2019 Scientific Conference on Network, Power Systems and Computing (NPSC 2019)

MTCNN: A Deep Neural Network for Recognizing Stochastic Phase Shifted Time Series Data

Junpeng BAO, Xinyi LI
School of Computer Science and Technology,
Xi'an Jiaotong University,
Xi'an, P.R. China
baojp@mail.xjtu.edu.cn

Yu GAO
China Xi'an Satellite Control Center
Xi'an, P.R. China

***Abstract*—Time series data is very popular and common in the world, which is widely produced in various real time monitoring systems, such as satellites, power plants, cardiac electrophysiology, and financial transactions. In these applications, it is a critical problem to precisely recognize pattern of a random window on a growing time series stream. Since the data in the window has a stochastic phase shift with the standard known pattern. That leads to a great recognition error for the existing methods. This paper presents a Multi-channel-scale Time series Convolutional Neural Network (MTCNN) to recognize the patterns of time series data with a stochastic phase shift. The MTCNN has several channels with different size convolution kernels so that each layer packs multiple different convolution kernels and pooling structures. It is believed that this deep neural network broadens layers to enhance capability of adapting stochastic phase shift. The experimental results show that the MTCNN is superior to the existing methods, including the Nearest Neighbor method based on Euclidean Distance, the Nearest Neighbor method based on Dynamic Time Warping, Multi-Layer Perception, and the common Convolutional Neural Network.**

Keywords*—*Stochastic Phase Shift; Time Series; Pattern Recognition; Convolutional Neural Network

I. INTRODUCTION

Patten recognition of time series data is a very important task in many real time applications[1-4], such as satellite monitor system on ground, power plant operating status monitor system, cardiac electrophysiology and vital signs monitor system, and online financial transactions supervision system. Each known time series pattern usually represents some kind of working state, which is a critical evidence for making a decision or diagnosis. A time series pattern is a limited length of time series data. Usually, this length is fixed to the observing window length. For an alive real time monitor system, data is continuously generated. There is no definite beginning and end point. It means that a randomly observing window can fall on any time point, and the data in the window can be any subsequence of the whole time series data. However, a standard known time series pattern always has fixed content. A randomly observing window may not match the standard known pattern even though they are derived from exactly the same sequence of time series data. Because there may be a time lag between the window and the pattern, namely the window can be considered as a phase shifted pattern.

Fig. 1 illustrates this situation. The phase shift of window s1 is 0, and that of rs1 is 50. Then the wave form in s1 is quite different from that in rs1. Since the window s1 and rs1 are observing exactly the same piece of time series data, they should represent the same working state. In other words, the working state is not changed at all during the period of time, all windows should represent the same pattern, no matter what difference the phase shifts are, and what difference the wave forms are.

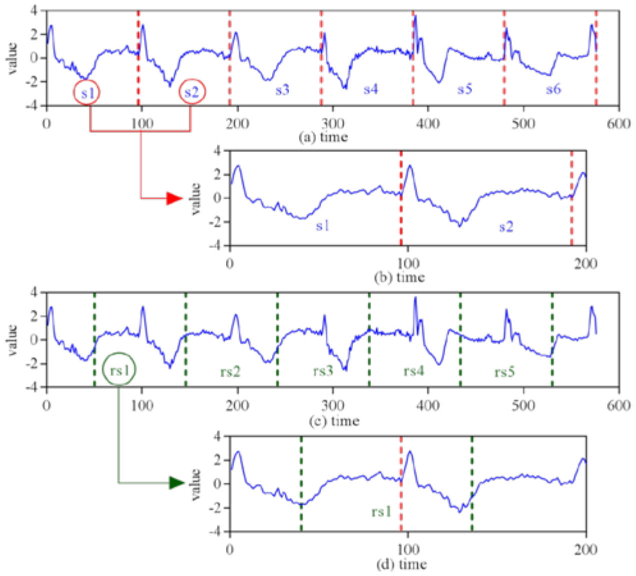

Fig. 1. A sequence of time series data is observed by different phase shifted windows so that the wave form in the window s1 is different from that in rs1

Unfortunately, the existing methods, including Euclidean Distance based similarity metric[5], Pearson Coefficients based similarity metric, Dynamic Time Warping(DTW) based similarity metric[6], Wavelet Transform based similarity metric, Multi-Layer Perception(MLP)[7], and the common Convolutional Neural Network(CNN)[8], failed to solve this time series stochastic phase shift issue because they statically measure the similarity between two sequences of time series data[9] rather than an alive time series stream.

In this paper, we present a Multi-channel-scale Time series Convolutional Neural Network (MTCNN) to recognize the patterns of time series data with a stochastic phase shift. The MTCNN has several channels with different size convolution kernels so that each layer packs multiple different convolution kernels and pooling structures. The experimental results show that the MTCNN is superior to the existing methods.

II. The Related Work

Most existing researches focus on the static similarity metric of the whole sequences of time series [10]. For example, Euclidean Distance and Pearson Coefficients can be directly used to measure time series data [5]. But these methods require sequences have the same length, and cannot resist noise.

Dynamic Time Warping (DTW) based method [6] can compare with different length time series data, which is often used to directly measure time series similarity. But it is also hard to depress noise. Wavelet Transform based method is able to extract both time and frequency domain features, which is commonly used to deal with noisy time series data.

In recent years, deep neural networks and deep learning methods [11] have been the most popular methods for pattern recognition problem because they have a strong feature learning ability. The Long Short Term Memory (LSTM) model [12,13], which is a kind of deep recurrent neural network, is designed for learning sequence patterns, so it is generally used for time series data to predict trend or recognize pattern. But the LSTM model has to carry out round by round and cannot perform parallel computing advantage. So, it consumes much longer training time. The Convolutional Neural Network (CNN) has full advantage of parallel computing and runs more faster than LSTM. The CNN has achieved a lot of successful applications in computer vision and speech recognition field [14-18].

Recently, the CNN is also applied to time series data. For example, Xu and Han [19] use an adaptive elastic echo state network to predict multivariate time series. Generally, there are two basic ways to deal with time series classification/pattern recognition problem with the CNN. One way directly takes the 1-Dimension time series as network input, and then modifies the CNN structure to fulfill the task. The other way firstly converts the 1-Dimension time series data into 2-Dimension images, and then uses the common CNN to deal with the image input.

For example, Zheng et al. [20] and Yi et al. [21] all exploit multi-channels CNN to classify time series. They consider multivariate time series data as several single variate time series data, which is sent to a separately CNN respectively for training. At last all outputs of each CNN are concatenated together as the input of a full connection MLP classifier. Cui et al. [22] propose a multi-scale CNN to extract different scale and frequency features from time series and perform classification after a smoothing operation.

Yang et al. [23] also propose a multi-channels CNN to recognize human activity. But they use a sliding window to cut long time series data into pieces, and change 1-Dimension pieces into 2-Dimension images. Then they use a CNN to learn image classification. Souza et al. [24,25] introduce two methods to encode a time series data into an image data, then use a CNN to recognize the image data. Hatami et al. [26] use recurrence plots to transform time series into 2-Dimension texture images and then take advantage of the deep CNN classifier. Wang and Oates [27] propose a framework to encode time series into different types of images, including Gramian Angular Summation/Difference Fields and Markov Transition Fields. Then they use tiled CNN for time series classification and imputation.

No matter we take the 1-Dimension time series directly or change it into images, the CNN structure has to be modified. The traditional CNN can only receive the fixed size input data. For the different size input, we have to cut or stretch them to fit the network. But these operations will lose some data and have a negative effect on the classification results.

III. The MTCNN Model

A. Motivation

Since the time series data has complex features and high dimensions. This paper combines the ideas of multi-channels and multi-scales to present a novel Multi-channel-scale Time series Convolutional neural network (MTCNN), which consists of several Multi-channel-scale blocks and full connection layers. A Multi-channel-scale block contains several parallel channels and each channel contains a convolution layer and a pooling layer whereas the different channel has different size of convolution kernel and pooling kernel. A block will concatenate all channels' output as the whole block's output. The Fig. 2 shows the structure of a Multi-channel-scale block, in which different size blue bars denote different size convolution kernels, and different size brown bars denote different size pooling kernels. The last yellow layer concatenates all channels' features as a whole output feature map, which will be the next block's input.

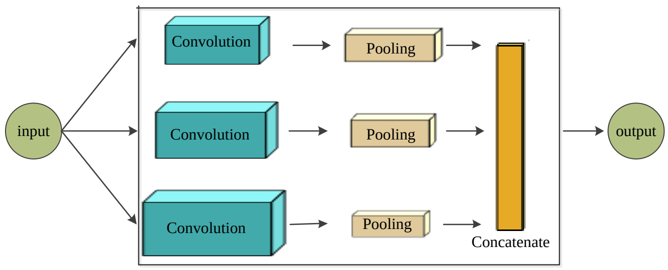

Fig. 2. The structure of a Multi-channel-scale block

Since different channel uses different size of kernel to extract different scale features, and different size of kernels has various receptive field. The output of the block will fuse multiple scale features and make the ability of adapting phase shifts. As a result, the whole network will adapt to the varying wave forms and phase shifts.

The Fig. 3 illustrates the whole structure of a MTCNN, which stacks many Multi-channel-scale blocks. The end of MTCNN is a Multi-Layer Perception(MLP) that consists of one or more full connection layers and a softmax output layer.

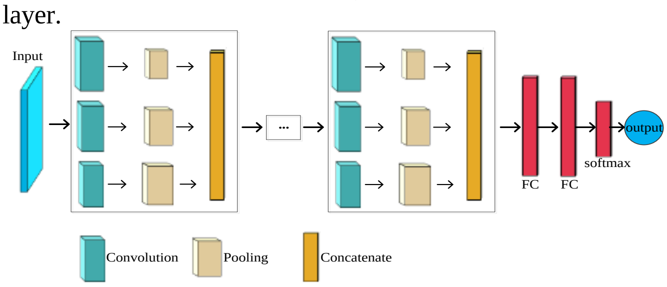

Fig. 3. The structure of a Multi-channel-scale Time series Convolutional neural network

B. The Structure Details

The input of the MTCNN is the original time series data that is directly input to all channels of the first Multi-channel-scale block without any conversion. In this paper, a Multi-channel-scale contains 3 channels that have different size of kernels in order to observe the data in multiple views. As well-known, for the same size of receptive field, the smaller the convolution kernel is, the fewer the model needs computation resources. It means that the smaller kernel can run faster than the bigger one. As a result, the convolution kernel sizes in 3 channels are 1×1,1×3, and 1×5 respectively. The 1×1 kernels can decrease the data dimensions, increase linear combinations among feature information in a channel.

In this paper, the number of convolution kernels of each layer grow with layer depth, i.e., 32, 64, 128, 256. The pooling layer chooses the max pooling method. We use the different size of pooling structure in order to make the different channel to produce the same size feature map output. After that, all channels' outputs are concatenated together at the end of block.

IV. EXPERIMENTAL RESULTS

A. The Pattern Label

When we test the time series pattern recognition problem, we must know the correct label of any input data. The input data is a number of fixed length observing windows randomly selected on the whole time series. However, we only have the labels of standard patterns, which have fixed position on the time series. The randomly selected observing window may not exactly match the standard pattern because they have a difference of phase shift (see Fig. 4). So, we have to make rules to label any randomly selected observing window.

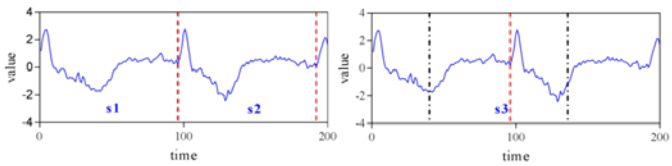

Fig. 4. A randomly selected observing window (s3) does not exactly match the standard pattern(s1 or s2), but they are all the same pattern because there is no any working state change

For example, in the Fig. 4, the s1 and s2 are two standard patterns, s3 is a randomly selected observing window. The s3 covers 56 data points at the left side with s1, and covers 40 data points at the right side with s2. Namely, the s3 contains 58% s1 and 42% s2. Then we define the cover rate cr(s3,s1)=58%, and cr(s3,s2)=42%. Since s1 and s2 are identical the same pattern, there is no state change from s1 to s2. So, the label of s3 should be identical to the s1 and s2. In real situation, a random observing window may not always have the same label with a standard pattern. We assign a label to an observing window according to its cover rates with two standard patterns.

For example, the Fig. 5 shows a long time series that is separated into 4 windows, in which contain 2 standard patterns A and B. The labels of the 4 windows are A, B, B, A respectively. However, a random observing window may overlap 2 patterns, i.e. (A,B), (B,B), and (B,A).

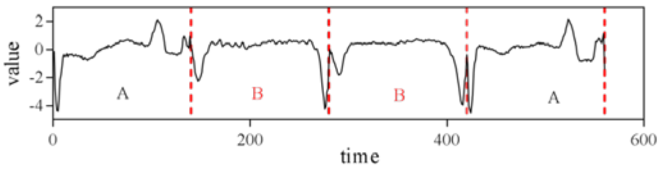

Fig. 5. A long time series is separated into 4 windows, in which contain 2 patterns

This paper makes window label rules according to the window's cover rate, as listed in table 1. In the table 1, a% and b% denote the cover rate of the observing window to the standard patterns, and a%+b%=1. A and B just represent two labels for two different standard patterns. The first two rules mean that if a window cover two identical standard patterns then their labels are identical too. Otherwise, If the covered patterns are not identical, then the window's label is decided by the lion's share, i.e. the rule 3 and 4. At last, the rule 5 means that if no covered pattern is in absolute proportion, i.e. no one is greater than 70%, then the label of the observing window is unknown, which is assigned a special symbol. C just means the observing window is different from any known standard pattern, which may be an unknown or uncertain or new pattern.

TABLE I. LABEL RULES FOR ANY OBSERVING WINDOW

Rule No.	The labels of the covered standard patterns	The cover rate	The label of the observing window
Rule1	A，A	a%，b%	A
Rule2	B，B	a%，b%	B
Rule3	A，B	a%，b%, and a% ≥ 70%	A
Rule4	A，B	a%，b%, and b% ≥ 70%	B
Rule5	A，B	a%，b%, max(a%, b%)<70%	C

This paper uses accuracy to evaluate test result. Its definition is as follows.

$$acc = m^T/m \tag{1}$$

where m denotes the total number of samples, mT denotes the number of samples that are correctly recognized.

B. The Datasets

This paper uses 5 datasets to test models. Table 2 lists details of these datasets.

TABLE II. THE DETAILS OF 5 DATASETS

Dataset	Number of standard patterns	Number of samples in training set	Number of samples in test set
ECG5000	5	500	4500
ECG200	2	100	100
Cardiology Challenge 2017	4	7677	1188
NN3	3	586	500
Satellite Data	5	5376	5376

The datasets ECG5000 and ECG200 are from UCR Time Series Classification Archive [28]. The dataset Cardiology Challenge 2017 is from the PhysioNet 2017 Challenge [29]. These 3 datasets are all electrocardiogram (ECG) data, which is a typical stationary periodic time series data. Each ECG data has a strong apparent periodicity with very few noise.

NN3 dataset is from a time series prediction contest held in 2007, which is a complex dataset that contains serveral kinds of time series data, including strong period data, period with trend data, complex uncertain period data (see Fig. 6). We combine original NN3 data into 3 patterns so that each pattern contains multiple modes of original data. Each pattern is a kind of nonstationary time series data that may have a trend so that the statistical informations of data may not be constant.

Satellite Data is a piece of data from a satellite monitoring system, which has some noises (see Fig. 7). The period of each statellite data pattern is not very strong, but each pattern is quite different from each other.

The data size of Cardiology Challenge 2017 and satellite are very large. The other 3 datasets are smaller than the former 2. Since the ECG5000 dataset is a very imbalanced dataset, we enhance the original ECG5000 dataset to make a enhanced ECG5000 dataset.

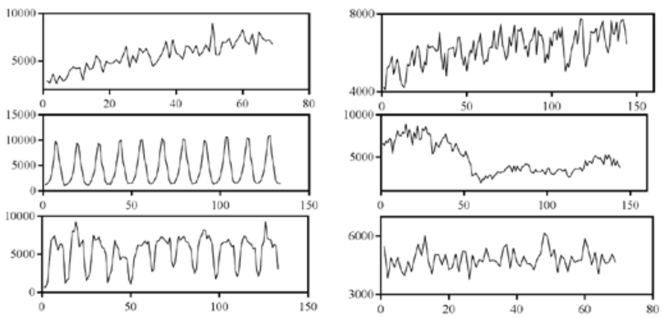

Fig. 6. Some data in the NN3 dataset

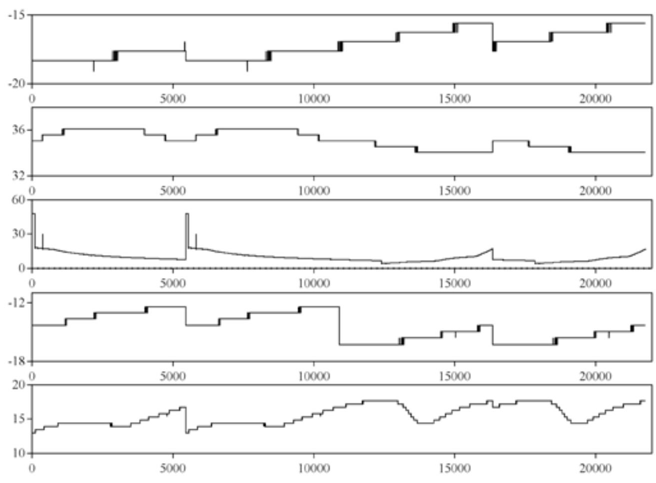

Fig. 7. Some data in Satellite Dataset

C. *MTCNN Test Results*

Table 3 lists all test results of MTCNN on the above 5 datasets. The ECG5000 dataset contains 5 standard patterns, but the sample number of each pattern is very imbalance. So, we enhanced the ECG5000 dataset by means of down sampling, sliding mean window, and adding noise to expand the training samples of small group pattern. The test data of ECG5000 is not modified. Namely, we only enhance the training set of ECG5000, no test set.

TABLE III. THE TEST RESULTS OF MTCNN ON 5 DATASETS

Dataset	**Pattern**	**Number of test samples**	**Number of correctly recognized**	**Accuracy (%)**
ECG200	1	64	62	96.88%
	2	36	33	91.67%
	sum	100	95	95.00%
ECG5000	1	2627	2609	99.31%
	2	1590	1431	89.94%
	3	86	27	31.45%
	4	175	71	40.57%
	5	22	3	13.63%
	sum	4500	4141	92.02%
Enhanced ECG5000	1	2627	2608	99.27%
	2	1590	1507	94.78%
	3	86	58	67.44%
	4	175	101	57.71%
	5	22	9	40.91%
	sum	4500	4283	95.18%
Cardiology Challenge 2017	1	299	282	94.31%
	2	299	289	96.66%
	3	293	276	94.19%
	4	297	271	91.25%
	sum	1188	1118	94.10%
NN3	1	176	117	66.14%
	2	162	112	69.63%
	3	162	115	70.86%
	sum	500	344	68.80%
Satellite Data	1	2544	2014	79.17%
	2	2000	1607	80.35%
	3	528	413	78.22%
	4	176	132	75%
	5	128	93	72.66%
	sum	5376	4259	79.22%

From the table 3, we can see that our MTCNN achieves a very good time series pattern recognition accuracy, which is greater than 94% for typical stationary periodic time series data, even though the test observing windows are randomly cut from the time series. It proves that our MTCNN can recognize the pattern of any selected observing window with any phase shifts. That is a critical ability for a model to run in a real time monitoring system and to perform an online detect/recognize task. It avoids time alignment between observing window and standard pattern so as to save a lot of time and speed up the system response.

Imbalanced training data always goes against machine learning. It is obvious that the enhanced ECG5000 dataset significantly improves accuracy from 92% to 95%. It is believed that the accuracy will greatly increase if we could get more samples of small pattern group. The pattern 1 of the ECG5000 has almost 100% recognition accuracy because it has sufficient samples. After the pattern 5 samples are enhanced, its accuracy is triply inflated. It indicates that the small sample data enhancement method is a better way to improve the generalization ability of model.

The MTCNN gets the worst performance on the NN3 dataset, in which most data are nonstationary time series data. The NN3 dataset contains three modes, i.e. steady trend, ascending trend, and downward trend. In NN3 dataset, a whole time series corresponds to one mode. Thus, a pattern contains too little data so that the network cannot be trained.

We combine original NN3 data into 3 patterns so that each pattern contains multiple modes of original data. As a result, the training process on the NN3 dataset fluctuates greatly.

The satellite data has many noises. Many samples of the same pattern are quite different. During the training process, the loss-function value fluctuates to a certain extent. However, the MTCNN can resists noise to achieve a good result.

D. The Comparison Test

In this section, we compare the MTCNN with other existing methods, including the Nearest Neighbor method based on Euclidean Distance that is marked as 1-NN(ED), the Nearest Neighbor method based on Dynamic Time Warping that is marked as 1-NN (DTW), Multi-Layer Perception (MLP), and the common Convolutional Neural Network (CNN). The datasets include ECG200, ECG5000, 50words, FaceALL, Cardiology Challenge 2017, NN3, and Satellite Data, where 50words and FaceALL are also from UCR Time Series Classification Archive [28].

The comparison result is shown in the Fig. 8, where each value is a method's accuracy over the whole dataset rather than over a single pattern. It can be obtained from the figure that the MTCNN proposed in this paper has significant advantage, whose accuracy is apparently always higher than the other recognition methods.

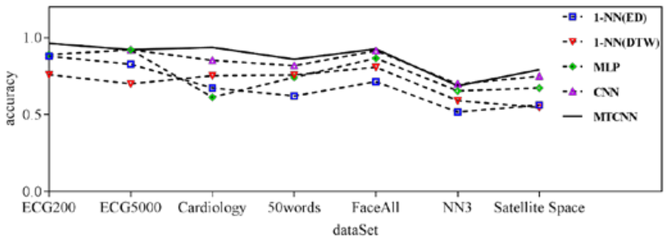

Fig. 8. The whole accuracies of time series pattern recognition of 5 methods on 7 datasets

V. CONCLUSIONS

The time series pattern recognition is a very important technology in many real time monitoring systems. The stochastic phase shift issue causes a lot of errors for the existing methods. This paper presents the MTCNN method to solve the issue. Based on the idea of combination multi-channels and multi-scales, the MTCNN constructs a deep neural network. It is believed that multiple channels with different size of kernels will fuse multiple scale features and make the whole network adapting stochastic phase shifts and varying wave forms.

We test the MTCNN on several datasets, including typical stationary periodic time series data, such as ECG data, nonstationary time series data, and time series with many noises, such as satellite data. We also compare the MTCNN with other 4 popular methods. The experimental results show that the MTCNN can precisely recognize the pattern of any selected observing window with any phase shifts, especially for the typical stationary periodic time series data. And the MTCNN outperforms the existing popular methods. However, the MTCNN still need improve its performance on the nonstationary time series data.

REFERENCES

[1] Shumway R, Stoffer D. Time series analysis and its applications. Springer, 2009.

[2] Xing Z, Pei J, Keogh E. A Brief Survey on Sequence Classification. ACM SIGKDD Explorations Newsletter, 2010, 12(1):40-48.

[3] Chen J. Similarity Measure of Time Series for Satellite Telemetry Data. Harbin Industrial University. 2015

[4] Jin W, Ping L, She H, et al. Bag-of-words representation for biomedical time series classification. Biomedical Signal Processing & Control, 2013, 8(6):634-644.

[5] Xing Z, Pei J, Yu P S. Early prediction on time series: a nearest neighbor approach. International Joint Conference on Artificial Intelligence (IJCAI). 2009, pp.1297-1302.

[6] Jeong Y S, Jeong M K, Omitaomu O A. Weighted dynamic time warping for time series classification. Pattern Recognition, 2011, 44(9):2231-2240.

[7] Haselsteiner E, Pfurtscheller G. Using time-dependent neural networks for EEG classification. Rehabilitation Engineering IEEE Transactions on, 2000, 8(4):457-463.

[8] Chien-Liang L, Wen-Hoar H, Yao-Chung T. Time Series Classification with Multivariate Convolutional Neural Network. IEEE Transactions on Industrial Electronics, 2018:1-1.

[9] Nanopoulos A, Alcock R, Manolopoulos Y. Feature-based classification of time series data. Information processing and technology. Nova Science Publishers, Inc. 2001:49-61.

[10] Hu, B., Chen, Y., Keogh, E. Time Series Classification under More Realistic Assumptions. In: SIAM International Conference on Data Mining, 2013, pp. 578

[11] Lecun Y, Bengio Y, Hinton G. Deep learning. Nature, 2015, 521(7553):436.

[12] Graves, A. Mohamed, A. Hinton, G. Speech recognition with deep recurrent neural networks, International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2013

[13] Liu YP, Hou D, Bao JP, et al. Multi-step Ahead Time Series Forecasting for Different Data Patterns Based on LSTM Recurrent Neural Network. The 14th Web Information Systems and Applications Conference (WISA), Liuzhou, China, 2017, pp. 305-310.

[14] LeCun, Y., Kavukcuoglu, K., Farabet, C. Convolutional networks and applications in vision. In: Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 253–256. IEEE (2010)

[15] Abdelhamid O, Mohamed A, Jiang H, et al. Convolutional Neural Networks for Speech Recognition. IEEE/ACM Transactions on Audio Speech & Language Processing, 2014, 22(10):1533-1545.

[16] Deng et al., Recent advances in deep learning for speech research at Microsoft. International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2013

[17] Graves, A. Mohamed, A. Hinton, G. Speech recognition with deep recurrent neural networks, International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2013

[18] Sainath T N, Mohamed A R, Kingsbury B, et al. Deep convolutional neural networks for LVCSR. IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013:8614-8618.

[19] M Xu, M Han. Adaptive Elastic Echo State Network for Multivariate Time Series Prediction. IEEE Transactions on Cybernetics. 2016, 46(10):2173-2183

[20] Zheng, Y. Liu, Q. Chen, E. Ge, Y. Zhao, J. Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks. Web-Age Information Management. 2014: 298-310

[21] Yi Z, Liu Q, Chen E, et al. Exploiting multi-channels deep convolutional neural networks for multivariate time series classification. Frontiers of Computer Science, 2016, 10(1):96-112.

[22] Cui, Z. Chen, W. Chen, Y. Multi-scale convolutional neural networks for time series classification. arXiv:1603.06995, 2016

[23] Yang J B, Nguyen M N, San P P, et al. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. International Joint Conference on Artificial Intelligence (IJCAI). 2015

[24] Souza, V. Silva, D. Batista, G. Extracting Texture Features for Time Series Classification. International Conference on Pattern Recognition (ICPR). 2014, pp. 1425-1430

[25] Souza, V. Silva, D. Batista, G. Time Series Classification Using Compression Distance of Recurrence Plots. IEEE International Conference on Data Mining (ICDM). 2013, pp.687-696

[26] Hatami N, Gavet Y, Debayle J. Classification of Time-Series Images Using Deep Convolutional Neural Networks. 2017.

[27] Wang, Z., Oates, T., Imaging Time series to Improve Classification and Imputation. International Joint Conference on Artificial Intelligence (IJCAI). 2015, pp.3939-3945

[28] Chen Y., Keogh E., Hu B., et al. The UCR Time Series Classification Archive. 2015. www.cs.ucr.edu/~eamonn/time_series_data/

[29] AF Classification from a Short Single Lead ECG Recording: the PhysioNet/Computing in Cardiology Challenge, 2017. https://physionet.org/challenge/2017/