting in recent years. In particular, the application of LSTM and GRU has made outstanding contributions to solving the long-distance dependence problem in time series forecasting. Since their introduction, these two methods have achieved great success in time series forecasting [18], time series classification [19], natural language processing [20], machine translation [21], speech recognition [22] and other fields. In recent years, the encoder/decoder network [23] and the attention-based encoder/decoder network [24] have further improved the computational efficiency and prediction accuracy of time series prediction models.

The R2N2 model introduced by Hardik Goel et al. in 2017 [25] decomposes the time series forecasting task into a linear forecasting part and a non-linear forecasting part. The linear forecasting part uses an autoregressive component, and the non-linear part uses an LSTM network for prediction. In 2017, the LSTNet model designed by Guokun Lai et al. [26] embodied the idea of specialized processing for periodic time series data. The model divides the periodic time series forecasting task into a linear forecasting part, a non-linear forecasting part and a periodic forecasting part. The linear forecasting part is composed of an autoregressive model, the non-linear forecasting part is composed of a LSTM network, and the periodic forecasting part is composed of a GRU network. The TPA-LSTM model proposed by Shun-Yao Shih et al. in 2018 [27] introduced the attention mechanism into time series prediction and proposed an attention mechanism in the direction of multivariate. Compared to previous attention models in the dimension of time step, this model has achieved better results on some datasets.

In 2017, based on LSTM and attention mechanism, Yao Qin et al. proposed the DA-RNN network [28]. DA-RNN is a kind of the non-linear autoregressive exogenous (NARX) model [29,30] which means that the data processed by the model has exogenous variables and contains nonlinear relationships inside. This type of model can predict the current value of a time series based on the previous value of the target series and the driving (exogenous) series. Making full use of the information contained in the target series and driving series is the advantage of this type of model [31]. On this basis, the DA-RNN model focuses on processing multivariate series and resolving long-distance dependence problems.

The DA-RNN model comprises two components: encoder and decoder. It is a novel two-stage recurrent neural network based on attention mechanism. In the encoder, the model introduces a new input attention mechanism, which makes it adaptively focus on related driving series and weight them. In the decoder, the model introduces a temporal attention mechanism to adaptively focus on the output of the encoder across all time steps. With the help of this design, the DA-RNN model achieved excellent performance in the test of several multivariate datasets. However, when dealing with periodicity and autocorrelation sequences, DA-RNN is difficult to achieve the best results.

To solve the long-distance dependence problem and sequence periodicity problem in time series forecasting better, this paper introduces the periodic gated recurrent network component (GRU-SKIP) and autoregressive component into the DA-RNN model to construct a new model called DA-SKIP that is more suitable for periodic time series datasets.

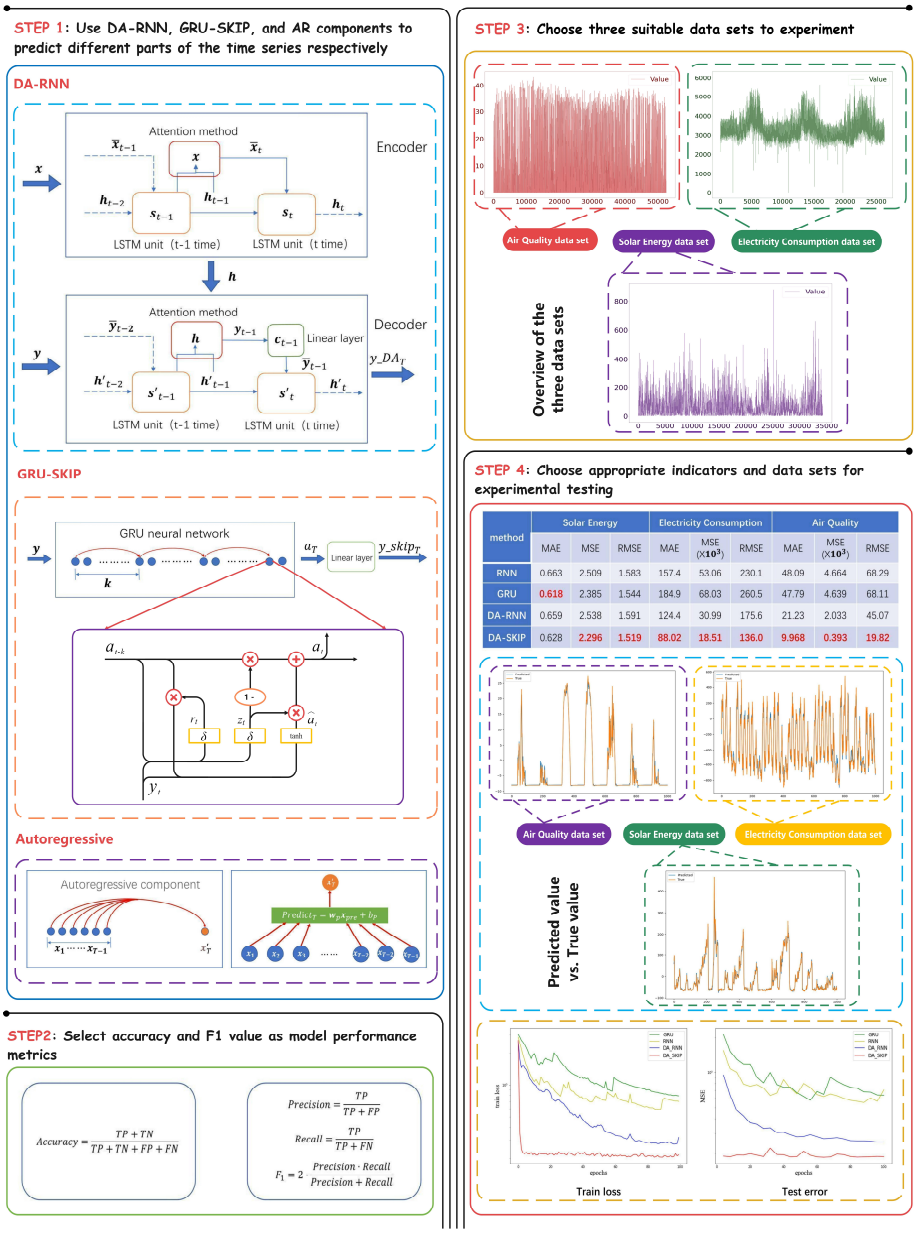
The DA-SKIP model combines the multivariate sequence processing and long-distance dependency processing capabilities of the DA-RNN model and the periodic data processing capabilities brought by the GRU-SKIP component. In the processing of periodic datasets, the non-linear law of the data can be captured nicely by the DA-RNN component, the periodic law of the data can be captured by the GRU-SKIP component, and the linear law of the data can be captured by the autoregressive component. The final test shows that on the periodic dataset, the DA-SKIP model performs significantly better than the RNN model, GRU model and DA-RNN model.

The innovations of this paper are as follows:

(1) The proposed model breaks down the prediction problems of periodic time series into three parts: linear forecasting, nonlinear forecasting and periodic forecasting, and uses three different model components to complete each prediction subtask.

(2) The characteristics of the DA-RNN components are used in the model to effectively solve the long-distance dependence problem in time series forecasting.
(3) The characteristics of the DA-SKIP components are used in the model to effectively solve the cyclical problem in time series forecasting.
(4) The characteristics of autoregressive components are used in the model to effectively solve the linear correlation problem in time series forecasting.

This paper is organized as follows. First, Section 2 introduces the structure of each component of the model. Next, Section 3 presents the datasets, comparison models and evaluation metrics used in the experiment. Then, Section 4 discusses some scientific problems that appeared in the experiment. Finally, Section 5 summarizes the findings and discusses the future research direction. The main content of the paper is shown in Figure 1.



method	Solar Energy			Electricity Consumption			Air Quality		
	MAE	MSE	RMSE	MAE	MSE ($\times 10^3$)	RMSE	MAE	MSE ($\times 10^3$)	RMSE
RNN	0.663	2.509	1.583	157.4	53.06	230.1	48.09	4.664	68.29
GRU	0.618	2.385	1.544	184.9	68.03	260.5	47.79	4.639	68.11
DA-RNN	0.659	2.538	1.591	124.4	30.99	175.6	21.23	2.033	45.07
DA-SKIP	0.628	2.296	1.519	88.02	18.51	136.0	9.968	0.393	19.82

Figure 1. The overall structure of the paper.

2. Materials and Methods

Generally, the forecasting task of periodic time series data can be divided into three parts: the prediction of non-linear, linear and periodic laws. Three components of our

proposed model correspond to these three parts: the DA-RNN encoder/decoder component is used to predict non-linear law, autoregressive component is used to predict linear law, and GRU-SKIP components is used to predict periodic law.

2.1. Model Task

The overall prediction task of the model is to use a multivariate driving series $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{T-1}\}$ and a target series $\boldsymbol{y} = \{y_1, y_2, \cdots, y_{T-1}\}$, to predict the target value y_T at time T. Where $\boldsymbol{x}_t = \{x_t^1, x_t^2, \cdots x_t^n\}$ $(1 \leq t \leq T-1)$ and $\boldsymbol{x}_t \in \mathbb{R}^n$, n is the number of variables in the input sequence, and T is the length of the input multivariate driving series and target series.

Taking the forecast of people flow in a scenic spot as an example, the multivariate driving series refers to the sequence data related to the people flow, such as the climate, temperature and air quality of the scenic spot. The target series refers to the historical data of the flow of people in the scenic spot. As shown in Figure 2, the task can be summarized as extracting information from k driving series and a target series, using data from time 1 to time $T-1$ to predict the flow of people y_T at the scenic spot at time T.

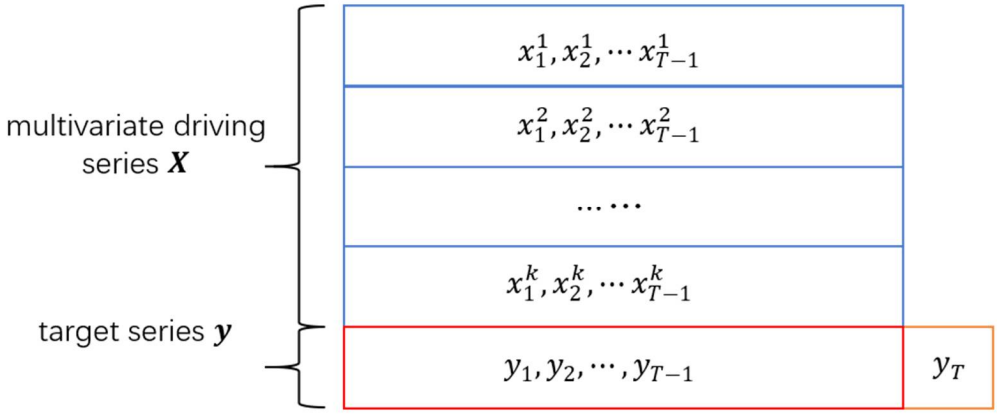


Figure 2. The data structure of multivariate time series forecasting.

2.2. Encoder Component

The overall structure of the model encoder and decoder component is shown in Figure 3. Among them, the encoder component of the model is broadly the same as the encoder component in DA-RNN. With the support of attention mechanism, the encoder can realize the function of weighting the input multivariate driving series, thereby capturing the correlation between different variables in the multivariate series.

The input series of the encoder is the multivariate drive series $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{T-1}\}$, where $\boldsymbol{x}_t \in \mathbb{R}^n$ $(1 \leq t \leq T-1)$, n is the number of variables in the input sequence. In the encoder, the model uses an LSTM network to map the driving series $\boldsymbol{x}_t$ at time t to the hidden state $\boldsymbol{h}_t$:

$$\boldsymbol{h}_t = f_1(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t) \tag{1}$$

f_1 is an LSTM unit, whose input is the hidden state at time $t-1$ and the driving series $\boldsymbol{x}_t$ at time t and outputs the hidden state at time t $(1 \leq t \leq T-1)$ as calculation result. The advantage of LSTM is that it does well in capturing long-distance dependency. Every time step of LSTM has a cell state $\boldsymbol{s}_t$, and each $\boldsymbol{s}_t$ is controlled by three sigmoid gating components. The three gates are respectively the forget gate $\boldsymbol{f}_t$, the update gate $\boldsymbol{u}_t$ and the output gate $\boldsymbol{o}_t$. The specific calculation formula is as follows:

$$\boldsymbol{s}_t = \boldsymbol{f}_t \odot \boldsymbol{s}_{t-1} + \boldsymbol{u}_t \odot tanh(\boldsymbol{W}_s[\boldsymbol{h}_{t-1}; \boldsymbol{x}_t] + \boldsymbol{b}_s) \tag{2}$$

$$\text{where } \boldsymbol{f}_t = \delta\left(\boldsymbol{W}_f[\boldsymbol{h}_{t-1}; \boldsymbol{x}_t] + \boldsymbol{b}_f\right) \tag{3}$$

$$u_t = \delta(W_u[h_{t-1}; x_t] + b_u) \tag{4}$$

$$o_t = \delta(W_o[h_{t-1}; x_t] + b_o) \tag{5}$$

$$h_t = o_t \odot tanh(s_t) \tag{6}$$

where $[h_{t-1} : x_t]$ is the matrix formed by concatenating the hidden state h_{t-1} at the previous moment and the input x_t at the current moment. W_u, W_f, W_o, W_s are the weight matrices that need to be learned and b_u, b_f, b_o, b_s are the bias terms that need to be learned. δ and $\odot$ are the sigmoid function symbol and the dot product symbol, respectively. LSTM here makes the model less prone to the problem of gradient disappearance and brings strong capability to capture long-distance dependency to the model.

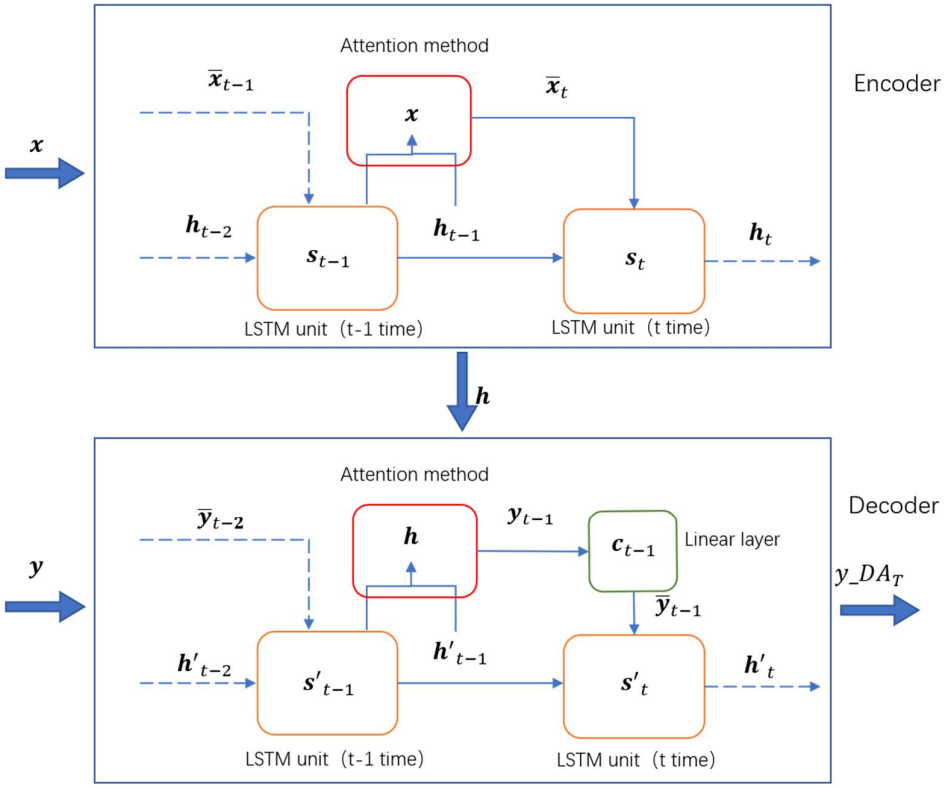


Figure 3. The overall structure of the encoder and decoder.

After inputting the sequence into the LSTM network, the hidden state h_t and the cell state s_t in the LSTM network at time t can be calculated. For each variable $X^k = \left\{x_1^k, x_2^k, \cdots x_{T-1}^k\right\}$ in the multivariate driving series, the model uses an attention component to associate it with the matrix $[h_{t-1}; s_{t-1}]$ at the previous moment, that is, at time $t-1$, and capture the connection between them:

$$e_t^k = V_e^\top tanh\left(W_e[h_{t-1}; s_{t-1}] + U_e x^k\right) \tag{7}$$

$$\alpha_t^k = \frac{exp\left(e_t^k\right)}{\sum_{i=1}^{n} exp\left(e_t^i\right)} \tag{8}$$

where V_e, W_e and U_e are the parameters needed to be learned. The model uses the attention mechanism in Formulas (7) and (8) to capture the association among the hidden state, the cell state and each variable, and the weight α_t^k of each variable at time t can be calculated using the softmax formula. Then, with these attention weights provided, the driving series at time t can be weighted: $\bar{x}_t = \left(\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, \cdots, \alpha_t^n x_t^n\right)^\top$. These weights make the model

focus on some crucial sequences and selectively ignore less important sequences. This mechanism helps the model to make better use of multivariate data.

The model weights the driving series $\boldsymbol{x}_t$ at time t through the hidden state $\boldsymbol{h}_{t-1}$ and cell state $\boldsymbol{s}_{t-1}$ at time $t-1$, and then replace the initial $\boldsymbol{x}_t$ with the weighted sequence $\overline{\boldsymbol{x}}_t$ in the calculation of the hidden state $\boldsymbol{h}_t$ at time t. At this time, Formula (1) should be amended to:

$$\boldsymbol{h}_t = f_1(\boldsymbol{h}_{t-1}, \overline{\boldsymbol{x}}_t) \tag{9}$$

where f_1 is an LSTM unit, and $\overline{\boldsymbol{x}}_t$ is a weighted multivariate sequence. The model map $\overline{\boldsymbol{x}}_t$ to the hidden state $\boldsymbol{h}_t$ via LSTM, and finally connect the $\boldsymbol{h}_t$ at each moment as $\boldsymbol{h} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_T\}$ as the output of the encoder and input it to the decoder.

2.3. Decoder Component

The input of the decoder is divided into two parts. The first part is the hidden state of the encoder at each moment $\boldsymbol{h} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_T\}$, and the second part is the target series $\boldsymbol{y} = \{y_1, y_2, \cdots, y_{T-1}\}$. In the decoder, the model firstly uses a LSTM network to decode the input sequence. The LSTM network takes the target series $\boldsymbol{y}$ as input, and the hidden state and cell state at time t are represented by $\boldsymbol{d}_t$ and $\boldsymbol{s}'_t$ $(1 \le t \le T-1)$, respectively.

To solve the long-distance dependence problem, a time attention mechanism is applied in the decoder to make the model adaptively focus on the important time steps in the hidden state time series. Specifically, the model connects the hidden state $\boldsymbol{h}'_{t-1}$ of the LSTM network in the decoder at $t-1$ with the cell state $\boldsymbol{s}'_{t-1}$ at the same moment to form the matrix $[\boldsymbol{h}'_{t-1}; \boldsymbol{s}'_{t-1}]$. Then, a temporal attention mechanism is used to capture the correlation between the $[\boldsymbol{h}'_{t-1}; \boldsymbol{s}'_{t-1}]$ matrix and the hidden state of the encoder at each moment. The attention weight of each hidden state in the encoder can be calculated at this time:

$$l_t^i = \boldsymbol{V}_h^\top tanh\left(\boldsymbol{W}_{\boldsymbol{h}}\left[\boldsymbol{h}'_{t-1}; s'_{t-1}\right] + \boldsymbol{U}_h \boldsymbol{h}_i\right) \tag{10}$$

$$\beta_t^i = \frac{exp\left(l_t^i\right)}{\sum_{j=1}^{T} exp\left(l_t^j\right)} \tag{11}$$

where $\boldsymbol{W}_h, \boldsymbol{V}_h, \boldsymbol{U}_h$ are the parameters that need to be learned. $\boldsymbol{h}_i$ represents the i-th hidden state in the encoder, and β_t^i represents the weight of $\boldsymbol{h}_i$. By calculating the weight of each moment, the hidden state from the encoder can be weighted at each moment:

$$\boldsymbol{c}_t = \sum_{i=1}^{T} \beta_t^i \boldsymbol{h}_i \tag{12}$$

$\boldsymbol{c}_t$ is called context vector, which is obtained by weighting all hidden state $\boldsymbol{h} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_T\}$ in encoder. Then, the model combines the context vector $\boldsymbol{c}_t$ with the given target series $\boldsymbol{y} = \{y_1, y_2, \cdots, y_{t-1}\}$:

$$\overline{y}_{t-1} = \overline{\boldsymbol{w}}^\top [y_{t-1}; \boldsymbol{c}_{t-1}] + \overline{b} \tag{13}$$

where $[y_{t-1}; \boldsymbol{c}_{t-1}]$ is the concatenation of the target value y_{t-1} and the context vector $\boldsymbol{c}_{t-1}$ at time $t-1$. $\overline{\boldsymbol{w}}$ and $\overline{b}$ are the parameters need to be learned, their role is to reduce the dimensionality of the concatenation vector to a constant. The calculated new target value $\overline{y}_{t-1}$ is used to replace the input y_{t-1} of the decoder LSTM at time t. The modified decoder LSTM operation formula is:

$$\boldsymbol{s}'_t = \boldsymbol{f}'_t \odot \boldsymbol{s}'_{t-1} + \boldsymbol{u}'_t \odot tanh\left(\boldsymbol{W}'_s\left[\boldsymbol{h}'_{t-1}; \overline{y}_{t-1}\right] + \boldsymbol{b}'_s\right) \tag{14}$$

$$\text{where } \boldsymbol{f}'_t = \delta\left(\boldsymbol{W}'_f\left[\boldsymbol{h}'_{t-1}; \overline{y}_{t-1}\right] + \boldsymbol{b}'_f\right) \tag{15}$$

$$\boldsymbol{u}'_t = \delta\left(\boldsymbol{W}'_u\left[\boldsymbol{h}'_{t-1}; \overline{y}_{t-1}\right] + \boldsymbol{b}'_u\right) \tag{16}$$

$$o'_t = \delta\left(W'_o\left[h'_{t-1}; \overline{y}_{t-1}\right] + b'_o\right) \tag{17}$$

$$h'_t = o'_t \odot tanh\left(s'_t\right) \tag{18}$$

where $\left[h'_{t-1}; \overline{y}_{t-1}\right]$ is the connection of hidden state h'_{t-1} and the corrected input $\overline{y}_{t-1}$ at $t-1$. W'_u, W'_f, W'_s, W'_o and b'_u, b'_f, b'_s, b'_o are the parameters that need to be learned. δ and $\odot$ are respectively the sigmoid function and the dot multiplication operation. The final prediction result can be expressed as:

$$y\_DA_T = V_y^{\top}\left(W_y\left[h'_T; c_T\right] + b_w\right) + b_v \tag{19}$$

where $\left[h'_T; c_T\right]$ is the concatenation of the hidden state h'_T of the decoder at time t and the context vector. The parameters W_y and b_w adjust the size of concatenation matrix to be the same as the size of hidden state in the decoder. Then, the calculation result is sent into the linear layer whose weight matrix is v_y and bias is b_v to generate the decoder's final prediction value y_DA_T.

2.4. GRU-SKIP Component

The role of the GRU-SKIP component in the model is to capture the periodicity of the series such that the model performs better in periodic time series datasets. The overall structure of GRU-SKIP components is shown in Figure 4. The model takes the period length k of the sequence as the length of time step and extract the jumping sequence $p = \left\{y_{T-k\times m}, y_{T-k\times(m-1)}, \cdots, y_{T-k}\right\}$ $(k \times m \leq T)$ of length m in the target series $y = \{y_1, y_2, \cdots, y_{T-1}\}$. For the jumping sequence p, the model uses the GRU network to extract its periodic trend.

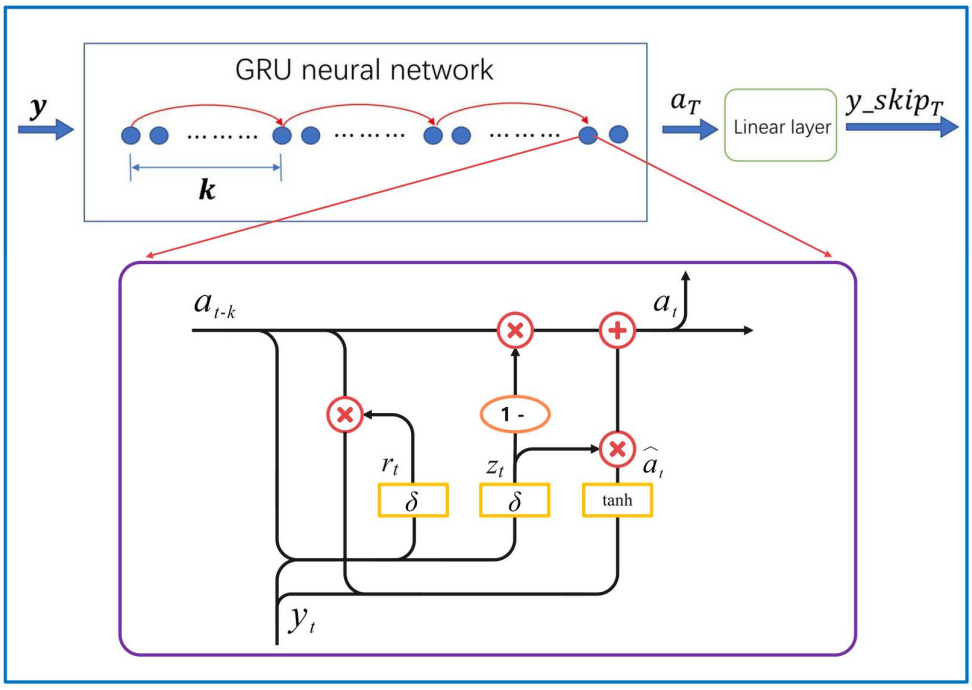


Figure 4. The overall structure of GRU-SKIP components.

Similar to the LSTM, the data at each time step in the GRU network is also input into a gated recurrent unit, and each unit is controlled by two gates: update gate z_t and reset gate r_t. The detailed calculation formula is as follows:

$$\hat{a}_t = tanh(W_a[r_t \odot a_{t-k}; y_t] + b_a) \tag{20}$$

$$\text{where } z_t = \delta(W_z[a_{t-k}; y_t] + b_z) \tag{21}$$

$$r_t = \delta(W_r[a_{t-k}; y_t] + b_r) \tag{22}$$

$$a_t = z_t \odot \hat{a}_t + (1 - u_t) \odot a_{t-k} \tag{23}$$

where δ and $\odot$ are respectively the sigmoid function and dot multiplication operation. k is the period length of the time series. $[a_{t-k}; y_t]$ $(1 \leq t \leq T-1)$ is the concatenation of the hidden state a_{t-k} at time $t-k$ and the input y_t at time t. W_a, W_z, W_r and b_a, b_z, b_r are all parameters that need to be learned.

The width of the hidden state at time t is equal to the hidden layer width h_a of the GRU_SKIP component. The model inputs the hidden state a_T at time T into a linear layer and reduce its width to 1, and then the periodic prediction value y_skip_T at time T can be calculated:

$$y\_skip_T = W_j a_T + b_j \tag{24}$$

where W_j and b_j are the weight matrix and bias term in the linear layer, respectively.

In addition to the core part of the GRU-SKIP component, an autoregressive component can be optionally added for predicting the linear part of the data. The autoregressive model can predict the sequence value at a specific time in the future based on the sequence information in the previous period. However, this prediction is limited to the case where there is autocorrelation in the sequence. Thus, autoregressive components are often used to extract linear relationships in the autocorrelation sequence.

The purpose of adding autoregressive components to the model is to enhance the prediction effect of autocorrelation sequences. The operation of the autoregressive component can be regarded as a hyperparameter, which can be selectively added during the tuning process according to the specific performance of the model. If an autoregressive component is added, the output of the GRU-SKIP component should be replaced with:

$$y\_skip_T = W_j a_T + b_j + W_i \boldsymbol{y} + b_i \tag{25}$$

The autoregressive component is implemented by a linear layer, where W_i is the weight matrix, b_i is the bias term, and $\boldsymbol{y}$ is the target series. The prediction target y_pred_T at time T can be divided into three parts: periodic part, linear-part and non-linear part. The output of the decoder y_DA_T is the forecast of the non-linear part, and the output of the GRU-SKIP component y_skip_T is the forecast of the periodic part and the linear part. So, the final prediction value y_pred_T is the sum of y_DA_T and y_pred_T:

$$y\_pred_T = y\_DA_T + y\_skip_T \tag{26}$$

y_pred_T is the final output of the DA-SKIP model. The overall architecture of DA-skip is shown in Figure 5.

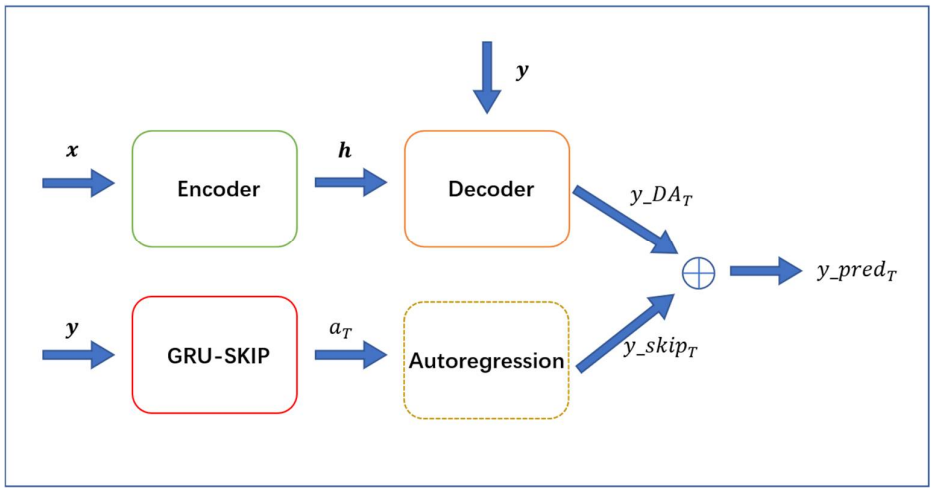


Figure 5. The overall structure of DA-SKIP.

3. Experiments

To test the actual performance of DA-SKIP, it was tested on three datasets and compared with the RNN model, the GRU model and the DA-RNN model. In the test, hyperparameter grid search method was used to adjust each model's hyperparameter, and then the model ran five times under the optimal parameter combination. Finally, the average value of each evaluation metric in these five tests was taken as the test result of the model. Equipment used in experiment can be find in Appendix A.

3.1. Datasets

Three datasets of Solar Energy, Electricity consumption and Air Quality were used in this experiment. The Solar Energy dataset recorded the power generation of 137 photovoltaic power stations in Alabama, the USA in 2006. Data in this dataset was collected every 15 min [32]. In the experiment, the first 136 rows of data were set as driving series input, and the last row of data was set as target series input. The Electricity consumption dataset recorded the electricity consumption of 321 corporate users in the United States from 2011 to 2014. Data in this dataset was collected every 10 min [33]. In the experiment, the first 320 rows of data were set as driving series input, and the last row of data was set as target series input. The Air Quality dataset recorded 18 indicators of Beijing's air quality from 2013 to 2017. Data in this dataset was collected every hour [34]. In the experiment, the first indicator was set as target series input, and the other data were set as driving series input. In all three datasets, the first 70% of the data was set as the training set and the last 30% of the data was set as the test set. An overview of the three experimental datasets is shown in Table 1.

Table 1. Overview of 3 experimental datasets.

Dataset	Driving Series	Train Size	Test Size
Solar Energy	136	32,473	15,768
Electricity Consumption	320	15,533	7892
Air Quality	17	23,284	10,287

The training is conducted as the following process: first, the best hyperparameter combination in the test is determined by hyperparameter gradient search. After that, the test is repeated five times under this hyperparameter, and the average of the five test results is used as the final test result.

In all experiments on three datasets, DA-SKIP is trained for 100 rounds, during which the learning rate drops by 10% every 10 rounds of training, while the initial value of learning rate is different: for the Solar Energy dataset its 0.0004, for the Electricity Consumption dataset its 0.08, for the Air Quality dataset its 0.0005. In the experiment, the sequence length corresponding to one day is used as the period length of the GRU-SKIP component.

3.2. Methods for Comparison

In the experiment, RNN, GRU and DA-RNN were selected as comparison models. DA-SKIP model and the three comparison models were trained in three experimental datasets. Finally, the performance of each model in the test sets was used to compare their prediction capabilities.

3.3. Evaluation Metrics

We choose the mean square error MSE, absolute average error MAE and root mean square error RMSE to measure the model's performance on the dataset. The formulas of these three indicators are as follows:

$$\mathrm{MAE} = \frac{1}{m}\sum_{i=1}^{m}\left|\left(y_i - \hat{y}_i\right)\right| \tag{27}$$

$$\mathrm{MSE} = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{28}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2} \tag{29}$$

where y_i is the true value of the time series at time i, $\hat{y}_i$ is the predicted value of the model at time i, and m is the length of the test set.

3.4. Results

The test results of each model on the three datasets are shown in Table 2.

Table 2. Test results of RNN, GRU, DA-RNN, and DA-SKIP models on three datasets. The MSE of the Electricity Consumption and Air Quality datasets are in units of 10^3 .

Method	Solar Energy			Electricity Consumption			Air Quality		
	MAE	**MSE**	**RMSE**	**MAE**	**MSE**	**RMSE**	**MAE**	**MSE**	**RMSE**
RNN	0.663	2.509	1.583	157.4	53.06	230.1	48.09	4.664	68.29
GRU	**0.618**	2.385	1.544	184.9	68.03	260.5	47.79	4.639	68.11
DA-RNN	0.659	2.538	1.591	124.4	30.99	175.6	21.23	2.033	45.07
DA-SKIP	0.628	**2.296**	**1.519**	**88.02**	**18.51**	**136.0**	**9.968**	**0.393**	**19.82**

Table 2 clearly shows that the test results of DA-SKIP on the three datasets are better than DA-RNN, GRU and RNN in most indicators. DA-SKIP achieved the best performance in eight out of nine indicators in the three datasets. On the Electricity Consumption dataset and Air Quality dataset, DA-SKIP has the most significant advantage that it surpasses the second place by 22.55% to 80.66% in all indicators.

DA-SKIP outperforms GRU and RNN mainly because of the advantages in handling long-distance dependence and making full use of external driving series, while DA-SKIP outperforms DA-RNN mainly because of the excellent periodicity forecasting ability of GRU-SKIP components.

On the Electricity Consumption dataset and Air Quality dataset, DA-SKIP has significant advantages, but when it comes to the Solar Energy dataset, DA-SKIP has relatively small advantages. This may be because the data in the Electricity Consumption dataset and the Air Quality dataset show relatively more obvious autocorrelation. As we presented above, DA-SKIP can capture not only the periodicity of data but also the autocorrelation of data by adding autoregressive component. Considering this, that's the possible reason why DA-SKIP performed significantly better than the comparison model. In the experiment on these two datasets, we found that once the autoregressive component of DA-SKIP is disabled, the advantage of DA-SKIP over other comparison models will reduce. This phenomenon supports the statement from another aspect and also proves the effectiveness of the autoregressive component in the model.

To explore the training efficiency of each model, Figure 6 is plotted to record the change trend of the training loss during the training of the four models on the Electricity Consumption dataset. In the experiment, the model is tested on the test set every four epochs of training. The right part of Figure 6 records the change of the MSE value of each model on the Electricity Consumption test set.

The left part of Figure 6 clearly shows that compared to the other three comparison models; DA-SKIP's training loss can quickly converge to a smaller value during the training process. The same trend can be seen on the MSE value when testing on the test set. The right part of Figure 6 proves that the MSE value of DA-SKIP model on the test set, shares the same rapid convergence trend as the training loss, and finally it also stays stable at a lower point than the other three models. These prove that DA-SKIP is significantly better than the comparison model in terms of training efficiency and convergence speed while ensuring the accuracy of prediction.

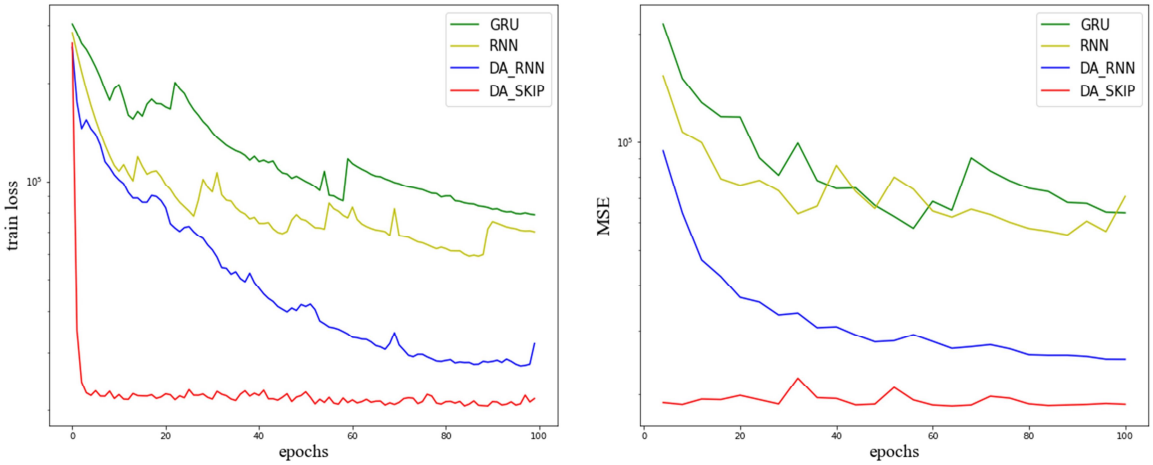


Figure 6. The change trend of the train loss and the MSE value on the test set with the number of training epochs on the Electricity Consumption dataset.

The above experimental results show that the introduction of task segmentation and integrated model ideas brings stronger long-distance prediction capabilities and periodic prediction capabilities to the model. It illustrates the advantages of DA-SKIP in dealing with periodic time series over the comparison models. The final prediction results of the DA-skip model on the three datasets are shown in Figures 7–9.

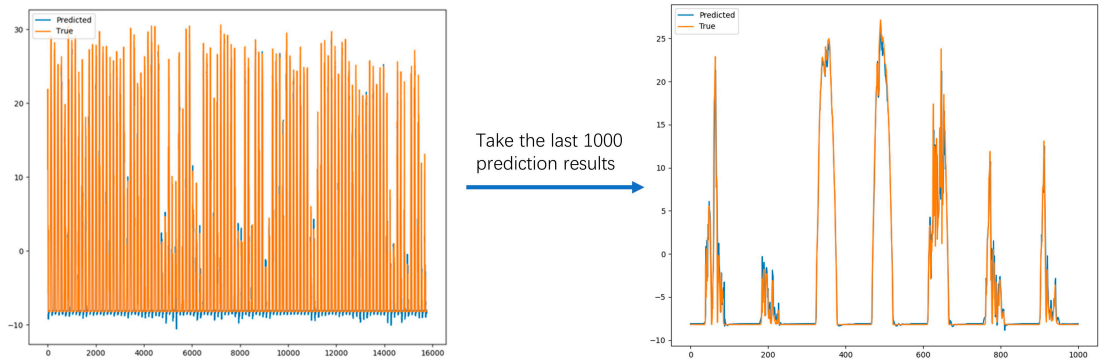


Figure 7. The prediction results of DA-SKIP on the Solar Energy dataset. The yellow line in the figure represents the true value, and the blue line represents the predicted value.

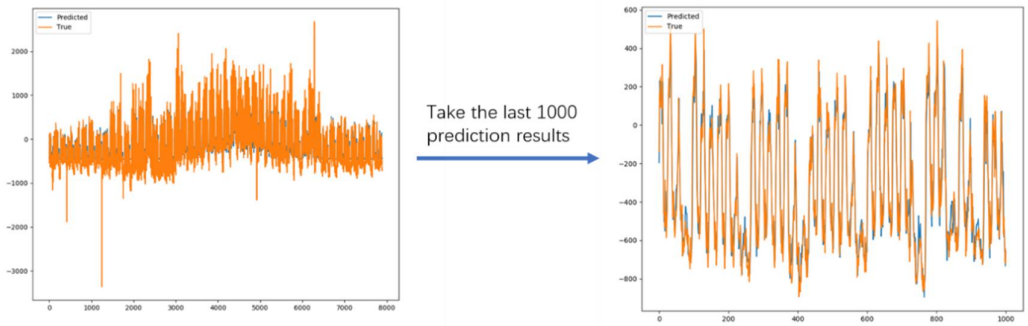


Figure 8. The prediction results of DA-SKIP on the Electricity Consumption dataset. The yellow line in the figure represents the true value, and the blue line represents the predicted value.

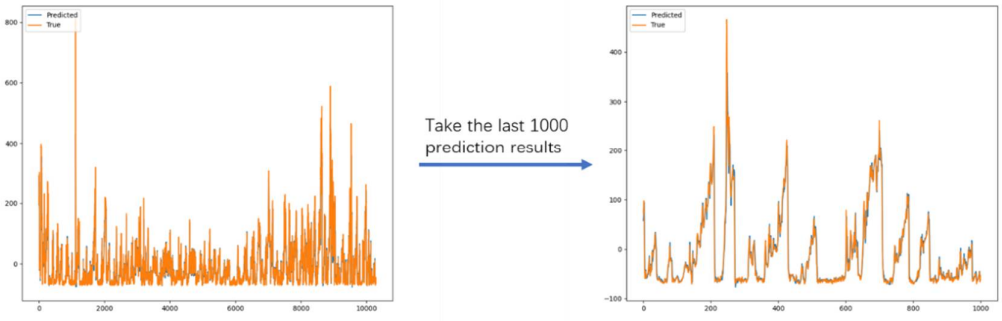


Figure 9. The prediction results of DA-SKIP on the Air Quality dataset. The yellow line in the figure represents the true value, and the blue line represents the predicted value.

4. Discussion

The result of the experiment proves that the time series prediction accuracy of the DA-SKIP model is significantly improved compared with the existing model, and the degree of improvement is related to the characteristics of the dataset itself. We guess that DA-SKIP model will have a better prediction effect for datasets with strong periodicity and autocorrelation. This conjecture has not been fully verified, and we will collect data on more datasets for in-depth research.

5. Conclusions

The DA-SKIP model designed in this paper is based on the DA-RNN model, and it is optimized for periodic datasets. In this model, the DA-RNN-based encoder/decoder component is used to capture the non-linear law of sequence data, the GRU-SKIP component is used to capture the periodic law of sequence data, and the autoregressive component is used to capture the linear law of sequence data.

DA-SKIP inherits DA-RNN's excellent processing capabilities for multivariate data and long-distance dependence. At the same time, the introduction of GRU-SKIP components enhances the model's processing capabilities for periodic sequences, the use of autoregressive components enhances the model's processing capabilities for autocorrelation sequences. After that, excellent performance was seen on the three datasets of Solar Energy, Electricity Consumption and Air Quality.

The model proposed in this paper is suitable for datasets with clear periodicity and known period length, such as photovoltaic power generation, urban electricity consumption, road traffic flow, tourist flow in scenic spots, and so on. The model is proposed for these kinds of practical problems; therefore, it has a wide range of application prospects in reality. However, the demand for a clear period length also limits the scope of application of our model to some extent. In future research, we can try to use the attention mechanism to adaptively extract the periodicity and period length in order to further expand the application range of the model.

Author Contributions: Conceptualization, H.Z.; methodology, H.Z. and B.H.; software, X.L.; validation, Y.Y. and X.G.; formal analysis, B.H.; investigation, X.L.; resources, Y.Y.; data curation, H.Z.; writing—original draft preparation, H.Z. and B.H.; writing—review and editing, Y.Y.; visualization, X.G.; supervision, H.Z.; project administration, B.H.; funding acquisition, Y.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key R&D Program of China, grant number 2018YFB1003205 and the Natural Science Foundation of Gansu Province, China, grant number 20JR10RA182.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Appendix A

Experiment apparatus: All experiments have been carried out through pytorch 1.7 on a PC equipped with Windows 10 64-bit, Inter Core i7-10700 CPU, 16GB RAM, GeForce RTX 2080 Ti GPU.

References

1. Hua, Y.; Zhao, Z.; Li, R.; Chen, X.; Liu, Z.; Zhang, H. Deep learning with long short-term memory for time series prediction. *IEEE Commun. Mag.* **2019**, *57*, 114–119. [CrossRef]
2. Yadav, A.; Jha, C.K.; Sharan, A. Optimizing LSTM for time series prediction in Indian stock market. *Procedia Comput. Sci.* **2020**, *167*, 2091–2100. [CrossRef]
3. Li, Y.; Huang, J.; Chen, H. Time series prediction of wireless network traffic flow based on wavelet analysis and BP neural network. *J. Phys. Conf. Ser. IOP Publ.* **2020**, *1533*, 032098. [CrossRef]
4. Sharadga, H.; Hajimirza, S.; Balog, R.S. Time series forecasting of solar power generation for large-scale photovoltaic plants. *Renew. Energy* **2020**, *150*, 797–807. [CrossRef]
5. Jallal, M.A.; Gonzalez-Vidal, A.; Skarmeta, A.F.; Chabaa, S.; Zeroual, A. A hybrid neuro-fuzzy inference system-based algorithm for time series forecasting applied to energy consumption prediction. *Appl. Energy* **2020**, *268*, 114977. [CrossRef]
6. Kaytez, F.; Taplamacioglu, M.C.; Cam, E.; Hardalac, F. Forecasting electricity consumption: A comparison of regression analysis, neural networks and least squares support vector machines. *Int. J. Electr. Power Energy Syst.* **2015**, *67*, 431–438. [CrossRef]
7. Chakraborty, P.; Marwah, M.; Arlitt, M.; Ramakrishnan, N. Fine-grained photovoltaic output prediction using a bayesian ensemble. In Proceedings of the AAAI Conference on Artificial Intelligence, Palo Alto, CA, USA, 2–9 February 2012; p. 26.
8. Akaike, H. Fitting autoregressive models for prediction. *Ann. Inst. Stat. Math.* **1969**, *21*, 243–247. [CrossRef]
9. Connor, J.T.; Martin, R.D.; Atlas, L.E. Recurrent neural networks and robust time series prediction. *IEEE Trans. Neural Netw.* **1994**, *5*, 240–254. [CrossRef]
10. Rasheed, F.; Alhajj, R. A framework for periodic outlier pattern detection in time-series sequences. *IEEE Trans. Cybern.* **2013**, *44*, 569–582. [CrossRef]
11. Zhang, M.; Jiang, X.; Fang, Z.; Zeng, Y.; Xu, K. High-order Hidden Markov Model for trend prediction in financial time series. *Phys. A Stat. Mech. Its Appl.* **2019**, *517*, 1–12. [CrossRef]
12. Box, G.E.P.; Pierce, D.A. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *J. Am. Stat. Assoc.* **1970**, *65*, 1509–1526. [CrossRef]
13. Kim, K. Financial time series forecasting using support vector machines. *Neurocomputing* **2003**, *55*, 307–319. [CrossRef]
14. Van Gestel, T.; Suykens, J.A.K.; Baestaens, D.E.; Lambrechts, A.; Lanckriet, G.; Vandaele, B.; De Moor, B.; Vandewalle, J. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Trans. Neural Netw.* **2001**, *12*, 809–821. [CrossRef] [PubMed]
15. Xu, K.; Xie, M.; Tang, L.C.; Ho, S.L. Application of neural networks in forecasting engine systems reliability. *Appl. Soft Comput.* **2003**, *2*, 255–268. [CrossRef]
16. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
17. Chung, J.; Gulcehre, C.; Cho, K.H.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv Prepr.* **2014**, arXiv:1412.3555.
18. Sagheer, A.; Kotb, M. Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing* **2019**, *323*, 203–213. [CrossRef]
19. Karim, F.; Majumdar, S.; Darabi, H.; Harford, S. Multivariate LSTM-FCNs for time series classification. *Neural Netw.* **2019**, *116*, 237–245. [CrossRef]
20. Kumar, A.; Irsoy, O.; Ondruska, P.; Iyyer, M.; Bradbury, J.; Gulrajani, I.; Zhong, V.; Paulus, R.; Socher, R. Ask me anything: Dynamic memory networks for natural language processing. In Proceedings of the 33rd International Conference on Machine Learning, PMLR, New York, NY, USA, 20–22 June 2016; Volume 48, pp. 1378–1387.
21. Merity, S.; Keskar, N.S.; Socher, R. Regularizing and optimizing LSTM language models. *arXiv Prepr.* **2017**, arXiv:1708.02182.
22. Graves, A.; Jaitly, N.; Mohamed, A. Hybrid speech recognition with deep bidirectional LSTM. In Proceedings of the 2013 IEEE workshop on automatic speech recognition and understanding, Olomouc, Czech Republic, 8–12 December 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 273–278.
23. Cho, K.; Van Merriënboer, B.; Bahdanau, D.; Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv Prepr.* **2014**, arXiv:1409.1259.
24. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv Prepr.* **2014**, arXiv:1409.0473.
25. Goel, H.; Melnyk, I.; Banerjee, A. R2N2, residual recurrent neural networks for multivariate time series forecasting. *arXiv Prepr.* **2017**, arXiv:1709.03159.

26. Lai, G.; Chang, W.C.; Yang, Y.; Liu, H. Modeling long-and short-term temporal patterns with deep neural networks. In Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, New York, NY, USA, 8–12 July 2018; pp. 95–104.
27. Shih, S.Y.; Sun, F.K.; Lee, H. Temporal pattern attention for multivariate time series forecasting. *Mach. Learn.* **2019**, *108*, 1421–1441. [CrossRef]
28. Qin, Y.; Song, D.; Chen, H.; Cheng, W.; Jiang, G.; Cottrell, G. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv Prepr.* **2017**, arXiv:1704.02971.
29. Lin, T.; Horne, B.G.; Tino, P.; Giles, C.L. Learning long-term dependencies in NARX recurrent neural networks. *IEEE Trans. Neural Netw.* **1996**, *7*, 1329–1338.
30. Gao, Y.; Er, M.J. NARMAX time series model prediction: Feedforward and recurrent fuzzy neural network approaches. *Fuzzy Sets Syst.* **2005**, *150*, 331–350. [CrossRef]
31. Menezes, J.M.P., Jr.; Barreto, G.A. Long-term time series prediction with the NARX network: An empirical evaluation. *Neurocomputing* **2008**, *71*, 3335–3343. [CrossRef]
32. Zhang, Y. Solar Power Data for Integration Studies, NREL. 2015. Available online: http://www.nrel.gov/grid/solar-power-data.html (accessed on 1 June 2021).
33. Trindade, A. ElectricityLoadDiagrams20112014 Dataset, UCI. 2006. Available online: https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014 (accessed on 1 June 2021).
34. Chen, S.X. Beijing Multi-Site Air-Quality Data Dataset, UCI. 2019. Available online: https://archive.ics.uci.edu/ml/datasets/Beijing+Multi-Site+Air-Quality+Data (accessed on 1 June 2021).