DPAST-RNN: A Dual-Phase Attention-Based Recurrent Neural Network Using Spatiotemporal LSTMs for Time Series Prediction

Shajia Shan[1,2], Ziyu Shen[1,2], Bin Xia[1,2], Zheng Liu[1,2], and Yun Li[1,2(✉)]

[1] Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, Nanjing University of Posts and Telecommunications, Nanjing, China
liyun@njupt.edu.cn

[2] School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, China

Abstract. For time series forecasting, the weight distribution among multivariables and the long-short-term time dependence are always very important and challenging. Traditional machine forecasting can't automatically select the effective features of multivariable input and can't capture the time dependence of sequences. The key to solve this problem is to capture the spatial correlations at the same time, the spatiotemporal relationships at different times and the long-term dependence of the temporal relationships between different series. In this paper, inspired by human attention mechanism including encoder-decoder model, we propose DPAST-based RNN (DPAST-RNN) for long-term time series prediction. Specifically, in the first phase we use attention mechanism to extract relevant features at each time adaptively then we use stacked LSTM units to extract hidden information of time series both from time and space dimensions. In the second phase, we use another attention mechanism to select the related hidden state in encoder to the hidden state of the decoder at the current time to make context vector which is embed into recurrent neural network in decoder. Thorough empirical studies based upon the VM-Power dataset we collected on OpenStack and the NASDAQ 100 Stock dataset demonstrate that the DPAST-RNN can outperform state-of-the-art methods for time series prediction.

Keywords: Time series prediction · Spatiotemporal LSTM · Attention mechanism · Encoder-decoder model

1 Introduction

Time series prediction algorithm has a wide range of applications, e.g., fine-grained photovoltaic output prediction [3], financial prediction [20], environmental forecasting [21], heart and brain signal analysis [7] and prediction of geo-sensor over future hours [13]. Generally, time series prediction can be divided

H. Yang et al. (Eds.): ICONIP 2020, LNCS 12534, pp. 568–578, 2020.
https://doi.org/10.1007/978-3-030-63836-8_47

into single variable problem and multivariable problem. However, in most cases, multivariable time series prediction problem is more in line with the needs of practical modeling. Different from the single variable time series prediction with strong periodicity, the problem of the multivariable prediction is mainly reflected in the following aspects: the correlation between the multivariable features at the same time, the correlation between the multivariable features at different times and the correlation between the multivariable features and the time of the target sequence. For some classical methods in time series prediction, ARIMA [1] assumes that the sequence variation is stable, so it is not suitable for non-stationary and multivariate time prediction. Support vector regression (SVR) [14], as a traditional regression method is used for time series prediction where feature sequences are mapped into high dimensional space, which pays more attention to the spatial correlations of these exogenous series at the same time, but ignores the time dependence. With the development of neural network, recurrent neural network (RNN) [18] especially Long short-term memory units (LSTM) [10] and gated recurrent unit (GRU) [5] are widely used in time series prediction. The encoder-decoder network structure was first proposed by Sutskever et al. [19] to solve the sequence to sequence machine translation problem. RNN based encoder-decoder network [5] was initially applied to machine translation. However, with the increasing length of vector representation, the performance of the encoder-decoder network deteriorated rapidly. Therefore, Bahdanau et al. [2] proposed the attention mechanism based on encoder-decoder structure. Attention mechanism has been widely used in machine translation [6], image caption [4], exogenous time series prediction [9], etc. Due to the success of attention-based encoder-decoder networks in sequence learning, Qin et al. [17] employ two-stage attention mechanism based on encoder-decoder structure to forecast multivariate time series. To capture the spatial dependency between sensors, Liang et al. [13] added global attention in GeoMAN. However, the decoder part of the models mentioned above does not fully consider the cyclic relationship between the target information and the encoded data in time.

In this paper, we use spatiotemporal LSTMs in the encoder network to obtain more accurate spatiotemporal relationship of the input data, and then embed the context information into the LSTM in the decoder network to enhance the attention of the target sequence to the encoding information in time. In addition, we build OpenStack virtual environment to collect VM power dataset and use DTW to preprocess and filter the data. The contributions of our work are three-fold:

- In the stage of data preprocessing, we use DTW [15] to analysis the original multivariate data and extract the effective feature variables in our dataset.
- In addition, considering that the single-layer LSTM can not transfer the effective information of multivariate input data, we use the spatiotemporal LSTMs to encode time series information as the input of decoder after the input attention mechanism.

– In the decoder, we embed the context vector generated by the temporal attention mechanism into recurrent neural network, so as to obtain a more accurate spatiotemporal relationship.

2 Model

The framework of the proposed forecasting model is shown in Fig. 1, which consists of encoder and decoder. The two phases attention modules are contained in the encoder and decoder respectively. The first phase in Encoder can adaptively select the most relevant input features while the second phase in Decoder uses categorical information to decode the stimulus. The Encoder encodes the time series conditioned on the input attention through the spatiotemporal LSTMs. In the decoder, the temporal attention is used to generated context vector c_t which represents a weighted sum of previous encoder hidden state across all the time steps. Then we combine the c_t with the hidden state in LSTM unit as the new hidden state fed to LSTM.

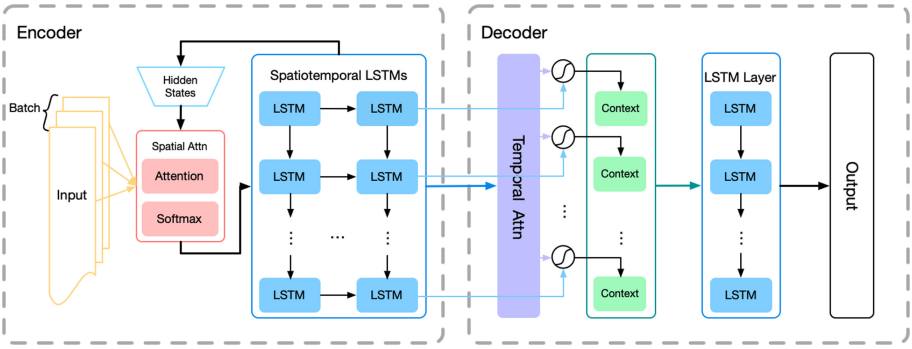

Fig. 1. Graphical illustration of the Dual-Phase Attention-based Recurrent Neural Network using Spatiotemporal LSTMs model.

2.1 Encoder

The encoder is used to encode the input sequence in time window T into the feature representation through RNN. Inspired by the DSTP [11] model which can select elementary stimulus features in the early stages of processing and input attention mechanism in DA-RNN [17], we use spatial attention to select the relevant driving series adaptively.

For time series prediction, given the input sequence $X = (\mathbf{x^1}, \mathbf{x^2}..., \mathbf{x^n})^\top$ where n is the number of driving (exogenous) series, it can be divided into a series of time windows with T. Given the k-th input driving (exogenous) series $x^k = (x_1^k, x_2^k, ..., x_T^k)^\top$, we can construct an input attention mechanism by referring to the previous hidden state h_{t-1} and the cell state s_{t-1} in the encoder LSTM unit with:

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

and

$$\alpha_t^k = \frac{exp(e_t^k)}{\sum_{i=1}^{n} exp(e_t^i)} \tag{2}$$

where $v_e \in R^T$, $W_e \in R^{T\times 2m}$, $U_e \in R^{T\times T}$ are parameters to learn. After that, we employ a softmax function to ensure all the attention weights at per time step sum to one. With these attention weights, we can adaptively extract the driving time series with:

$$\tilde{x} = (\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, ..., \alpha_t^n x_t^n) \tag{3}$$

Then the encoder is applied to learn a mapping from x_t to h_t (at time step t) with $h_t = f_e(h_{t-1}, x_t)$ can be updated as $h_t = f_e(h_{t-1}, \tilde{x}_t)$ where f_e is a spatiotemporal LSTM architecture based on LSTM units can be summarized as follows:

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

$$i_t = \sigma(W_i[h_{t-1}; x_t] + b_i) \tag{5}$$

$$o_t = \sigma(W_o[h_{t-1}; x_t] + b_o) \tag{6}$$

$$s_t = f_t \odot s_{t-1} + i_t \odot tanh(W_s[h_{t-1}; x_t] + b_s) \tag{7}$$

$$h_t = o_t \odot tanh(s_t) \tag{8}$$

where $[h_{t-1}; x_t] \in R^{m+n}$ is a concatenation of the previous hidden state h_{t-1} and the current input x_t, W_f, W_i, W_o, $W_s \in R^{m\times(m+n)}$, and b_f, b_i, b_o, $b_s \in R^m$ are parameters to learn.

In order to enhance the ability of LSTM to capture long-term memory, we use two layers of stacked LSTM to transmit information in space and time. At every time step t, the first layer of LSTM is $h_t^l = f_e^l(h_{t-1}^l, \tilde{x}_t)$ where $l = 1$. Given the current level of LSTM layer l where $l \geqslant 2$, the output can be updated with:

$$h_t^l = f_e^l(h_{t-1}^l, h_t^{l-1}) \tag{9}$$

then the output is a concatenation of the previous T hidden state of the LSTM units as the encoded input driving series.

2.2 Decoder

In order to predict the output $\tilde{y_t}$, we use another LSTM to decode the input infomation. In the decoder, the attention weight of the decoder hidden state at time t is calculated based upon the previous decoder hidden state d_{t-1} and the cell state of the LSTM unit $s_{t-1}^{'}$ with:

$$l_i^t = \mathbf{v_d}^\top tanh(W_d[d_{t-1}; s_{t-1}^{'}] + U_d h_i), 1 \leqslant i \leqslant T \tag{10}$$

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

where $[d_{t-1}; s'_{t-1}] \in R^{2p}$ is a concatenation of the previous hidden state and cell state of the LSTM unit in the decoder and h_i is concatenation of the hidden state in last time window T. $v_d \in R^m$, $W_d \in R^{m \times 2p}$, $U_d \in R^{m \times m}$ are parameters to learn. The weights of the i-th encoder hidden states β_t^i represent the importance it take at time t^i. Since each encoder hidden state h_i is mapped to a temporal component of the input, the context vector c_t can be computed as a weighted sum of all encoder hidden states $\{h_1, h_2, ..., h_T\}$,

$$c_t = \sum_{i=1}^{T} \beta_t^i h_i \tag{12}$$

Then the updated history target value can be combined with c_{t-1} and the given target series $y_{t-1} = \{y_{t-1}, y_{t-1}, ..., y_{t-1}\}$:

$$\tilde{y}_{t-1} = \mathbf{y_{t-1}}^\top \cdot c_{t-1} \tag{13}$$

where $\mathbf{y_{t-1}}^\top \cdot c_{t-1}$ is the point product of the decoder input y_{t-1} and the computed context vector c_{t-1}.

In order to enhance the influence of context vector on decoder, we combine the context vector with the hidden state of the decoder at every moment, the new hidden state can be updated after a linear layer as,

$$\tilde{d_t} = \mathbf{v_c}^\top tanh(W_c[c_t; d_{t-1}]) \tag{14}$$

where $[c_t; d_{t-1}] \in R^{m \times p}$ is a concatenation of the previous hidden state in LSTM unit of decoder and the current context vector c_t. We choose the nonlinear function f_d as a LSTM unit [10] to model long-term dependencies. Then the hidden state d_t can be updated as:

$$d_t = f_d(\tilde{d}_{t-1}, \tilde{y}_{t-1}) \tag{15}$$

and the final prediction can be computed as:

$$\tilde{y_T} = \mathbf{v_y}^\top (W_y[d_T; c_T] + b_w) + b_v \tag{16}$$

where $[d_T; c_T] \in R^{p+m}$ is a concatenation of the decoder hidden state and the context vector and the W_y, b_w, b_v are the parameters to learn.

2.3 Training Procedure

The model is based on encoder-decoder structure and parameters can be learned by standard back propagation with mean squared error as the objective function:

$$L(y_T, \tilde{y}_T) = \frac{1}{N} \sum_{i=1}^{N} (y_T^i - \tilde{y}_T^i)^2 \tag{17}$$

where N is the number of training samples. We choose Adam optimizer [12] to train the model and the size of the minibatch is 128. The learning rate is 0.001. Specifically, the proposed DPAST-RNN can make the loss function converge quickly.

3 Experiments

In this section, we first introduce the two datasets for this experiment. In addition, we introduce the collection process of VM-Power dataset. Then we discuss the parameter settings for DPAST-RNN and the evaluation metrics. Finally, we compare the DPAST-RNN with three different baseline methods.

3.1 Data Acquisition

In order to verify the performance of our DPAST-RNN model on more time series data, we configured the OpenStack environment to collect the indicators and real power of the virtual machine to make VM-Power dataset. There is an OpenStack controller node, an OpenStack compute node, and a monitor node for collecting the data from the compute node. These nodes are connected to the same (Local Area Network) LAN. The power of IT equipment can be measured by the Power Distribution Unit (PDU). The architecture is shown in Fig. 2.

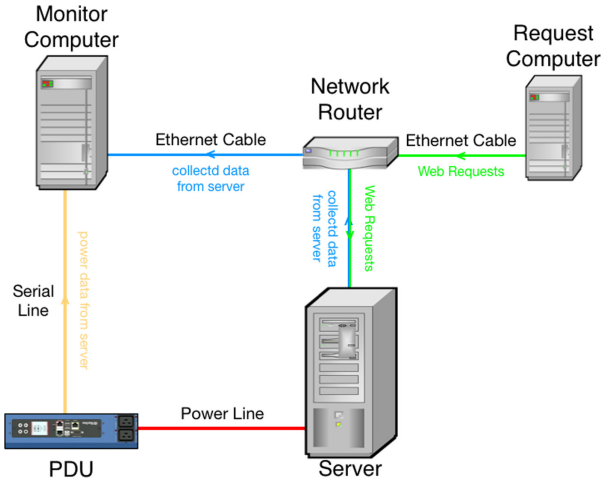

Fig. 2. The architecture of data collection procedure for VM-Power.

We deployed a collector called collectd [8] on the compute node to collect metrics of the compute node. The sampling frequency of collectd is set to 1 Hz, the same as the sampling frequency of PDU. Specifically, we use a client machine with a Quad-core CPU to request web resource and collect virtual machine metrics per seconds with real power in PDU.

3.2 Datasets and Setup

In this experiment, we used two datasets NASDAQ 100 Stock and VM-Power as shown in Table 1 where the size of encoder hidden states m and decoder hidden states p are set as $m = p = 64$ and 128 to test the performance of different methods for time series prediction.

Table 1. The statistics of two datasets.

Dataset	Driving series	Target series	Size		
			Train	Valid	Test
VM-Power	10	1	2636	263	528
NASDAQ 100 Stock	80	1	40551	4055	8111

The NASDAQ 100 Stock is a public dataset which contains the stock prices of 81 major corporations under NASDAQ 100. In this dataset, we use the share price of NDX as the target sequence and the share price of the remaining 80 companies as the driving time sequence.

From over 100 metrics we collected in origin VM-Power dataset, we draw a line chart of power and some features to simply analyze the correlation between them. As shown in Fig. 3, the trend of the four CPU cores usage is roughly as same as the trend of the power curve. On the contrary, memory-free, memory-cached, irq-CAL, cup-2-idle, are not related to or even contrary to power trend, so we use dynamic time warping (DTW) [15] to measure the similarity between feature variables and target sequences and select effective variables in the data preprocessing stage to enhance the robustness of the model. Compared with the traditional Euclidean distance, DTW can better compare the similarity of two time waveforms by distorting the sequence on the x-axis. The 10 metrics from DTW selection are cpu-0-usage, cpu-1-usage, cpu-2-usage, cpu-3-usage, cpu-0-user, cpu-1-user, cpu-2-user, cpu-3-user, cpu-0-system, cpu-1-system.

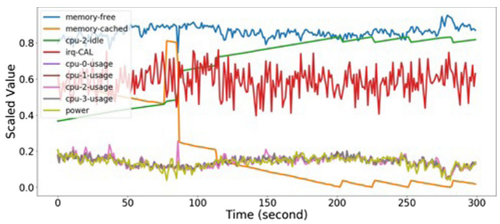

Fig. 3. The curves of power and the features selected of VM-Power dataset.

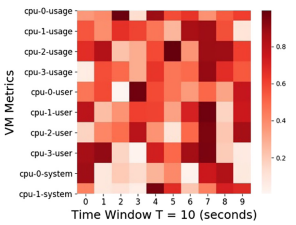

Fig. 4. Plot of input spatial attention weights in one time window $T = 10$ for 10 virtual machine energy consumption index variables in VM-Power dataset.

3.3 Parameter Settings

We initialized the size of hidden states 128 both in encoder and decoder and choose the window size T=10 where T $\in$ {5, 10, 15, 20, 25} that achieve the best performance over the validation set are used for evaluation. To measure the effectiveness of various methods for time series prediction, we consider two different evaluation metrics, root mean squared error (RMSE) [16] and mean absolute error (MAE). Given y_t is the target at time t and $\hat{y}_t$ is the predicted value at time t, RMSE is defined as $RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i^t - \hat{y}_i^t)^2}$ and MAE is defined as $MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i^t - \hat{y}_i^t|$.

3.4 Results: Time Series Prediction

We compared our DPAST-RNN with three baseline methods in two datasets and proved its effectiveness. The results of prediction in two datasets are shown in Fig. 5 and 6. Among these baselines, LSTM [10] is a basic method to address time series prediction in RNN. From the prediction results in Fig. 5, the model based on RNN can better predict the time series data with more severe fluctuations. For the rising part of continuous oscillation, our model can better reduce the time delay. We also show the visual attention distribution in Fig. 4. We observe that the different characteristic variables get different weights in time window T which indicates that input attention mechanism can effectively extract the relevant driving sequence.

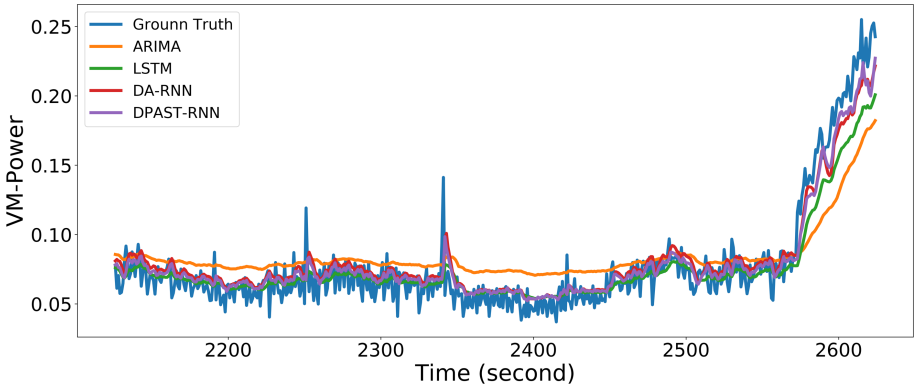

Fig. 5. VM-Power prediction result.

The time series prediction results of DPAST-RNN and baseline methods over the two datasets are shown in Table 2. In Table 2, the results of the $RMSE$ of ARIMA is generally worse than the RNN based methods. This is because ARIMA only consider the target series rather than the relationship between

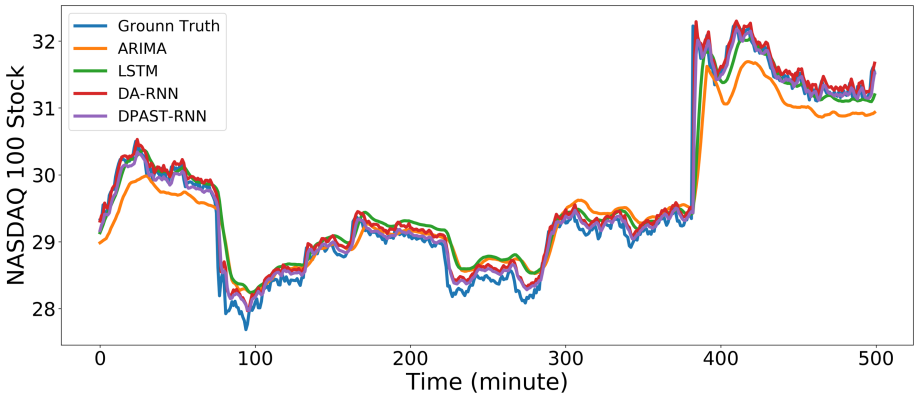

Fig. 6. NASDAQ 100 Index prediction result.

Table 2. Time series prediction results over the Vm-Power dataset and NASDAQ 100 Stock dataset (best performance displayed in boldface).

Models	VM-Power dataset		NASDAQ 100 Stock dataset	
	MAE	RMSE	MAE	RMSE
ARIMA	1.97	2.66	0.92	1.47
LSTM(64)	0.282 0.003	0.362 0.003	0.262 0.005	0.390 0.003
LSTM(128)	0.270 0.003	0.347 0.003	0.251 0.005	0.380 0.003
DA-RNN(64)	0.014 0.003	0.019 0.001	0.216 0.002	0.310 0.003
DA-RNN(128)	0.016 0.004	0.021 0.005	0.229 0.002	0.330 0.003
DPAST-RNN(64)	0.015 0.001	0.017 0.001	0.218 0.002	0.319 0.005
DPAST-RNN(128)	**0.012 0.001**	**0.014 0.001**	**0.212 0.002**	**0.298 0.005**

driving series. The encoder-decoder structure with integration of the input attention mechanism as well as temporal attention mechanism performs better than original LSTM. With integration of the input attention mechanism and spatiotemporal LSTMs in encoder as well as context vector embedded in recurrent neural network in decoder, our DPAST-RNN achieves the best MAE and RMSE across two datasets since it not only uses spatiotemporal LSTMs in encoder with input attention to extract relevant driving series, but also combine the context vector with hidden state in LSTM in the encoder to obtain a more accurate spatiotemporal relationship across all time steps.

4 Conclusion and Future Work

In this paper, we propose a DPAST-RNN model based on spatiotemporal LSTM network for time series prediction, which consists of two phases attention mechanism. In the proposed model, we use DTW to remove the noise of multivariate input time series. In the encoder part of DPAST-RNN, the spatiotemporal

LSTMs can accurately encode the driving series after input attention mechanism. In the decoder part of DPAST-RNN, the updated hidden state in LSTM with context vector can naturally capture the long-range temporal information of the encoded inputs. The experimental results on two datasets demonstrate a higher performance than other baseline methods.

In the future, we will explore time series prediction based on attention mechanism without RNN structure. Moreover, we will extend our method to solve the problem of long-term prediction.

Acknowledgments. This work was supported by National Key Research and Development Program of China (2018YFB1003702) and Jiangsu Scientific Research Innovation Practice Project (KYCX20_0760).

References

1. Amini, M.H., Kargarian, A., Karabasoglu, O.: ARIMA-based decoupled time series forecasting of electric vehicle charging demand for stochastic power system operation. Electr. Power Syst. Res. **140**, 378–390 (2016)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
3. Chakraborty, P., Marwah, M., Arlitt, M., Ramakrishnan, N.: Fine-grained photovoltaic output prediction using a Bayesian ensemble. In: Twenty-Sixth AAAI Conference on Artificial Intelligence (2012)
4. Cheng, Y., Huang, F., Zhou, L., Jin, C., Zhang, Y., Zhang, T.: A hierarchical multimodal attention-based neural network for image captioning. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 889–892 (2017)
5. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
6. Di Gangi, M.A., Federico, M.: Deep neural machine translation with weakly-recurrent units. arXiv preprint arXiv:1805.04185 (2018)
7. Fernandez-Fraga, S., Aceves-Fernandez, M., Pedraza-Ortega, J., Ramos-Arreguin, J.: Screen task experiments for EEG signals based on SSVEP brain computer interface. Int. J. Adv. Res. **6**(2), 1718–1732 (2018)
8. Forster, F., Harl, S.: Collectd - the system statistics collection daemon (2012). https://collectd.org
9. Guo, T., Lin, T.: Multi-variable LSTM neural network for autoregressive exogenous model. arXiv preprint arXiv:1806.06384 (2018)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
11. Hübner, R., Steinhauser, M., Lehle, C.: A dual-stage two-phase model of selective attention. Psychol. Rev. **117**(3), 759 (2010)
12. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
13. Liang, Y., Ke, S., Zhang, J., Yi, X., Zheng, Y.: GeoMAN: multi-level attention networks for geo-sensory time series prediction. In: IJCAI, pp. 3428–3434 (2018)
14. Liu, J., Zio, E.: SVM hyperparameters tuning for recursive multi-step-ahead prediction. Neural Comput. Appl. **28**(12), 3749–3763 (2017). https://doi.org/10.1007/s00521-016-2272-1

15. Müller, M.: Dynamic time warping. In: Müller, M. (ed.) Information Retrieval for Music and Motion, pp. 69–84. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74048-3_4
16. Plutowski, M., Cottrell, G., White, H.: Experience with selecting exemplars from clean data. Neural Netw. **9**(2), 273–294 (1996)
17. 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 preprint arXiv:1704.02971 (2017)
18. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533–536 (1986)
19. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)
20. Wu, Y., Hernández-Lobato, J.M., Ghahramani, Z.: Dynamic covariance models for multivariate financial time series. arXiv preprint arXiv:1305.4268 (2013)
21. Zamora-Martinez, F., Romeu, P., Botella-Rocamora, P., Pardo, J.: On-line learning of indoor temperature forecasting models towards energy efficiency. Energy Build. **83**, 162–172 (2014)