THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,
Volume 23, Number 3/2022, pp. 309–316

RELATION PATTERNS EXTRACTION TO CLIMATE DATA USING LONG SHORT-TERM MEMORY NETWORKS

Yun YU, Xiao ZHANG

Jiangsu College of Finance & Accounting, Jiangsu, 222061, China
Email: yunyujs@163.com

Abstract. Climate data contains much time series set with unknown and spatial series set consisting of longitude and latitude, therefore, climate data can be considered to be a kind of typical time series data. Climate data has lots of rich information, through exploring those hidden relation pattern between time variables and spatial variables within climate data, the hidden internal regular of climate change can be exposed for providing some insights for predicting climate evolution. Whereas, the complex components and multi-dimensional characteristic of climate data bring a challenge for relation patterns extraction, moreover, the interference of redundant information, e.g., noise, hidden in climate data also creates a lot of trouble for relation patterns extraction. To address this, this paper proposed a long short-term memory network method for relation patterns extraction in climate data. Experiment results on the ECMWF climate dataset show that the proposed method not only wins competitors in extracted accuracy of relation patterns, but also can capture those advanced relation patterns by filtering the non-eigenvalue information in climate data. We find that these captured trajectories of climate present not only a cyclical continuity within an annual, but also they have a local smoothness in the four seasons. Moreover, these captured trajectories of climate are more affected by time series than spatial series does, implying that for the time series and spatial series, the main factor affecting the evolution of climate is time, followed by spatial location.

***Key words**:* climate data, relation pattern, long short-term memory network.

1. INTRODUCTION

Climate data is a kind of typical time series data, since it contains a lot of time series with unknown. Not only that, climate data has to do with spatial location consisting of longitude and latitude, so climate data has also spatial characteristic. Climate data consists of these complex time series and spatial series, which has lots of rich and valuable information. Through analyzing those relation patterns between time variables and spatial variables within climate data, we can explore the hidden internal regular of climate change, providing some insights for predicting climate evolution. Clearly, the composition components of climate data are really complex and multi-dimension, so that it becomes difficulty for relation patterns extraction between variables using manual manner, usually, machine intelligence methods are considered. Unfortunately, multi-dimensional characteristics of climate data are challenging the learning ability of machine intelligence methods, meanwhile, the interference of redundant information (e.g., noise) hidden in climate data creates much trouble for machine intelligence methods. As such, it is a tough task for relation patterns extraction from climate data. Some efforts for relation patterns extraction have been obtained. Currently, common methods are pattern extraction-based methods, feature selection-based methods and deep network architectures-based methods, which are capability of capturing relation between variables within data and learning those meaningful representations from complicated data. For (i) pattern extraction-based methods, e.g., these methods in [1−4], although such methods can gain valuable relation patterns from original data, they have to predefine some pattern rules according to those extracted patterns in tasks. In application, the operation of predefined rules is difficult to be performed. Indeed, those data similar to climate data are difficult for us to predefine those rules for relation patterns extraction because of the complexity and multi-dimensionality of those variables within data.

(ii) Feature selection-based methods, e.g., these instances implemented in [5] and in [6]. For such methods, the variation of features has effects on them, for example, the method in [7] is composed of obtaining a clustering for each subset of features and selecting the m features with the highest relevance measure, which gains the results with lower K-means error than competitors. However, the method in [7] needs to spend much time due to have to calculate per subset of features. Similar to the [7], the method proposed in [8] has increasing demands for selecting features in data long with the rapid increase of the data size, thereby increasing the computational cost.

(iii) Deep network architectures-based methods, such methods have become increasingly popular because of excellent ability of learning meaningful representations, e.g., the Deep Belief Networks [9], the Convolution Neural Networks [10]. Similarly, the [11] and the [12]. Deep network architectures-based methods show these outstanding advantages to capture variable relations since those hidden layers in neural networks are not simply to learn an identity function [13], more specifically, they can filter those redundant information from original data [14] so that the extracted representations become more compact [15]. For instance, sparse auto encoders are also commonly used to extract low-dimensional representations in high-dimensional data [16]. L. Zhao [17], et al proposed that the multi-modal neural networks to discover informative and heterogeneous feature patterns from different feature groups. The multi-modal neural networks [17] learn advanced variables representations, but they require predefine loss function aiming at different task features. Whereas, it is really hard to predefine loss function for different applications. In addition, Zheng [18] developed the neural networks with three layers to discover those relation patterns from climate data. Although the network architectures of three layers in [18] achieve a compact encoding and gain the desired results, unfortunately, climate data is typical time series data, which is closely related to time. So those relation patterns captured by the [18] do not present the time characteristics in the four seasons.

Motivation. The motivation for this work is to extract relational patterns from complex climate data. By exploring those relational patterns hidden in climate variables, some theoretical insights into climate evolution can be provided. As an important application, these patterns discovered are essential for exploring the relation between climate variation and greenhouse effect. Therefore, given that the time-series characteristics of climate data, this paper proposes a long short-term memory network to capture those relation patterns hidden in climate data.

We summarize main contributions in this work.

(i) Those architectures based on long short-term memory networks are better at treating climate time series data than other network architectures.

(ii) Temporal variables are the main factor affecting climate evolution, followed by spatial variables.

(iii) These captured trajectories in climate show a cyclical continuity within an annual, and exhibit a local smoothness in four seasons.

2. THE PROPOSED MODEL

Long short-term memory (LSTM) network is a special recurrent neural network (RNN). The difference from RNN is that LSTM introduces two concepts of gate mechanism and cell state. Although RNN is very sensitive to short-term input, it is much less effective for long-term input processing, whose of the disadvantage can be addressed by LSTM [19], because of introducing the three gate mechanisms: input gate, output gate, and forget gate, LSTM is better at treating long-term information. Clearly, this ability of LSTM to handle long-term information is more suitable for relational patterns extraction in climate data, since climate data is a long-term time series data.

The proposed LSTM consists of input gate, output gate and forget gate, as shown in Fig. 1. Input gate is used to update the cell state, which accepts the current input x_t, the output h_{t-1} of the previous stage and the cell state c_{t-1} in the hidden layer of the previous stage, together. The forget gate is responsible for deciding whether to save or discard information. Output gate obtains the current output h_t and the current cell state c_t after calculating, then h_t and c_t are used as the input at the next stage, together. Formally, the mathematical formulas of the update of an LSTM unit are given as following,

$$\begin{cases} i_t = \sigma\left(w_{xi}x_t + w_{hi}h_{t-1} + w_{ci}c_{t-1} + b_i\right) \\ f_t = \sigma\left(w_{xf}x_t + w_{hf}h_{t-1} + w_{cf}c_{t-1} + b_f\right) \\ c_t = f_t c_{t-1} + i_t h_t \\ o_t = \sigma\left(w_{xo}x_t + w_{ho}h_{t-1} + w_{co}c_t + b_o\right) \\ h_t = \tanh\left(w_{xh}\left[h_{t-1}, x_t\right] + b_h\right) \end{cases} \tag{1}$$

where σ, tanh is activation function. w, b is weight and bias. i_t, f_t, c_t and o_t represents input gate, forget gate, cell state and output gate, respectively. h_t is used to store information.

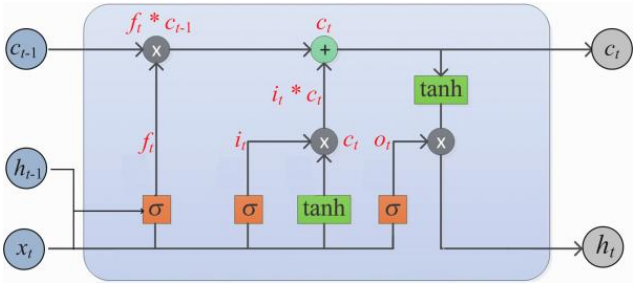


Fig. 1 – One cell of the LSTM memory block.

Batch normalization (BN) has been widely used to increase the training speed for neural network and to distribute inputs stably without affecting the reduction of internal covariate shift [20]. The reduction of internal covariate shift has effects on the parameters of the previous layer as well as on the distribution of the input to each layer, for instance, training a network with a data stack with size n being input, the formula for BN is given in Eq. (2).

$$\begin{cases} u_n = \dfrac{1}{n}\sum_{i=1}^{n} x_i \\ \sigma_n^2 = \dfrac{1}{n}\sum_{i=1}^{n}(x_i - u_n)^2 \\ x^{BN} = \dfrac{x_i - u_n}{\sqrt{\sigma_n^2 + \delta}} \end{cases} \tag{2}$$

where u_n is mean feature in a batch. σ_n^2 is variance feature in a batch. x^{BN} represents normalize feature of samples. δ is a constant to prevent the denominator from being zero. For the output of linear transformation, we used the formula in [21], as following

$$y_i \leftarrow \gamma x_i^{BN} + \beta \equiv BN(x_i). \tag{3}$$

Equation (3) represents the linear scale and shifts. γ, β is the pre-set parameters.

2.1. Hyper parameters of the model

Training the proposed LSTM means training the hyper parameters, including batch size, learning rate, etc. In order to ensure that LSTM can converge to a suitable value, meanwhile, to prevent over-fitting and under-fitting, the following hyper parameters were carefully studied since they have substantial effects on the training results.

(1) Learning rate. Learning rate, denoted as lr, has a significant impact on the convergence of the model, for instance, learning rate is too large, which may cause that loss value fast decreases, so as to appear

over-fitting. Therefore, cross-validation is used to determine learning rate, i.e., $lr \in \Upsilon$ ={1e–3, 1e–4, 1e–5, 1e–6, 1e–7}.

(2) Batch size. For the selection of batch size, local optimum may occur when batch size is too large, however, a small batch size introduces greater randomness, which difficulty achieves convergence. Smith [22, 23] et al show that for a fixed learning rate, there is an optimal batch size that can maximize test accuracy, moreover, batch size is positively correlated with learning rate and a training set scale, as shown in Fig. 2. According to Fig. 2, we dynamically adjust batch size in process of training.

Batch_Size	5000	2000	1000	500	256	100	50	20	10	5	2	1
Total Epoches	200	200	200	200	200	200	200	200	200	200	200	200
Total Iterations	1999	4999	9999	19999	38999	99999	199999	499999	999999	1999999	cannot converge	
Time of 200 Epoches	1	1.068	1.16	1.38	1.75	3.016	5.027	8.513	13.773	24.055		
Achieve 0.99 Accuracy at Epoch	-	-	135	78	41	45	24	9	9	-		
Time of Achieve 0.99 Accuracy	-	-	2.12	1.48	1	1.874	1.7	1.082	1.729	-		
Best Validation Score	0.015	0.011	0.01	0.01	0.01	0.009	0.0098	0.0084	0.01	0.032		
Best Score Achieved at Epoch	182	170	198	100	93	111	38	49	51	17		
Best Test Score	0.014	0.01	0.01	0.01	0.01	0.008	0.0083	0.0088	0.008	0.0262		
Final Test Error (200 epoches)	0.0134	0.01	0.01	0.01	0.01	0.009	0.0082	0.0088	0.008	0.0662		

Fig. 2 – Selection of batch size [20, 21].

(3) Convergence judgment. Currently, it is difficult to have a general method to accurately determine whether neural networks have been well trained and are the optimal, while there are still some ways to judge whether neural networks have converged, so as to stop training at a suitable position. For example, one approach is to monitor the loss on the training and testing sets, i.e. observing training curves. When the training loss and testing loss remain in a relatively stable state and the gap of both hardly changes, we can consider networks to be well trained.

2.2. Training of the model

The training for LSTM is given in algorithm 1. For step 1 and step 2, the input sample set S is firstly divided into training set S_{train}, testing set S_{test} and validation set S_{val}. Dataset S_{train} and S_{test} are used for the parameter training and testing, respectively. Dataset S_{val} is use for verifying the model. The procedure between step 3 to step 13 performs parameter testing to get the optimal parameter *opt*, i.e., cross-validation of parameters. After obtaining the optimal *opt*, together, using training set S_{train} and *opt* trains LSTM. As for step 14 to step 20, during performing p iterations computation, there will stop LSTM training until it can convergence. Finally, the procedure given in step 21 to step 25 shows that once LSTM is well trained, validation set S_{val} is used to verify LSTM, and LSTM sends out the training accuracy Train_Acc(max) and the maximum validation accuracy Acc(max).

Noting that in the initial batch size, according to Fig. 2 and the size of the input samples, we initialized an appropriate value for batch size, i.e., let *batch_size* be equal to 64. During training for LSTM, we dynamically adjust the value of *batch_size*, as shown in step 7 and step 8.

Algorithm 1. Training for LSTM
Input: sample set S, hyper parameters γ, β, Υ, *batch_size*=64, iterative epoch Q, P;
Output: training accuracy *Train_Acc*(*max*), maximum validation accuracy *Acc*(*max*);

```
1   S is randomly divided into three subsets;
2   Let S=S_val + S_test + S_train ;
3   for p=1 to P do:
4     foreach lr in Υ
5       Train LSTM with dataset S_train ;
6       Calculate training accuracy T_Acc= LSTM (S_train ; p; batch_size);
7       Update parameters with Eq. (2) and Eq. (3);
        Adjust batch_size according to TrainAcc ;
9       Test LSTM with dataset S_test ;
```

Calculate testing accuracy *Testing_Acc* = LSTM (S_{test} ; *p*; *batch_size*) ;
end foreach
end for
Get the optimal value *opt*=arg max(*Testing_Acc*; *lr*; *batch_size*);
for *q*=1 to *Q* **do**:
Train LSTM with data set S_{train} and *opt* ;
Calculate training accuracy *Train_Acc*(*q*) = LSTM(S_{train}; *opt*) ;
Update parameters with Eq. (2) and Eq. (3);
Verify LSTM with dataset S_{val};
Calculate validation accuracy *Val_Acc*(*q*) =LSTM(S_{val}) ;
end for
Select *q* so that q_{max}=arg max(*Train_Acc*(*q*)) ;
Get the maximum training accuracy *Train_Acc*(*max*) in the q_{max}-th iteration ;
Train_Acc(*max*)=LSMT(S_{train}; *opt*; q_{max});
Get the maximum validation accuracy *Acc*(*max*) =LSMT(S_{val}, q_{max}) ;
Output *Train_Acc*(*max*), *Acc*(*max*);

3. EXPERIMENTAL SETTINGS

3.1. Dataset

We selected the ECMWF climate data in an annul as the experiment dataset, with 512 dimensionalities and a data volume of 40000. Each dimensionalities is composed of eight variables at different spatial locations, including KM, KMLS, KH, KHLS, KHSFC, RI, 25(100 hPa), 27(100 hPa). For the detailed description of the ECMWF climate data, please see at https://atmosphere. copernicus.eu/

The ECMWF climate dataset was divided into three subsets, of which of 60% of the data is used as the training subset to train the model, and 20% of the data is used as the testing subset to test those parameters of the model, and the remaining data is used as the validation set to verify the ability of the proposed model to extract relation patterns.

3.2. Comparison methods and assessment metrics

We compared the DLM [18] with the proposed LSTM, because the DLM [18] is used for relation patterns extraction in climate data. To have fair results, these parameters of the DLM observed in the corresponding literature were adopted. In addition, PCA [18] is also used as a comparison. For other parameters, their default values were used.

For machine learning tasks, Receiver operating characteristic curve (ROC) and corresponding area under the curve (AUC) are often used as the evaluation metrics to evaluate the learning ability of methods. Hence, we used the AUC as assessment metric to assess the extracted precision of relation patterns.

We implemented the corresponding algorithms of the three methods using Python 3.6 on Tensorflow framework in Linux operation system. Unless otherwise stated, our method and comparison method all ran on a single GPU.

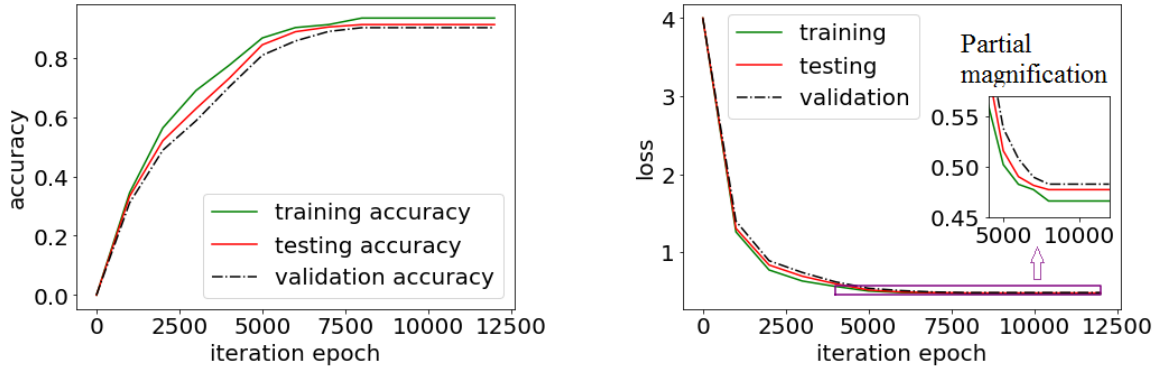
4. RESULTS

In this section, experiment results were given, including extracted accuracy and the captured relation patterns. To further analyze the internal regular of climate, these captured relation patterns were visualized by using the low-dimensional representations. The detailed results are below.

4.1. Mode testing

Figure 3a presents the training, testing and validation accuracies of the proposed model on the training subset, testing subset and validation subset. It can be seen that when iteration epoch is equal to 8000, the model starts to converge so that the training, testing and validation accuracies no longer change, meanwhile,

the training, testing and validation accuracies are 0.9330, 0.9111 and 0.9007, respectively. Figure 3b displays the corresponding the changes of loss value. Clearly, the loss values also do not occur change when iteration epoch reaches 8000. Together, these results in Fig. 3 also indicate that the model did not appear over-fitting.



(a) (b)

Fig. 3 – Results on training, testing and validation subsets.

4.2. Accuracy

Figure 4 displays the extracted accuracy of the three methods (our LSTM and two competitors). Results show that the proposed LSTM wins the two competitors in the extracted accuracy. The accuracy extracted by LSTM reaches 90.07%. While for the two competitors DLM and PCA, their extracted accuracy is 85.11% and 75.39%, respectively. These results indicate that the long short-term memory network method is better at treating time series data.

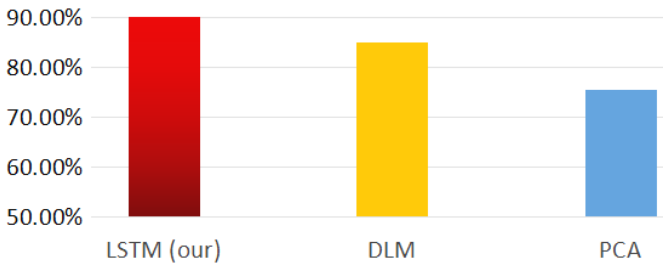


Fig. 4 – Extracted accuracy. The results are the average of 100 independent experiments.

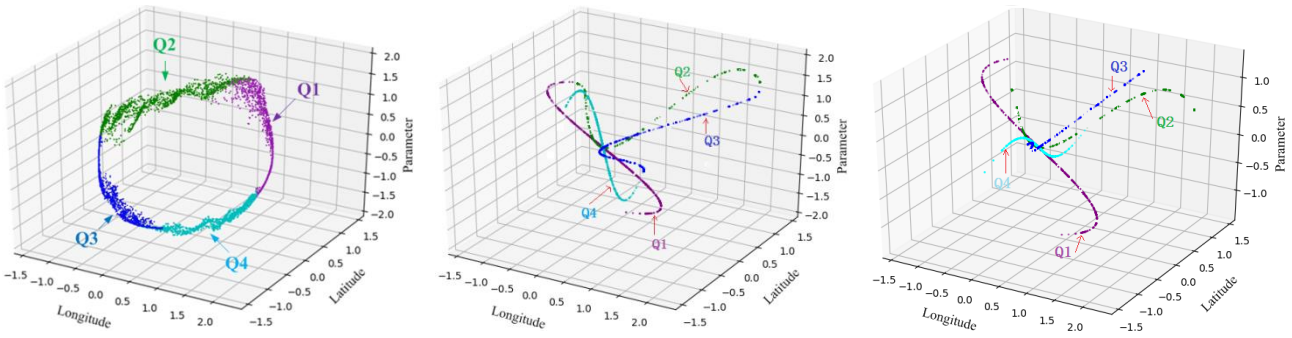
4.3. Relation patterns extraction

These extracted relation patterns were visualized by using the three-dimensional manner, as shown in Fig. 5. Low-dimensional visualized results show that the three methods can capture relation patterns from climate data. Nevertheless, these climate trajectories extracted by LSTM present more regular characteristics than that of extracted by the two competitors, i.e., these captured trajectories of climate present continuous periodic change in an annual through LSTM disposing in Fig. 5a. As for DLM in Fig. 5b and PCA in Fig. 5c, these captured trajectories of climate are weaker in the continuity of periodic change in an annual. These imply that the long short-term memory network method is capable of learning the deeper relation patterns in the time series data. Hence, LSTM can capture those desired relation patterns between climate variables, and those captured relation patterns show a degree of continuity and season layered characteristic what we expect.

Some observations can be gained from Fig. 4 and Fig. 5: (i) through filtering the non-eigenvalue information from climate data, LSTM captures these advanced relation patterns, implying that the long short-term memory network method is better at treating climate time series data than the other network architectures.

(ii) These captured trajectories of climate present a cyclical continuity within an annual, meanwhile, these captured trajectories also have a local smoothness in the four seasons.

(iii) These captured trajectories of climate are more affected by time series than spatial series does, implying that for the two time series and spatial series set, the main factor affecting the evolution of climate is time, followed by spatial location.



(a) patterns extracted by LSTM (b) patterns extracted by DLM (c) patterns extracted by PCA

Fig. 5 – Results of patterns extraction. Three-dimensional representations of all data points produced by LSTM, DLM and PCA. Colours represent quarters of a year, i.e., first quarter (Q1) is purple, second (Q2) is blue, third (Q3) is green, fourth (Q4) is cyan.

5. CONCLUSION

In this work, the LSTM network is proposed to extract the relation patterns from climate data. Experiment results on the ECMWF climate dataset show that the proposed method outperforms the mainstream comparison methods in the extracted precision of relation patterns. We demonstrate that the trajectory of climate change shows a cyclical continuity within an annual cycle. More importantly, these trajectories of climate change are more affected by time series than spatial series does, implying that for the time series and spatial series, the main factor affecting the evolution of climate is time, followed by spatial location. In the future work, we will look at exploring the relation between climate change and carbon dioxide concentration, in order to provide theory insights for greenhouse effects.

REFERENCES

1. Gaofeng MENG, Chunhong PAN, Shiming XIANG, Ying WU, *Baselines, extraction from curved document images via slope fields recovery*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **42**, pp. 793–808, 2020.
2. Yuting KONG, Dong NI, *Qualitative and quantitative analysis of multi-pattern wafer bin maps*, IEEE Transactions on Semiconductor Manufacturing, **33**, pp. 578–586, 2020.
3. Xiaosong JIA, Liting SUN, Masayoshi TOMIZUKA, Wei ZHAN, *IDE-Net: Interactive Driving Event and pattern extraction from human data*, IEEE Robotics and Automation Letters, **6**, pp. 3065–3072, 2021.
4. Yue XIANG, Yang WANG, Shiwei XIA, Fei TENG, *Charging load pattern extraction for residential electric vehicles: A training-free nonintrusive method*, IEEE Transactions on Industrial Informatics, **17**, pp. 7028–7039, 2021.
5. Wei ZHOU, Ling ZHANG, Shengyu GAO, Xin LOU, *Gradient-based feature extraction from raw Bayer pattern images*, IEEE Transactions on Image Processing, **30**, pp. 5122–5137, 2021.
6. Wei ZHOU, Shengyu GAO, Ling ZHANG, Xin LOU, *Histogram of oriented gradients feature extraction from raw Bayer pattern images*, IEEE Transactions on Circuits and Systems II: Express Briefs, **67**, pp. 946–950, 2020.
7. Marco CAPÓ, Aritz PÉREZ, Jose A. LOZANO, *A cheap feature selection approach for the K-means algorithm*, IEEE Transactions on Neural Networks and Learning Systems, **32**, pp. 2195–2208, 2021.
8. Xiaojun CHEN, Guowen YUAN, Feiping NIE, Zhong MING, *Semi-supervised feature selection via sparse rescaled linear square regression*, IEEE Transactions on Knowledge and Data Engineering, **32**, pp. 165–176, 2020.
9. Masoud KARIMI, Mehrdad MAJIDI, Hamed MIRSAEEDI, Mohammad Mehdi AREFI, Mohammad OSKUOEE, *A novel application of deep belief networks in learning partial discharge patterns for classifying corona, surface, and internal discharges*, IEEE Transactions on Industrial Electronics, **67**, pp. 3277–3287, 2020.
10. S. ZHENG, Y. HAO, D. LU, H. BAO, J. XU, H. HAO, B. XU, *Joint entity and relation extraction based on a hybrid neural network*, Neurocomputing, **257**, pp. 59–66, 2017.
11. N. ALANGARI, R. ALTURKI, *Predicting students final GPA using 15 classification algorithms*, Romanian Journal of Information Science and Technology, **23**, *3*, pp. 238–249, 2020.
12. P.K. UPADHYAY, C. NAGPAL, *Wavelet based performance analysis of SVM and RBF kernel for classifying stress conditions of sleep EEG*, Science and Technology, **23**, *3*, pp. 292–310 , 2020.

13. Jian ZHENG, Jianfeng WANG, Yanping CHEN, Shuping CHEN, Jingjin CHEN, Wenlong ZHONG, Wenling WU, *Neural networks trained with high-dimensional functions approximation data in high-dimensional space*, Journal of Intelligent & Fuzzy Systems, **41**, pp. 3739–3750, 2021.
14. Jian ZHENG, Jianfeng WANG, Yanping CHEN, Shuping CHEN, Jingjin CHEN, Wenlong ZHONG, Wenling WU, *Effective approximation of high‐dimensional space using neural networks*, Journal of Supercomputing, **78**, pp. 4377–4397, 2021.
15. Jian ZHENG, Jianfeng WANG, Jiang LI, Shuping CHEN, Lei SHU, Yike PENG, *Deep neural networks for detection abnormal trend in electricity data*, Proceedings of the Romanian Academy, Series A, **22**, *3*, pp. 291–298, 2021.
16. Xinyu GUO, Ali A. MINAI, Long J. LU, *Feature selection using multiple auto-encoders*, IEEE 2017 International Joint Conference on Neural Networks (IJCNN), 2017, pp. 4602–4609.
17. L. ZHAO, Q. HU, W. WANG, *Heterogeneous feature selection with multi-modal deep neural networks and sparse group LASSO*, IEEE Transactions on Multimedia, **17**, pp. 1936–1948, 2015.
18. Jian ZHENG, *Relational patterns discovery in climate with deep learning model*, Comptes Rendus de l'Académie Bulgare des Sciences, **74**, pp. 34–43, 2021.
19. H. SAK, A. SENIOR, F. BEAUFAYS, *Long short-term memory recurrent neural network architectures for large scale acoustic modeling*, Proc. 15th Annu. Conf. Int. Speech Commun. Assoc., 2014, pp. 338–342.
20. J. SARSFIELD, D. BROWN, N. SHERKAT, C. LANGENSIEPEN, J. LEWIS, M. TAHERI, C. MCCOLLIN, C. BARNETT, L. SELWOOD, P. STANDEN, P. LOGAN, C. SIMCOX, C. KILLICK, E. HUGHES, *Clinical assessment of depth sensor-based pose estimation algorithms for technology supervised rehabilitation applications*, Int. J. Med. Inform., **121**, pp. 30–38, 2019.
21. Pandapotan SIAGIAN, Erick FERNANDO, *Long short term memory networks for stroke activity recognition base on smartphone*, IEEE 5th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE), 2021, pp. 18–23.
22. S.L. SMITH, P.J. KINDERMANS, C. YING, *Don't decay the learning rate, increase the batch size*, arXiv: 1711.00489, 2017.
23. S.L. SMITH, Q.V. LE, *A Bayesian perspective on generalization and stochastic gradient descent*, arXiv: 1710.06451, 2017.

Received April 5, 2022