IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON DATA MINING FOR INTERNET OF THINGS

Received October 1, 2019, accepted October 21, 2019, date of publication October 24, 2019, date of current version November 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2949287

Time Series Data Classification Based on Dual Path CNN-RNN Cascade Network

CHAO YANG, WENXIANG JIANG, AND ZHONGWEN GUO
School of Information Science and Engineering, Ocean University of China, Qingdao 266100, China

Corresponding author: Zhongwen Guo (guozhw@ouc.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61827810 and in part by the National Key Research and Development Project of China under Grant 2016YFF0202704.

ABSTRACT Time series data classification is a significant topic as its application can be found in a various domain. Recent studies have shown that data-driven approach based on deep learning is powerful for data mining tasks. A typical deep learning method, Artificial Neural Network (ANN), has been proven to be capable for match complicated functions thus leading to the popularity. Convolutional neural network (CNN) is a special kind of ANN that has been widely used in the area of image processing tasks as its ability for extracting spatial features. However, it remains a challenge for implementing CNN in time series data classification. Recurrent Neural Network (RNN) is popular for tackling time series data as it can effectively utilize temporal information. But it is time-consuming to train RNN. This paper proposes a Dual Path CNN-RNN Cascade Network (DPCRCN) that achieves an end-to-end learning for classification. We use a dual path CNN to achieve a multi-size receptive field for better feature extraction, then using RNN and the following fully-connected layers to learn the map between the given features and the output. We also use Region of Interest (RoI) pooling to make our model capable for a flexible shape of data. We evaluate our model on Activity Recognition system based on Multisensor data fusion (AReM) dataset and we compare with many popular algorithms. We also evaluate our model using different shape of data. The results show that our model outperforms the alternatives. In addition, we provide the details of training our model.

INDEX TERMS Convolutional neural network (CNN)-recurrent neural network (RNN) cascade model, dual path convolutional neural network, time series data classification, data mining.

I. INTRODUCTION

In recent years, time series classification has attracted lots of attention as its application can be seen in many aspects such as heart disease prediction [1], weather forecasting [2], automatic device classification [3], etc. For classification, how to sufficiently utilize the temporal information of time series data is the essential part. Machine learning algorithms are popular for data mining thanks to its solid statistical theory. Researchers extract features from the raw data as feature vectors and they are harnessed as the input of the subsequent machine learning algorithm. But traditional machine learning algorithms are based on the handcrafted features. It is obviously that designing the features manually is time-consuming. Besides, extracting appropriate features for the task relies on the expertise of the researchers. Comparing with traditional machine learning methods, deep learning methods are able to extract features automatically thus allowing the researchers to build an end-to-end model for concrete tasks. With the arrival of the era of big data, more and more data are available to train a deep learning model and it turns out the more data that are used to train the model, the stronger performance of the model is. So, deep learning algorithms outperforms the traditional machine learning algorithms and have played an important role in data mining, image processing and natural language processing [4], etc.

Deep learning algorithms are mainly about Artificial Neural Network (ANN) [5]. Neural network with multiple layers, also called Multi-Layer Perception (MLP), is capable for matching sophisticated function to solve different tasks. Images commonly contains much of spatial information that are tricky for MLP to extract and utilize effectively. To better harness the spatial information of the input data, Convolutional Neural Network (CNN) [6] is developed. Convolutional layers work like filters that have a significant output when facing a particular kind of spatial information such as edges or corners in the images which results in a better ability to utilize spatial information comparing with MLP.

The associate editor coordinating the review of this manuscript and approving it for publication was Tie Qiu.

Thanks to its ability of taking advantage of spatial information of images, CNN has remarkable achievements in image processing related areas such as image classification [7], object detection [8]–[10], image caption [11], etc.

The success of CNN in image processing also inspired researches in applying CNN in time series data as time series data contains temporal information that are kind of similar to spatial information of the images. For time series data, CNN is a filter that is sensitive to a certain kind of changing regulation. Thanks to that trait, CNN can extract features based temporal information of time series data and makes a fast and effective end-to-end time series data classification model possible. However, a robust and reliable based CNN model which is adaptive for a various kind of time series data is still lacking.

Recurrent Neural Network (RNN) [12] is a special structure that aims to take advantage of temporal information of the input data. As MLP suffers for ignoring the relationship of the current input data with the former and latter input data, RNN is developed. Comparing with MLP, RNN can pass the temporal information through the network. It plays an important role in the time series data related tasks such as machine translation [13], speech recognition [14], emotion classification [15], etc. But training a deep RNN is extremely time-consuming. Therefore, we consider a CNN-RNN cascade model for time series data classification.

The contributions of this paper are in the following two aspects:

Firstly, we propose a model based on dual path convolutional neural network to extract features based on temporal information of the time series data. Then we use RNN layers and fully connected layers to learn the map between the features and the output. Our model is named as Dual Path CNN-RNN Cascade Network (DPCRCN). The DPCRCN is able to extract features automatically and effectively and analyze not only the temporal information of a single variable but the interactions of a group of variables. In addition, we use adaptive Region of Interest (RoI) pooling to make our model adaptive to a flexible length of time series data.

Secondly, we conduct experiments on Activity Recognition system based on Multisensor data fusion (AReM) dataset [16] to assess our model and compare with both the methods used in [17] and other popular machine learning methods. What's more, we divide the validation data into pieces which contain different number of sequences. We examine our model on five different validation datasets to test the generalization of our model. Besides, we provide the details of the dataset, the model and the training procedure.

The rest of this paper is organized as follows. Section II presents related surveys and techniques. Section III introduces the proposed algorithm. Section IV demonstrates the result of the experiment and details of the training process. Section V presents the discussion. The conclusion and future work are shown in Section VI.

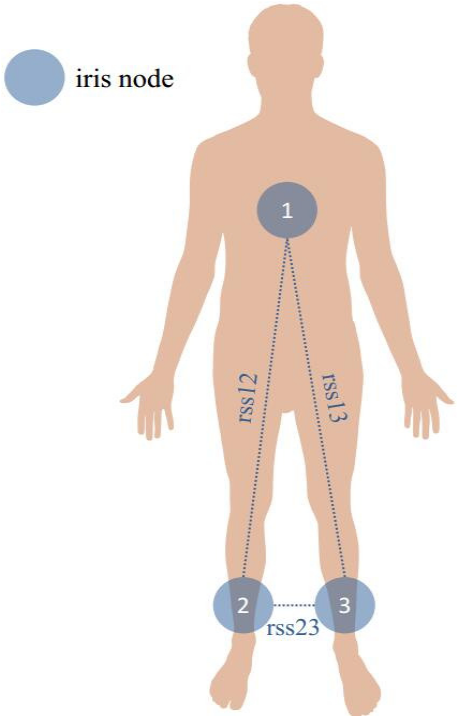

FIGURE 1. Schematic diagram of sensor placement.

II. RELATED WORK

This section introduces AReM dataset and the works that are related to our model.

A. AREM DATASET

This dataset represents a real-life benchmark in the area of Activity Recognition applications, as described in [16]. The classification tasks consist in predicting the activity performed by the user from time-series generated by a Wireless Sensor Network (WSN)

In this activity recognition system, information from coming the implicit alteration of the wireless channel due to the movements of the user are used. The devices measure the RSS of the beacon packets they exchange among themselves in the WSN [16]. The illustration of sensor placement given in [16] is shown in Figure 1.

The dataset providers collect RSS data using IRIS nodes embedding a Chipcon AT86RF230 radio subsystem that implements the IEEE 802.15.4 standard and programmed with a TinyOS firmware. They are placed on the user's chest and ankles. From the raw data the dataset providers extract time-domain features to compress the time series and slightly remove noise and correlations. They choose an epoch time of 250 milliseconds according to the EVAAL technical annex. In such a time slot they elaborate 5 samples of RSS (sampled at 20 Hz) for each of the three couples of WSN nodes (i.e. Chest-Right Ankle, Chest-Left Ankle, Right Ankle-Left Ankle). The features include the mean value and standard deviation for each reciprocal RSS reading from worn WSN sensors. For each activity 15 temporal sequences of input RSS data are present. Furtherly, the bending activity is divided into

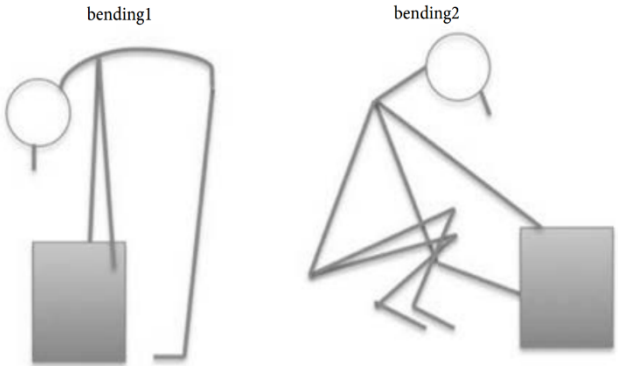

FIGURE 2. Schematic diagram of the two different kinds of bending.

bending1, which contains 7 temporal sequences, and bending2, which contains 6 temporal sequences. The schematic diagram of the two different kinds of bending given in [16] is shown in Figure 2.

The dataset contains 480 sequences, for a total number of 42240 instances.

Palumbo *et al.* [17] proposed two methods, called Leaky Integrator Echo State Network (LI-ESN) and Delay Neural Networks (IDNNs), for activity recognition. Accuracy and F1 score of the two mentioned methods are given in their paper.

B. MACHINE LEARNING METHODS FOR TIME SERIES CLASSIFICATION

Machine learning methods are widely used in classification tasks [18]–[20]. Logistic Regression (LR) is a typical method that tries to learn the relationship between different variables to predict the class of the data correctly. Support Vector Machine (SVM) [21] is a well-known supervised algorithm. The thought of SVM is to find a surface that maximize the distance of the support vectors to that surface. A method that often works with SVM called Kernel method allows mapping the data to higher dimension to find the ideal surface and that makes SVM powerful and popular in the area of data mining. ANN is another famous supervised learning algorithm. Based on forward propagation and backward propagation, ANN can learn a group of weights to fit complex functions. Although it suffers for unexplainable, ANN has achieved satisfied performances in a various kind of tasks. Ensemble learning method combines different kinds of classifiers into one model and generally has a better performance. Random Forest (RF) [22] and eXtreme Gradient Boosting (XgBoost) [23] are two representative ensemble learning algorithms. Ensemble learning has been widely used in many data science competition thanks to its powerful ability.

C. CONVOLUTIONAL NEURAL NETWORK AND RECURRENT NEURAL NETWORK

CNN is a specifically designed structure which is fast to train and capable to capture the spatial information of the raw data because of its weight sharing and translation invariance characteristics. The neurons in CNN respond to stimuli in a restricted region, often called receptive filed, of the visual field. The CNN can extract abstract features of the image in the shallow layers and aggregates the features in the deeper layers to achieve image understanding for image processing tasks. Szegedy *et al.* [24] proposed a structure called Inception which uses multiple convolutional layers with different kernel size to extend the width of the network result in multi-size receptive field for better capturing and utilizing spatial information. In an Inception module, the input data simultaneously pass through several layers including different convolutional layers and max pooling layers. The results are concatenated as the output of the Inception module. Thus, the Inception module is capable for extract multilevel and multiscale features from the input data which is helpful for improving the performance of the neural networks. In the latter work of Szegedy *et al.* [25], the original Inception was improved by using multiple 3×3 convolutional layers instead of 5×5 convolutional layers to reduce the number of parameters and faster the training process meanwhile maintaining the receptive field. They also use 1×1 convolutional layers to reduce the number of channels to shorten the training time. In [26], they proposed a new structure, called Inception-Resnet, to achieve a very deep and wide neural network by the combination of Inception and ResNet [27] structure which achieves a significant performance on image processing related tasks.

Batch Normalization [28] is helpful for accelerating training deep network. It can make the optimization landscape significantly smoother and help optimization [29]. It is a common layer used in CNN. To avoid overfitting, dropout [30] is a typical method. During the training process, neurons will randomly be invalid so the neural network will not be highly influenced by certain neurons and that enables the model to have a better generalization.

Yu and Koltun [31] proposed dilated convolutions to expand the receptive field, which achieves a state-of-the-art model for semantic segmentation. Generally, pooling layers are used to extend the receptive field but it suffers for information loss. The advantage of dilation convolutions is that it can extend the receptive field without information loss. An illustration of the difference between the normal convolution and dilated convolution is shown in Figure 3. By adding dilated convolutions, the receptive field is extended and all the information is used during computation which makes sure there is no information loss during the process.

To make neural network adaptable for a changeable and flexible shape of input data, Ren *et al.* [9] proposed RoI pooling to adjust the data of different sizes into a fixed size. In this way, the input shape of the neural network can be arbitrary. RoI pooling has been widely used in object detection and temporal action detection.

Although CNN has been successfully applied to images, applying CNN to time series data is still a challenging task. Cui *et al.* [32] tried a structure called Multiscale Convolutional Neural Network (MCNN) which applies different transformations to time series data. To tackle multivariate

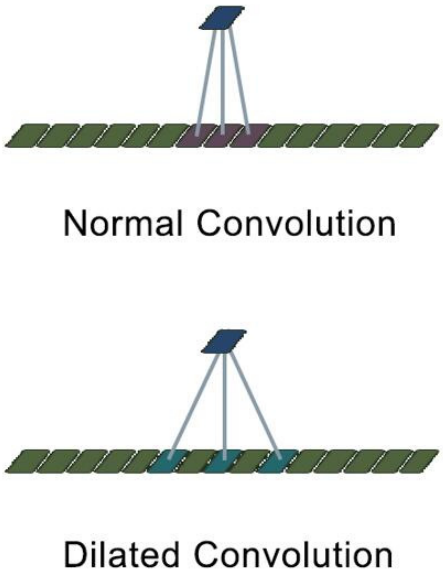

FIGURE 3. Illustration of the difference between normal convolution and dilated convolution.

time series data, Yi *et al.* [33] proposed Grouped Convolutional Neural Network (GCNN). Liu *et al.* [34] proposed an architecture called Multivariate Convolutional Neural Network (NVCNN) for multivariate time series classification. However, their models are not practicable for different kinds of time series data. Besides, the models are only capable for a fixed shape of time series data. So, an algorithm based on CNN that is feasible for a various kinds of time series data with a changeable and flexible size is still lacking.

RNN is a significant tool for time series data classification. Popular variant of RNN such as Long Short-Term Memory (LSTM) [35] and Gated Recurrent Unit (GRU) [36] are developed to prevent gradient vanishing and enable RNN to sufficient utilize temporal information on a relatively long-time span. LSTM and GRU have been widely used in an architecture called sequence-to-sequence model for time series related tasks. Although deep RNNs have made huge achievements for time series data analysis, it is extremely time-consuming to train a deep RNN model thus leading to consider a combination of CNN and RNN for time series data classification. Bai *et al.* [3] proposed an RNN-CNN cascade model for time series data classification aims to utilize the advantage of RNN and CNN to faster the training process and better utilize the temporal information of the data. But an effective architecture of the cascade model for time series data classification lacks for exploring and remains an open problem.

III. DUAL PATH CONVOLUTIONAL NEURAL NETWORK-RECURRENT NEURAL NETWORK CASCADE MODEL

This paper proposes a novel model for time series data classification. This section demonstrates the structure and the details of our model. An overview of the framework of DPCRCN can be seen in Figure 4.

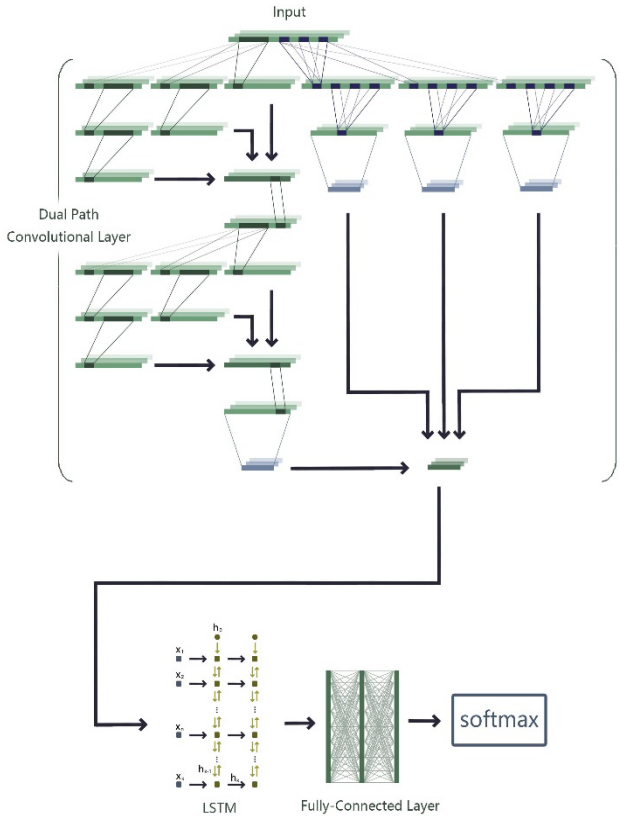

FIGURE 4. Framework of DPCRNCN.

DPCRCN contains three parts: dual path convolutional neural network, LSTM and fully-connected layers.

These three components will be introduced in the following part.

A. DUAL PATH CONVOLUTIONAL NEURAL NETWORK

The dual path convolutional neural network contains two paths which is shown in Figure 4. We design two blocks that are used in the two paths called One-Dimension

Inception Block (ODIB) and Multi-Dilated Convolutional Block (MDCB). In the following part, we will give the details about the two blocks.

1) ONE-DIMENSION INCEPTION BLOCK

ODIB is inspired by Inception-Resnet. The structure can be seen in Figure 5.

In Figure 5, B means *batch_size*, C means *in_channels* and L means *in_length*.

It is apparently that time series data has a special format. Generally, time series data can be represented as a 2-D tensor, namely *num_data* × *num_feature* × *length*. *length* represents the size or the length of the feature while *num_feature* represents the number of features. Generally, the data are divided into batches, so for each batch, the shape of the data is *batch_size* × *num_feature* × *length*. We use one-dimension convolutional layers which is a basic component

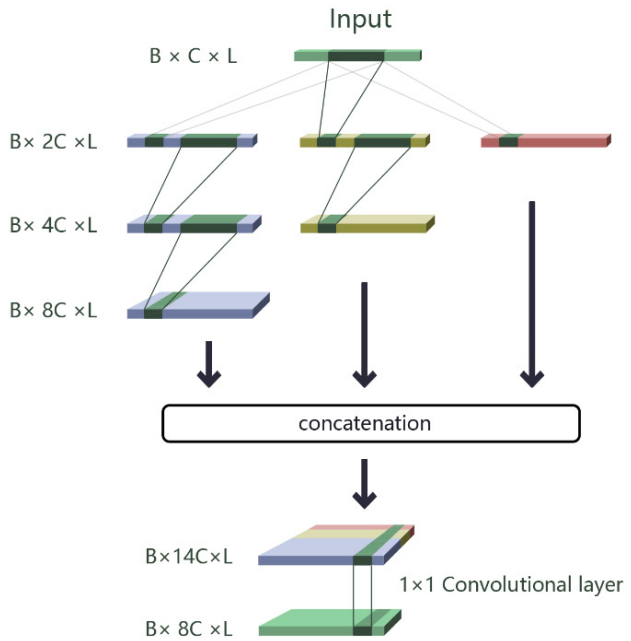

FIGURE 5. Illustration of ODIB.

of ODIB to extract features. The input format of a standard one-dimension convolutional layers is a 3-D tensor, namely *batch_size* × *in_channels* × *in_length*. We set *in_channels* equals to *num_feature* and *in_length* equals to *length*.

Each ODIB contains three paths with different numbers of convolutional layers. The kernel size of a single convolutional layer is 3, padding is 1, stride is 1. In this way, the *length* of the input data and output data remains the same. We set the output channel of the convolutional layers twice as the input channel. So assuming the shape of the input data is *batch_size* × *C_in* × *length*, the output of the three paths will be *batch_size* × 2*C_in* × *length*, *batch_size* × 4*C_in* × *length*, and *batch_size* × 8*C_in* × *length*. Each convolutional layer in three paths is followed by batch normalization layer and activation layer. The results are concatenated in the channel dimension. So, after concatenation, the shape of the output is *batch_size* × 14*C_in* × *length*. A short-cut which is commonly used in ResNet is also used in our structure. We use a 1x1 convolutional layer to expand the channels of the input to make it adaptable for addition. After the addition computation, we use a 1x1 convolutional layer to reduce the number of channels from 14*C_in* to 8*C_in*. The shape of the output of ODIB is *batch_size* × 8*C_in* × *length*.

In our experiment, we use two ODIBs in our network considering the concrete dataset. We firstly use two convolutional layers which kernel size is 5, padding is 1 and stride is 1. Then the data are fed into two ODIBs. After that, we use a one-dimension RoI pooling layer to adjust the length of the data in order to make our model practical for different shape of the input data.

2) MULTI-DILATED CONVOLUTIONAL BLOCK

In another path of the dual path convolutional neural network, we use MDCB. An illustration can be seen in Figure 6.

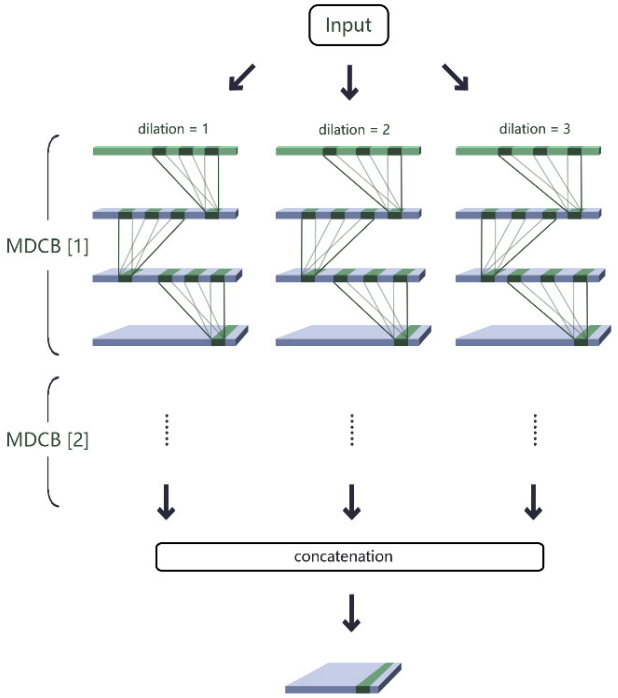

FIGURE 6. Illustration of MDCB.

Similar to ODIB, there are also three paths in a MDCB. The difference is that all the three paths have three convolutional layers. The kernel size of all the convolutional layers is 3, padding is 1 and stride is 1. The dilation of the convolutional layers is different in each path. We set the dilation equals to 1, 2 and 3 in different paths in order to have a multi-size receptive field. After three layers are the batch normalization layer and activation layer. The input data will pass through three dilated convolutional layers. The number of channels of the output will expand to eight times and the length will reduce at the same time. According to our dataset, we use 2 MDCBs. Different from ODIB, we use a one-dimension RoI pooling layer in each path to adjust the length of the output data of each path and then we concatenate them in channel dimension.

B. LONG SHORT-TERM MEMORY

The structure of the LSTM that used in this paper can be seen in Figure 7.

We use a two-layers bi-directional LSTM. For a basic LSTM cell, the inputs include two parts: X_k and h_{k-1}. h_{k-1} is the output of the last cell in the same layer.

The computation of LSTM is defined by the equation (1):

$$\begin{aligned} g^{(k)} &= \tanh(W^{gx}x_k + W^{gh}h_{(k-1)} + b^g) \\ i^{(k)} &= \sigma(W^{ix}x_k + W^{ih}h_{(k-1)} + b^i) \\ f^{(k)} &= \sigma(W^{fx}x_k + W^{fh}h_{(k-1)} + b^f) \\ o^{(k)} &= \sigma\left(W^{ox}x_k + W^{oh}h_{(k-1)} + b^o\right) \\ s^{(k)} &= g^{(k)} \odot i^{(k)} + s^{(k-1)} \odot f^{(k)} \\ h_{(k)} &= \tanh(s^{(k)}) \odot o^{(k)} \end{aligned} \tag{1}$$

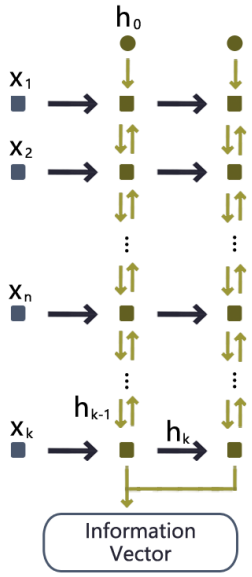

FIGURE 7. Illustration of LSTM.

where W^{gx}, W^{gh}, W^{ix}, W^{ih}, W^{fx}, W^{fh}, W^{ox}, W^{oh} and b^g, b^i, b^f, b^o are parameters to be learnt, $h_{(k)}$ is the output of LSTM. For multi-layer LSTM, $h_{(k)}$ is also the input of the next layer.

Generally, a standard input size of the LSTM layer is *batch_size* × *sequence_length* × *features* when we set the parameter batch_first equals to True. We transpose the

output of the dual path convolutional neural network from $batch\_size \times channel_{out} \times length_{out}$ to $batch\_size \times length_{out} \times channel_{out}$ where $channel_{out}$ represents the number of channels of the output and $length_{out}$ represents the size or the length of the features of the output. We then feed the data into LSTM. In this way, the LSTM both considers the information from a single channel and the relationship between different channels. We use the last hidden state h_k of the LSTM layer as the information vector which is fed into the fully-connected layers. To avoid overfitting, dropout is used, the invalid probability is 0.3.

C. FULLY-CONNECTED LAYERS

Finally, we apply fully-connected layers to the output of LSTM. In our model, we use two-layers fully-connected layers to learn the map between the information vector and the output. We also use dropout to prevent overfitting and the invalid probability is 0.5. We use softmax at the last layer. The output of softmax layer is the possibility of each label according to the input. Therefore, our model is practicable for multi-class time series data classification task.

IV. EXPERIMENT

This section will give the details of data preprocessing and the training process.

A. AREM DATASET PREPROCESSING

The dataset consists of 7 different kinds of activities (we treat bending 1 and bending 2 as two different kinds of activity). A small piece of data is shown in Table 1.

TABLE 1. A small piece of dataset of bending1 activity.

Avg_rss s12	Var_rss 12	Avg_rss s13	Var_rss 13	Avg_rss s23	Var_rss 23
39.25	0.43	22.75	0.43	33.75	1.3
39.25	0.43	23	0	33	0
39.25	0.43	23.25	0.43	33	0
39.5	0.5	23	0.71	33	0
39.5	0.5	24	0	33	0
39.25	0.43	24	0	33	0
39.25	0.43	24	0	33	0
39	0	23.75	0.43	33	0
39.5	0.5	24	0	33	0
39.5	0.5	23	0	33	0
39.5	0.5	23.25	0.43	33	0
39.5	0.5	23.5	0.5	32.75	0.43
39.5	0.5	23.75	0.43	32.5	0.5
39.67	0.47	23.75	0.43	33	0
39.5	0.5	24	0	33	0

As we can see in Table 1, there are 6 columns which represents 6 different kinds of features of the raw data. Avg means average and Var means variance. The whole sequence consists of 480 sequences. We use 7 temporal sequences of each activity (6 of bending2) as our dataset. For data augmentation, we divide the data into 416 pieces, each piece contains 64 sequences. Then the shape of a single input data is 6×64, where 6 is the number of features and 64 is the length. After data augmentation, we use 70% percent of the whole data for training and 30% percent for validating. We use Min-max normalization to compress the value of the features into [0,1]. Given a sequence $X = (x_1, x_2, \ldots c \ldots c, x_n)$, the min-max normalization formulation is as follows:

$$y_i = \frac{x_i - \min\ X}{\max\ X - \min\ X} \tag{2}$$

then the new sequence $Y = (y_1, y_2, \ldots c \ldots c, y_n) \in [0, 1]$.

We store the maximum value $\max X$ and the minimum value $\min X$ of the training set and use the two values for min-max normalization in the validation set.

After data preprocessing, we have 13977 data for training and 5991 data for validating. I.e. the shape of the training set is $13977 \times 6 \times 64$ and the shape of the validation set is $5991 \times 6 \times 64$. We set batch size equals to 32. Then the shape of each batch is $32 \times 6 \times 64$.

B. TRAINING PROCESS AND EXPERIMENT RESULT

In our experiment, we use Adam as our optimizer. All the activation function is ReLU. We set learning rate equals to 1e-2, and the learning rate will multiply 0.1 after each 200 epochs. We train our model for 800 epochs totally. Accuracy, recall rate and F1 score is used to evaluate the performance as they are commonly used for evaluating the result of classification. After we finished training our model, we examine our model on validation dataset for 20 epochs and we use the average value of the accuracy, recall rate and F1 score to assess our model.

We compare the performance of our model with LI-ESN and IDNNs and some popular machine learning

TABLE 2. Experiment results of different algorithms.

Algorithm	Accuracy	Recall Rate	F1 Score
DPCRCN	**99.97**	**99.91**	**99.94**
LR	60.37	60.54	60.21
SVM (RBF)	72.75	72.77	72.57
SVM (Poly)	63.89	62.19	61.04
SVM (GK)	66.42	66.57	66.15
RF	89.04	88.69	88.81
XgBoost	96.73	96.64	96.68
Dual Path Convolutional network	95.65	95.54	95.33
LSTM (6 layers)	72.23	71.37	67.91
LI-ESN	98.80	N/A	95.60
IDNNs	96.90	N/A	88.50

methods such as LR, SVM, RF and XgBoost. Only accuracy and F1 score of LI-ESN and IDNNs are given by the developer.

XgBoost and RF are two ensemble learning methods that use multiple trees for decision making. In statistics, LR is used to explain the relationship between one dependent binary variable and one or more independent variables. The prediction is based on using a logistic function with the linear combination of the features as the input and the outcome is a real value between 0 and 1 which can be interpreted as a probability. The purpose of SVM is to find a surface that maximize the margin between the surface and the support vectors. It is a popular algorithm for both classification and regression. We tried SVM with different kernels including radial basis function (RBF), polynomial (Poly) and Gaussian kernel (GK) in our experiment. Besides, we train our dual path convolutional network and LSTM separately to show why we need cascade model. In our DPCRCN model, the number of layers of LSTM is 2. Because of the feature extraction ability of dual path convolutional network, we do not need a very deep LSTM for classification. We change the depth of LSTM to 6 layers in order to have a better performance. The results are shown in Table 2.

To evaluate the generalization and reliability for different shape of time series data, we also divide the validation data into pieces that contains 38 sequence, 40 sequence, 44 sequence and 128 sequence. That means that we examine our model with the data which have a completely different shape from the training data and have not been used to trained the network. We use the five new validation sets to examine our trained model. The results are shown in Table 3.

TABLE 3. Experiment results of five different validation sets.

Algorithm	Accuracy	Recall Rate	F1 Score
Data with 38 sequences	98.85	98.73	98.65
Data with 40 sequences	99.22	99.17	99.01
Data with 42 sequences	99.59	99.57	99.52
Data with 44 sequences	99.79	99.77	99.73
Data with 128 sequences	100.00	100.00	100.00

V. DISCUSSION

As shown in Table 2, our model achieves the best performance since our accuracy, recall rate and F1 score is the highest among all the models. And the cascade structure has a positive affection on the performance on time series data classification because the cascade structure outperforms dual path convolutional network and LSTM. Comparing with other models, because of the utilization of RoI pooling layer, our model is flexible for data with different shape. Even facing the time series data contains different number of sequences and haven't been used to train our model, the performance is still satisfied. So, we believe that the proposed method is competitive and reliable.

Two-dimension CNN has made huge achievements in image processing tasks. That inspired the researchers to furtherly use CNN to address more problems. Xu *et al.* [37] firstly tried three-dimension CNN for temporal activity detection. Kamnitsas *et al.* [38] proposed three-dimension CNN for accurate brain lesion segmentation. Gehring *et al.* [39] used CNN instead of RNN in sequence-to-sequence model to accelerating the training process. However, comparing with Two-dimension CNN and three-dimension CNN, application about one-dimension CNN is much fewer. We consider two challenges. Firstly, it is difficult to choose a proper kernel size of the convolutional layer. Secondly, one-dimension CNN is likely to lose the relationship information between different variables or features.

For the first challenge, we design the dual path convolutional neural network. The basic idea of the network is using different convolutional layer to make the model have multi-size receptive field. In the aspect of signal processing, the time series data contains temporal information and frequency information. Fourier transformation [40] is a traditional and typical method which can exact frequency information. The computation of Fourier transformation is defined as equation (3):

$$F(\omega) = \mathcal{F}[f(t)] = \int_{-\infty}^{\infty} f(t)\, e^{-iwt} dt \tag{3}$$

the basic idea of Fourier transformation is using different trigonometric functions to match the original signal to obtain the frequency information. However, Fourier transformation suffers for poorly extracting information changing with time. To improve the shortcoming of Fourier transformation,

wavelet transformation [41] is developed. The computation of wavelet transformation can be seen in equation (4):

$$WT\ (a, \tau) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f\ (t)\ \psi(\frac{t-\tau)}{a} dt \tag{4}$$

in the equation, we can see that different from Fourier transformation, the wavelet transformation has two variables: a (scale) and τ (translation), a controls the stretch and shrink of the wavelet function and τ controls the translation. In this way, the wavelet function is able to extract flexible frequency information. Its main characteristic is that it can fully highlight some aspects of the frequency information and focus on any detail of the signal through transformation. Thus, it can solve the problem of Fourier transformation. Wavelet transformation is a major breakthrough in scientific methods since the Fourier transformation and it has been an ideal tool for signal time-frequency analysis and processing. We believe one-dimension convolution is kind of similar to transformation. One-dimension convolution filter is sensitive to a certain kind of changing regulation which represents the temporal information. So, based on the thought of wavelet transformation, we carefully design our dual path convolutional neural network. We use multiple convolutional layers in our model in order to have a various size of receptive field to capture and utilize more information than using single convolutional layer. We believe that adding more paths or using deeper neural networks is helpful according to the concrete situation of tasks and dataset. Besides, thanks to the utilization of RoI pooling layer, our model is capable for a changeable shape of data thus making our model more competitive than alternatives.

For the second challenge, we use CNN-RNN cascade model. We concatenate all the output of dual path convolutional neural network in channel dimension so each channel contains different information. We use RNN to analyze the information in each channel and the relationship between different channels. Deep RNN usually has a good performance but training a deep RNN is extremely time-consuming. CNN is capable for feature extraction and training a CNN is faster thanks to its characteristics. By the utilization of CNN, we do not need to train a very deep RNN to have a satisfied performance. The results show that cascade model can make use of the advantages of CNN and RNN to build a reliable and powerful end-to-end classifier to address time series data classification problems.

Last decade, data-driven methods especially deep learning methods have achieved huge successes. The applications based on deep learning are everywhere and they have strongly influenced our daily life. This paper proposes a novel deep learning architecture to tackle time series data. Our experiment results show that our DPCRCN is a competitive and reliable method for time series data classification.

VI. CONCLUSION AND FUTURE WORK

The problem of time series data classification can be seen in a various domain. To solve that problem, we proposed a deep learning architecture which is feasible for both multi variate time series data and single variate time series data. In our model, we use a dual path convolutional neural network to extract the features of the input data and use LSTM and the fully-connected layers to learn the map between the extracted features and the output. Our model, called DPCRCN, outperforms other alternatives. In the future, we are going to investigate more novel architectures both for time series data classification and time series data prediction. Other popular structures that are used in image processing will be considered such as DenseNet and SPPNet.

Besides, we also focus on the explainability of one-dimension convolutional neural networks. The stimuli of the filters of convolutional layers may suggest some essential information. It is likely that the understanding of stimuli is helpful for us to better comprehend the time series data resulting in a positive influence on decision making.

In addition, the method of data fusion is still a question. Simply concatenating the output of convolutional layers in channel dimension may not be the best way. What's more, it is likely to lose information if we just take the hidden state of the last time step of LSTM as information vector. So, we may explore a better solution for data fusion in the future.

REFERENCES

[1] R. Chitra and V. Seenivasagam, "Heart disease prediction system using supervised learning classifier," *Bonfring Int. J. Softw. Eng. Soft Comput.*, vol. 3, no. 1, pp. 1–7, 2013.

[2] S. Scher, "Toward data-driven weather and climate forecasting: Approximating a simple general circulation model with deep learning," *Geophys. Res. Lett.*, vol. 45, no. 22, pp. 12,616–12,622, 2018.

[3] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Z. Yang, "Automatic device classification from network traffic streams of Internet of Things," in *Proc. IEEE 43rd Conf. Local Comput. Netw. (LCN)*, Oct. 2018, pp. 1–9.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[5] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, Mar. 1996.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[10] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2117–2125.

[11] A. Show, "Tell: Neural image caption generation with visual attention," *Kelvin Xu et. al. arXiv Pre-Print*, p. 23, 2015.

[12] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Comput.*, vol. 1, no. 2, pp. 270–280, Jun. 1989.

[13] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: https://arxiv.org/abs/1409.0473

[14] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[15] R. Rana, "Gated recurrent unit (GRU) for emotion classification from noisy speech," 2016, *arXiv:1612.07778*. [Online]. Available: https://arxiv.org/abs/1612.07778
[16] F. Palumbo, P. Barsocchi, C. Gallicchio, S. Chessa, and A. Micheli, "Multisensor data fusion for activity recognition based on reservoir computing," in *Proc. Int. Competition Evaluating AAL Syst. Through Competitive Benchmarking*. Berlin, Germany: Springer, 2013, pp. 24–35.
[17] F. Palumbo, C. Gallicchio, R. Pucci, and A. Micheli, "Human activity recognition using multisensor data fusion based on reservoir computing," *J. Ambient Intell. Smart Environ.*, vol. 8, no. 2, pp. 87–107, 2016.
[18] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowl. Inf. Syst.*, vol. 7, no. 3, pp. 358–386, 2005.
[19] G. A. Susto, A. Schirru, S. Pampuri, and S. McLoone, "Supervised aggregative feature extraction for big data time series regression," *IEEE Trans. Ind. Informat.*, vol. 12, no. 3, pp. 1243–1252, Jun. 2016.
[20] R. D. Telford, S. Galloway, B. Stephen, and I. Elders, "Diagnosis of series DC Arc faults-A machine learning approach," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1598–1609, Aug. 2017.
[21] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[22] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[23] T. Chen and T. He, "xgboost: eXtreme gradient boosting," Tech. Rep., 2016.
[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 1–9.
[25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2818–2826.
[26] C. Szegedy, S. Ioffe, A. A Alemi, and V. Vanhoucke, "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. AAAI Conf. Artif. Intell.*, Feb. 2017.
[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: https://arxiv.org/abs/1502.03167
[29] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" 2018, *arXiv:1805.11604*. [Online]. Available: https://arxiv.org/abs/1805.11604
[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[31] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," 2016, *arXiv:1511.07122*. [Online]. Available: https://arxiv.org/abs/1511.07122
[32] Z. Cui, W. Chen, and Y. Chen, "Multi-scale convolutional neural networks for time series classification," 2016, *arXiv:1603.06995*. [Online]. Available: https://arxiv.org/abs/1603.06995
[33] S. Yi, J. Ju, M.-K. Yoon, and J. Choi, "Grouped convolutional neural networks for multivariate time series," 2017, *arXiv:1703.09938*. [Online]. Available: https://arxiv.org/abs/1703.09938
[34] C.-L. Liu, W.-H. Hsaio, and Y.-C. Tu, "Time series classification with multivariate convolutional neural network," *IEEE Trans. Ind. Electron.*, vol. 66, no. 6, pp. 4788–4797, Jun. 2019.
[35] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[36] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Oct. 2014, pp. 1724–1734.
[37] H. Xu, A. Das, and K. Saenko, "R-C3D: Region convolutional 3D network for temporal activity detection," 2017, *arXiv:1703.07814*. [Online]. Available: https://arxiv.org/abs/1703.07814
[38] K. Kamnitsas, C. Ledig, V. F. J. Newcombe, J. P. Simpson, A. D. Kane, D. K. Menon, D. Rueckert, and B. Glocker, "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation," *Med. Image Anal.*, vol. 36, pp. 61–78, Feb. 2017.
[39] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," 2017, *arXiv:1705.03122*. [Online]. Available: https://arxiv.org/abs/1705.03122?source=post_page
[40] R. N. Bracewell, *The Fourier Transform & Its Applications*. New York, NY, USA: McGraw-Hill, 1986.
[41] L. Debnath, "Wavelet transforms and their applications," *Phys. Today*, vol. 56, no. 4, p. 68, 2003.

CHAO YANG was born in China, in 1993. He received the B.S. degree in electronic information and communication engineering from Chongqing University, Chongqing, China, in 2016. He is currently pursuing the master's degree with the Department of Computer Science and Technology, Ocean University of China. His current research interests include machine learning, computer vision, natural language processing, and artificial intelligence.

WENXIANG JIANG was born in China, in 1995. He received the dual B.S. degrees in civil engineering and finance from Shandong Agriculture University, Shandong, China, in 2017. He is currently pursuing the master's degree with the Department of Computer Science and Technology, Ocean University of China. His current research interests include machine learning, sensor autocommunication, and artificial intelligence.

ZHONGWEN GUO was born in China, in 1965. He received the B.S. degree in computer science and technology from Tongji University, Shanghai, China, in 1987, and the M.S. and Ph.D. degrees from the Ocean University of China, Qingdao, China. He is currently a Professor and a Doctoral Advisor with the Department of Computer Science and Technology, Ocean University of China. His current research interests include sensor networks, distributed measurement systems, and ocean monitoring.