Jia JG, Zhou YF, Hao XW *et al.* Two-stream temporal convolutional networks for skeleton-based human action recognition. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 35(3): 538–550 May 2020. DOI 10.1007/s11390-020-0405-6

# Two-Stream Temporal Convolutional Networks for Skeleton-Based Human Action Recognition

Jin-Gong Jia[1], Yuan-Feng Zhou[1,*], *Senior Member, CCF*, Xing-Wei Hao[1], Feng Li[1], Christian Desrosiers[2], and Cai-Ming Zhang[1], *Senior Member, CCF*

[1] *School of Software, Shandong University, Jinan 250101, China*

[2] *Department of Software and IT Engineering, University of Quebec, Montreal H3C 3P8, Canada*

E-mail: jingongjia@mail.sdu.edu.cn; {yfzhou, hxw, lifeng6699}@sdu.edu.cn; Christian.Desrosiers@etsmtl.ca
czhang@sdu.edu.cn

Received February 29, 2020; revised April 5, 2020.

**Abstract** With the growing popularity of somatosensory interaction devices, human action recognition is becoming attractive in many application scenarios. Skeleton-based action recognition is effective because the skeleton can represent the position and the structure of key points of the human body. In this paper, we leverage spatiotemporal vectors between skeleton sequences as input feature representation of the network, which is more sensitive to changes of the human skeleton compared with representations based on distance and angle features. In addition, we redesign residual blocks that have different strides in the depth of the network to improve the processing ability of the temporal convolutional networks (TCNs) for long time dependent actions. In this work, we propose the two-stream temporal convolutional networks (TS-TCNs) that take full advantage of the inter-frame vector feature and the intra-frame vector feature of skeleton sequences in the spatiotemporal representations. The framework can integrate different feature representations of skeleton sequences so that the two feature representations can make up for each other's shortcomings. The fusion loss function is used to supervise the training parameters of the two branch networks. Experiments on public datasets show that our network achieves superior performance and attains an improvement of 1.2% over the recent GCN-based (BGC-LSTM) method on the NTU RGB+D dataset.

**Keywords** skeleton, action recognition, temporal convolutional network (TCN), vector feature representation, neural network

## 1 Introduction

Human action recognition is an important and challenging research problem in computer vision. At present, this problem has a wide range of applications in the fields of video surveillance, somatosensory games, patient monitoring, intelligent security, human-machine interaction, and robotics [1–3]. In addition, human action recognition and crowd evacuation algorithms [4] can be combined to analyze crowd behaviors and provide navigation. With the development of data acquisition devices, dynamic human skeleton sequences can be efficiently obtained. Therefore, it is highly desired to design a framework to encode both the spatial and temporal changes in human data.

Initial work on human action recognition focused on RGB videos [5,6]. Hou *et al.* [7] proposed a method to extract the key-frames of videos. Sermanet *et al.* [8] proposed a self-supervised approach for learning representations and robotic behaviors entirely from unlabeled videos. However, it is non-trivial to capture the complete human skeleton transitions in the 3D space due to

Regular Paper

Special Section of CVM 2020

The work was supported by the National Natural Science Foundation (NSFC)-Zhejiang Joint Fund of the Integration of Informatization and Industrialization of China under Grant Nos. U1909210 and U1609218, the National Natural Science Foundation of China under Grant No. 61772312, and the Key Research and Development Project of Shandong Province of China under Grant No. 2017GGX10110.

*Corresponding Author

the lack of depth channel in the source images/videos. With the innovation of 3D data acquisition technology, RGB-D data has become popular in recent years, which makes it possible to infer the motion sequence of a skeletal joint in the 3D space. For example, Sholl *et al.*[9] proposed an algorithm for obtaining human skeletons in real time with a depth sensor. Wang *et al.*[10] also proposed an efficient and robust human pose estimation algorithm on RGB videos. Significant advances have been made in human action recognition based on RGB and RGB-D data[11,12]. With the increasing availability of skeleton acquisition tools, research on human action recognition using skeleton data has generated growing interest.

In this paper, we simultaneously consider the spatial and temporal changes of the human skeleton and propose a more powerful learning model to capture skeleton variability in both spatial and temporal dimensions. Most existing methods lack the ability to extract the spatiotemporal feature representations. In such methods, it is often difficult to extract a single feature representation that can be used to recognize all action classes. Designing a model with a greater learning ability for spatiotemporal feature representations is also a key problem in human action recognition. Previous methods for identifying human actions are mainly based on convolutional neural networks (CNNs)[13–15], recurrent neural networks (RNNs)[16–19], or graph convolutional networks (GCNs)[20–23]. Typically, these methods only consider a single feature representation of the human body. In recent years, the temporal convolutional networks (TCNs)[24,25] have shown outstanding ability in processing time sequence data, and extensive experiments have shown that TCNs are superior to RNNs such as Long Short Term Memory networks (LSTMs). Based on TCNs, designing a multi-channel network model that learns multiple feature representations simultaneously can improve the accuracy of human action recognition. We consider two important feature representations in the new network, i.e., the movements of each skeletal joint between two adjacent action frames and the relative positions of the constituent joints in a single skeletal frame. The main contributions of our work include the followings.

- We propose a novel method that leverages both the inter-frame vector feature representation between adjacent frames and the intra-frame vector feature representation within a single frame. Experiments show that these two vector feature representations play the role of mutual promotion in recognition of many action classes.
- We redesign residual blocks for TCNs and propose the two-stream temporal convolutional networks (TS-TCNs) that can integrate multiple feature representations to bring notable improvement in recognition performance.
- We perform a comprehensive experimental validation using four widely well-known datasets: NTU RGB+D[11], NTU RGB+D 120[26], Northwestern-UCLA[27], and UTKinect-Action[28]. Our results show the proposed two-stream network achieves superior performance compared with most previous methods.

## 2 Related Work

In this section, we review relevant literature on human action recognition. First, we present methods for extracting the dynamics feature representation of human actions. We then describe network-based models to process skeleton sequences for human action recognition.

### 2.1 Dynamics Representation

The human action recognition task consists in identifying human body behaviors from sequence data such as images, videos, and skeletons. The main contents of action behaviors include gestures, actions in daily life, interaction and group activities. Early research on human action recognition focused on still images and videos[5,12]. RGB data is rich in color, shape, and texture features. Initial methods for action recognition mainly use the color and texture information in 2D images. However, various factors, such as background clutter and human body occlusion, make this identification task complicated. Liu *et al.*[29] proposed a method based on deep learning that uses depth sequences and the corresponding skeleton joint information. Since depth images lack information such as color and texture, related work based on depth maps is limited. Wang *et al.*[30] proposed a method using RGB and depth features to coordinate training for action recognition. Skeleton data, which has obvious advantages over RGB and depth data, contains 3D information on the joint points of the human body and thus provides higher-level geometric features. Wang *et al.*[31] developed an action ensemble model that characterizes the conjunctive structure of 3D human actions by capturing the correlations of the joints. Zhang *et al.*[32] introduced a related geometric feature on joints and selected lines. Liu *et al.*[16] proposed a more powerful tree-structure based traversal method. Zhang *et al.*[33] proposed a novel view adaptation scheme to let the network selected by itself the most suitable observation viewpoints. Ke *et al.*[34] proposed a method to transform a skeleton sequence into three clips, and then used a multi-task learning network. Ghorbel *et al.*[35] adopted a learning method using a Log-Euclidean distance and kernel methods such as support vector machines and multiple kernel learning to classify actions. The images and skeletons can be obtained by devices such as the Microsoft Kinect. The 3D skeletons are outputted by the corresponding depth images. In general, a 3D skeleton has 25 joints or 20 joints.

### 2.2 Skeleton-Based Networks on Human Action Recognition

We review existing work on human action recognition using deep networks. We can divide the proposed models into three categories: CNN-based models[13–15], RNN-based models[16–19], and GCN-based models[20–23]. Many previous methods are based on CNN. Huang *et al.*[13] adopted a CNN called LieNet and designed a rotation mapping layer and a rotation pooling layer. Ke *et al.*[14] converted the skeleton features to the image and fed the converted data into a deep CNN. Weng *et al.*[15] applied Naive-Bayes Mutual Information Maximization (NBMIM)[36] to CNN for recognizing actions. The majority of RNN models for action recognition are based on LSTM. Liu *et al.*[16] introduced a new gate mechanism in LSTM to learn action sequences and proposed a framework based on tree-structure traversal. Lee *et al.*[17] designed a temporal sliding LSTM that includes short-term, medium-term, and long-term units. Zhang *et al.*[18] proposed an element-wise-attention gate to empower the RNN's attentiveness capability. Attention-based recurrent neural networks[37] also have some applications in active object recognition. Meng *et al.*[19] proposed a sample fusion model that is combined with an LSTM autoencoder. Their main improvement is a data augmentation model to extend the original training dataset. Graph convolutional networks also led to recent progress in human action recognition. Yan *et al.*[20] proposed a graph convolutional network and presented several partitioning strategies to construct convolution operations. Li *et al.*[22] combined the actional and structural links into a generalized skeleton graph to learn more features for action recognition. Si *et al.*[23] proposed an attention enhanced graph convolutional LSTM network for action recognition. However, these GCN methods need to rely on a fixed graph topology and cannot utilize the inter-frame and intra-frame vector feature representations that we design. Recently, TCNs[24, 25] have shown outstanding performance in handling time sequences in various tasks. Compared with the above models, we propose a two-stream temporal convolutional network (TS-TCN) with better learning ability for spatiotemporal skeleton sequences.

## 3 Vector Feature Representations of Skeleton Sequence

In this section, we introduce the feature representation of the proposed network, including the extraction of spatial variation of the skeletal joints in the time dimension and the spatial feature representation of the skeletal joints in each skeleton frame.

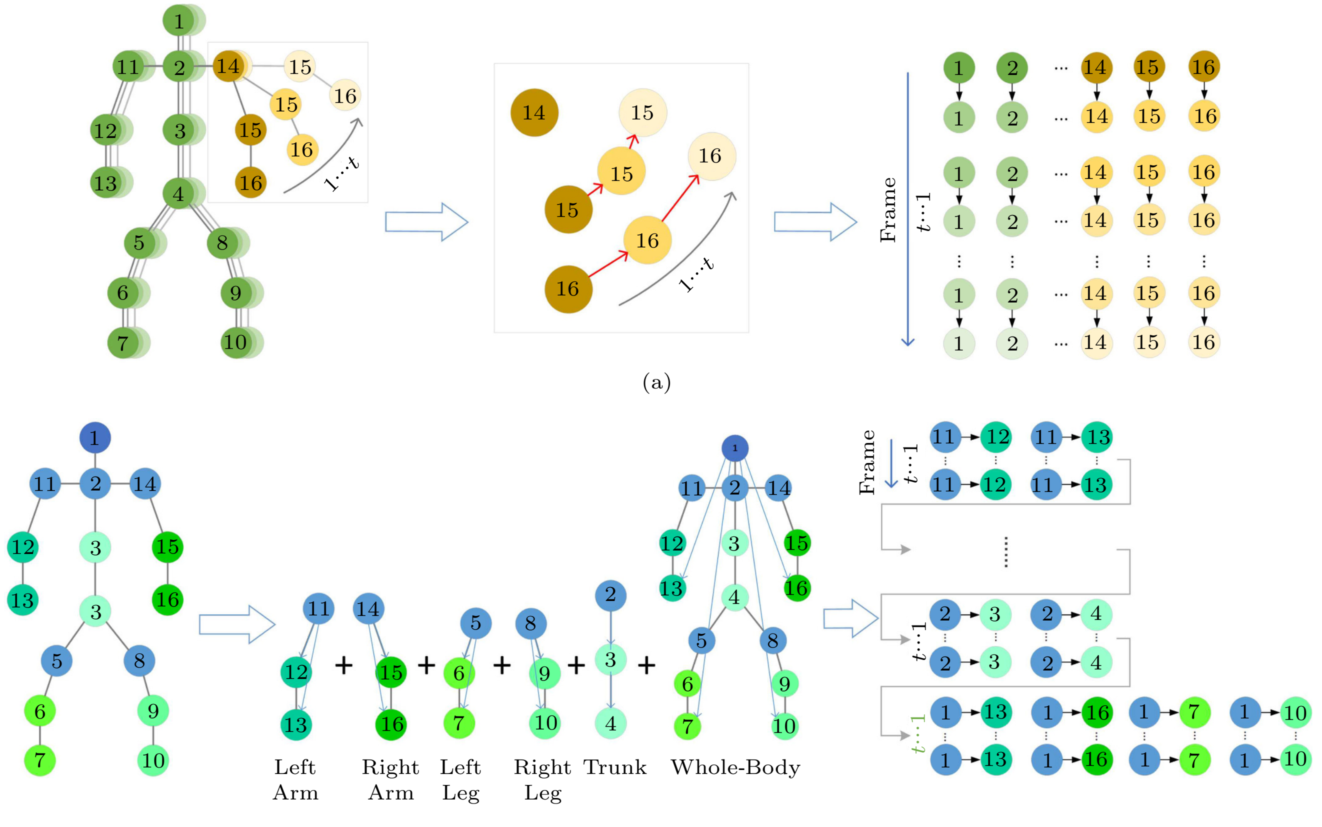
In our proposed network, the input data is 3D skeleton sequences that can be captured by RGB-D cameras. A single skeletal frame comprises a set of skeletal joints defined as 3D points and connection relationships between joints. Here, we take the 16-joint skeleton sequence as an example to explain the definition of the input vector feature representations.

As shown in Fig.1(a), converting an action sequence to an inter-frame vector feature representation makes it easier to capture varying joints. We can subtract the joints of the previous skeletal frame with the corresponding joints of the latter skeletal frame to get the inter-frame vector feature representation. The action shown in Fig.1(a) is "raising the arm". Based on the above inter-frame vector feature representation, we can make the network focus on the movement of certain joints. Let $\boldsymbol{s}_t^i \in \mathbb{R}^{1\times d}$ be the coordinates of the $i$-th joint of the $t$-th skeletal frame, where $d$ is the dimension of the skeletal joint. Moreover, $\boldsymbol{s}_t \in \mathbb{R}^{1\times D}$ denotes the coordinates concatenation of skeletal joints of the $t$-th frame:

$$\boldsymbol{s}_t = concat\big([\boldsymbol{s}_t^0, \boldsymbol{s}_t^1, \boldsymbol{s}_t^2, ..., \boldsymbol{s}_t^{\mathrm{J}-1}], 1\big),\ \forall t \in T,$$

where $concat([\text{elements}], k)$ and $k = 0, 1$ denote the concatenation along the $k$-th axis of the elements. Here, $T$ and J represent the total number of frames in an action sequence and the total number of joint points on each frame, respectively. $D = d \times \mathrm{J}$. The inter-frame vector feature representation concatenation in Fig.1(a) can be obtained by

$$\boldsymbol{x}_t^{(\mathrm{a})} = \boldsymbol{s}_t - \boldsymbol{s}_{t-1},\ \forall t \in T,\ t > 0.$$



(a)

(b)

Fig.1. Vector feature representations of the skeleton sequence used in our proposed TS-TCN network. (a) Skeleton inter-frame vector feature representation, which shows the process to acquire the inter-frame vector feature representation. (b) Skeleton intra-frame vector feature representation, which shows the process to acquire the intra-frame vector feature representation. In (b), the basic skeletal joints of the left arm, the right arm, the left leg, the right leg, the trunk, and the whole-body are 11, 14, 5, 8, 2, 1, respectively.

We use inter-frame vector feature representation $\boldsymbol{x}_t^{(\mathrm{a})}$ as the first input feature representation of our network.

The feature representation in Fig.1(b) is extracted from each skeletal frame. Instead of using the original coordinates of the skeleton in each frame, the feature representation is extracted by dividing the skeleton into five main body parts. We divide the whole skeleton into five parts: the left arm, the left leg, the right arm, the right leg, and the trunk. We also extract features encoding the relationship between these five parts and select joint 1 as the basic point. These basic points usually move with a slight magnitude when a human performs actions, and thus can better capture the movement characteristics of the human skeleton. We represent the five parts including the left arm, the right arm, the left leg, the right leg, and the trunk as $\boldsymbol{s}_t^{(\mathrm{la})}$, $\boldsymbol{s}_t^{(\mathrm{ra})}$, $\boldsymbol{s}_t^{(\mathrm{ll})}$, $\boldsymbol{s}_t^{(\mathrm{rl})}$, $\boldsymbol{s}_t^{(\mathrm{t})}$, respectively. The relationship between these five parts is noted as $\boldsymbol{s}_t^{(\mathrm{fp})}$. For example, $\boldsymbol{s}_t^{(\mathrm{la})}$ is obtained by

$$\boldsymbol{s}_t^{(\mathrm{la})} = concat\big([\boldsymbol{s}_t^{12} - \boldsymbol{s}_t^{11}, \boldsymbol{s}_t^{13} - \boldsymbol{s}_t^{11}], 1\big),\ \forall t \in T,$$

as shown in Fig.1(b). Then, the intra-frame vector feature representation $\boldsymbol{x}_t^{(\mathrm{b})}$ is defined as

$$\boldsymbol{x}_t^{(\mathrm{b})} = concat\big([\boldsymbol{s}_t^{(\mathrm{la})}, \boldsymbol{s}_t^{(\mathrm{ra})}, \boldsymbol{s}_t^{(\mathrm{ll})}, \boldsymbol{s}_t^{(\mathrm{rl})}, \boldsymbol{s}_t^{(\mathrm{t})}, \boldsymbol{s}_t^{(\mathrm{fp})}], 0\big).$$

Finally, we get the two input feature representations of the network: $\boldsymbol{x}_t^{(\mathrm{a})}$ and $\boldsymbol{x}_t^{(\mathrm{b})}$. Note that we also test using the angle and the distance between the joint points of the skeleton as intra-frame feature representations. These preliminary experiments show no significant advantage over the feature representations in our proposed network.

## 4 Two-Stream TCNs Framework

In this section, we introduce the components of our proposed two-stream temporal convolutional networks (TS-TCNs) in detail.

### 4.1 Residual Block of Networks

Inspired by the residual block in Res-TCNs[25], we redesign a residual block with greater learning ability. In Res-TCNs, the residual block contains only one convolution layer, and each residual block is set to the same stride size. The residual block we propose contains two convolution layers that have the same number of convolution kernels and expand the receptive field as the network becomes deeper. The expansion of the receptive field is more favorable to long-term time-dependent actions. Additionally, we perform BatchNormalization[38], PReLU[39] activation before each convolution layer. The two convolution layers will make the residual block learn more discriminative features.

We use PReLU to implement nonlinear activations in the network. Compared with ReLU, PReLU adds a linear term to the negative input, and the slope of the linear term is learned in model training. In the PReLU layer, the network adds only a few parameters; therefore the computation of the network and the risk of over-fitting increase minimally. The PReLU activation formula is defined as follows:

$$PReLU(x) = \begin{cases} x, & \text{if } x > 0, \\ ax, & \text{if } x \leqslant 0. \end{cases}$$

Fig.2 shows our modified residual block. It uses a residual connection to enhance the accuracy and interpretability of the recognition of time sequence data. When the number of network layers increases, the residual connection can solve the problem of vanishing gradient and network degradation. The residual block of the $l$-th layer can be calculated by the following equation:

$$\boldsymbol{Y}_l = \boldsymbol{Y}_{l-1} + F\big(\boldsymbol{W}_{l,1}, \boldsymbol{W}_{l,2}, \boldsymbol{Y}_{l-1}\big),$$

where $\boldsymbol{Y}_{l-1}$ is the input of the $l$-th layer, and $\boldsymbol{W}_{l,1}$ and $\boldsymbol{W}_{l,2}$ represent the weights of the first convolution layer and the second convolution layer of block $l$, respectively.

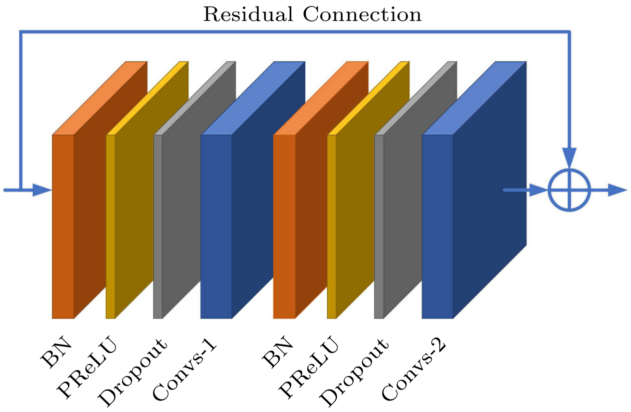


Fig.2. Residual block architecture we redesign. Convs-1 represents the first convolution layer and Convs-2 represents the second convolution layer. BN is the BatchNormalization layer.

The residual block $F$ is defined as

$$F(\boldsymbol{W}_{l,1}, \boldsymbol{W}_{l,2}, \boldsymbol{Y}_{l-1}) = \boldsymbol{W}_{l,2} \cdot \sigma\big(\boldsymbol{W}_{l,1} \cdot \sigma\big(\boldsymbol{Y}_{l-1}\big)\big),$$

with $\sigma$ representing the PReLU activation function.

### 4.2 Temporal Convolutional Networks Branch

In this subsection, we give a brief introduction to the modified TCN, which is originally designed for time-series tasks[24, 25]. We adopt the architecture of the temporal convolutional networks which is presented in [25]. However, our TCN is stacked up by 12 basic residual blocks and can learn longer-term historical information. The number of output channels for each block is 64, 64, 64, 128, 128, 128, 256, 256, 256, 512, 512 and 512, respectively. The improved TCN framework is shown in Fig.3. As can be seen, we add three residual blocks (B10, B11, B12) into the framework. The number of output channels in these three residual blocks is 512, 512, 512 and the stride length is 2, 1, 1, respectively. B10 adopts a convolution with a stride size of 2. This block can help TCN improve the recognition ability by getting high-level features.

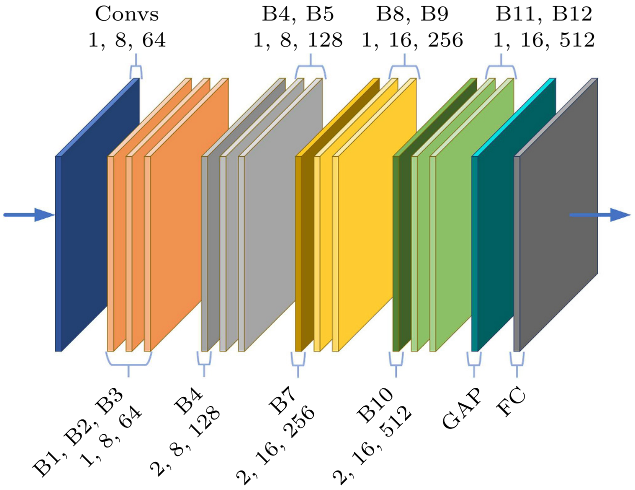


Fig.3. Illustration of the redesigned TCN. There are a total of 12 residual blocks (B1-B12). The three numbers of each block represent the stride, the size of the convolution kernels, and the number of convolution kernels, respectively. GAP represents the global average pooling layer. FC represents a fully connected layer.

The input of our TCN is a $D$-dimensional feature obtained from each skeletal frame. For each action, we splice the input $\boldsymbol{X}$ with features of all frames, denoted as $\boldsymbol{X} \in \mathbb{R}^{T\times D}$. Each non-linear activation is followed by a convolution layer so that features can be extracted step by step from the input data. Specifically, in the $l$-th convolution layer, there are $N_l$ convolution kernels with a size of $d_l$, and the set of all kernels is denoted by $\{\boldsymbol{W}^{(i)}\}_{i=1}^{N_l}$ where each convolution kernel is $\boldsymbol{W}^{(i)} \in \mathbb{R}^{d_l \times N_{l-1}}$. If the output of the upper block is $\boldsymbol{Y}_{l-1}$ and the next block has a stride of 2, the next block output is

$$\boldsymbol{Y}_l = \boldsymbol{W}_{l,2} \cdot (\boldsymbol{W}_{l,1} \cdot \boldsymbol{Y}_{l-1}) + F(\boldsymbol{W}_{l,1}, \boldsymbol{W}_{l,2}, \boldsymbol{Y}_{l-1}),$$

where the non-linear activation function is PReLU. TCN uses a backpropagation algorithm during the training process.

### 4.3 Two-Stream TCNs Architecture

We present a two-stream TCN architecture to train the inter-frame vector feature representation and intra-frame vector feature representation with two parallel TCNs. As shown in Fig.4, our overall network comprises two TCN branches, which are used to focus on learning two different vector feature representations presented in Section 3. Each TCN branch comprises a one-dimensional convolution and 12 residual blocks. BatchNormalization can improve the training speed of the network and avoid the problem of disappearing in the network. Moreover, dropout is used to avoid overfitting in the training process. At the end of the two-stream network, we use the Softmax layer for classification and employ cross-entropy over the two branches as loss function. In the testing phase, we take the average of the softmax output in each branch as our final output. The first layer of the two TCN branches in our network uses a one-dimensional convolution that acts on vector feature representations to generate activation map $\boldsymbol{Y}_1^{(\mathrm{a})}$. Assuming that each branch of the two-stream TCN has $M$ residual blocks, $\boldsymbol{Y}_1^{(\mathrm{a})}$ is given by

$$\boldsymbol{Y}_1^{(\mathrm{a})} = \boldsymbol{W}_1^{(\mathrm{a})} \cdot \boldsymbol{X}_n^{(\mathrm{a})}, \tag{1}$$

and the activation map generated by the $M$-th block is

$$\boldsymbol{Y}_{M+1}^{(\mathrm{a})} = \boldsymbol{Y}_1^{(\mathrm{a})} + \sum_{i=2}^{M+1} \boldsymbol{W}_{i,2}{}^{(\mathrm{a})} \cdot \sigma\big(\boldsymbol{W}_{i,1}{}^{(\mathrm{a})} \cdot \sigma(\boldsymbol{Y}_{i-1}^{(\mathrm{a})})\big),$$

where (a) represents the first TCN branch. Similarly, the activation map generated by the first layer in the second branch is

$$\boldsymbol{Y}_1^{(\mathrm{b})} = \boldsymbol{W}_1^{(\mathrm{b})} \cdot \boldsymbol{X}_n^{(\mathrm{b})},$$

and the activation map generated by the $M$-th layer in this branch is

$$\boldsymbol{Y}_{M+1}^{(\mathrm{b})} = \boldsymbol{Y}_1^{(\mathrm{b})} + \sum_{i=2}^{M+1} \boldsymbol{W}_{i,2}^{(\mathrm{b})} \cdot \sigma\big(\boldsymbol{W}_{i,1}^{(\mathrm{b})} \cdot \sigma(\boldsymbol{Y}_{i-1}^{(\mathrm{b})})\big),$$

where (b) represents the second TCN branch.

### 4.4 Loss Function

Taking the first TCN branch as an example, $\boldsymbol{Y}_1^{(\mathrm{a})}$ is the result with no nonlinear activation. The result of $\boldsymbol{W}_{i,2}^{(\mathrm{a})} \cdot \sigma(\boldsymbol{W}_{i,1}^{(\mathrm{a})} \cdot \sigma(\boldsymbol{Y}_{i-1}^{(\mathrm{a})}))$ is added to $\boldsymbol{Y}_{i-1}^{(\mathrm{a})}$, and all remaining residual blocks are added based on $\boldsymbol{Y}_1^{(\mathrm{a})}$ in (1). As shown in Fig.4, through our proposed Two-Stream TCNs, vector feature representations are integrated to improve the training accuracy. After the last residual block of each network branch, we apply a global average pooling followed by a Softmax layer in which the number of neurons is equal to the number of classes. In the training phase, the Softmax functions of the two TCN branches are thus

$$\boldsymbol{S}_c^{(\mathrm{a})} = \frac{\exp\left(\boldsymbol{z}_c^{(\mathrm{a})}\right)}{\sum\limits_{k=0}^{N_c-1} \exp\left(\boldsymbol{z}_k\right)}, \quad \boldsymbol{S}_c^{(\mathrm{b})} = \frac{\exp\left(\boldsymbol{z}_c^{(\mathrm{b})}\right)}{\sum\limits_{k=0}^{N_c-1} \exp\left(\boldsymbol{z}_k\right)},$$

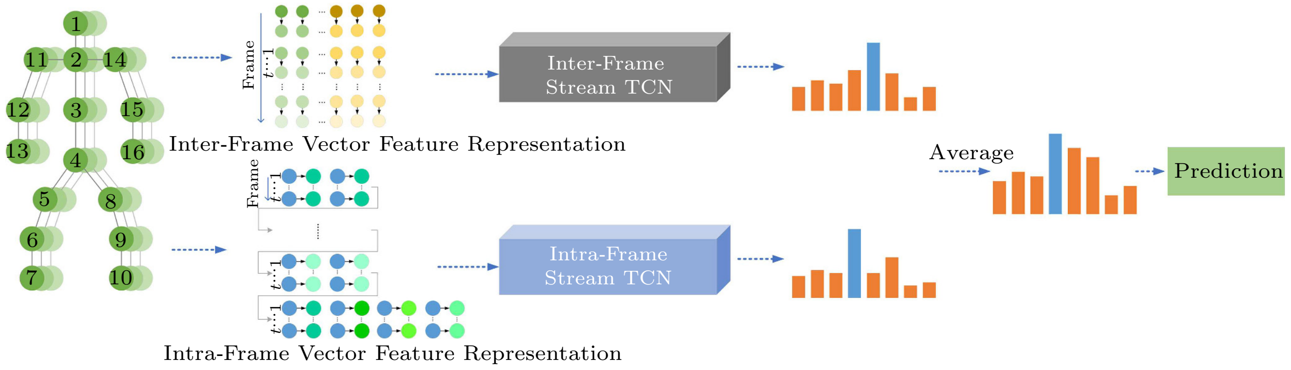


Fig.4. Illustration of the overall architecture of TS-TCN. The above branch network focuses on the learning of the inter-frame vector feature representation (Fig.1(a)), and the bottom branch network focuses on the learning of the intra-frame vector feature representation (Fig.1(b)). The scores of two streams are averaged to obtain the final prediction.

where $c$ and $N_c$ are the corresponding class index and the total number of action classification, respectively. Let $\boldsymbol{z}_k$ be the $k$-th action class values. Finally, the total loss is the sum of the loss values of the two branches, computed with cross-entropy:

$$L = -\sum_{k=0}^{N_m-1} \boldsymbol{y}_k \ln\left\{\boldsymbol{S}_k^{(\mathrm{a})} \cdot \boldsymbol{S}_k^{(\mathrm{b})}\right\},$$

where $N_m$ and $\boldsymbol{y}_k$ are the total number of samples in the training set and the ground-truth label of $k$-th sample respectively. During learning, we train our model by minimizing the loss function.

In the testing phase, we get the output with an ensemble average among the two linear activation values $\boldsymbol{S}_c^{(\mathrm{a})}$, $\boldsymbol{S}_c^{(\mathrm{b})}$. Finally, the output value and the ground-truth are used to calculate the loss value of the testing set.

## 5 Experiments

In this section, we validate our proposed approach and compare it with recent methods based on CNN, RNN, and GCN. Our experiments use four well-known action recognition datasets: NTU RGB+D [11], NTU RGB+D 120 [26], Northwestern-UCLA [27], and UTKinect-Action [28]. Next, we introduce these datasets and analyze the performance of our approach.

### 5.1 Dataset and Network Parameters

*NTU RGB+D Dataset.* This dataset is the most widely used and challenging dataset for action recognition and includes 60 types of single and double-person daily action. The evaluation method on this dataset includes two benchmarks: cross-subject (CS) evaluation, in which half the subjects are used as the training set and the other half is used as the testing set; cross-view (CV) evaluation, where the data collected on the second and third devices is used as the training set and the data collected on the first device is used as the testing set.

*NTU RGB+D 120 Dataset.* The dataset expands the NTU RGB+D dataset by adding 60 additional action classes and 57 600 additional action sequences. The addition of categories and changes in perspective makes this dataset even more challenging. For the huge amount of data, this dataset is more suitable for deep learning based on action recognition.

*Northwestern-UCLA Dataset.* This dataset contains 1 494 action sequences, including 10 daily action classes. Three Kinect V1.0 devices are used to collect each action sequence. All skeletons contain the 3D coordinates of 20 joint points. Each type of movements is collected by 10 different subjects. The challenge of the dataset is the repetitive actions of subjects from different directions.

*UTKinect-Action Dataset.* The dataset contains 199 skeleton action sequences that are captured by a stationary Kinect V1.0. The whole dataset contains 10 action classes which are completed by 10 different subjects, and each subject repeats every type of actions twice. This dataset is very challenging because of changes in viewpoints and changes within classes.

As described in Subsection 4.3, we divide the entire process into a training part and a testing part. In the training part, we use backpropagation to continuously optimize the network parameters after calculating the total loss value. In testing, we do not calculate the loss value in the last layer but use the ensemble average output of the two branch networks as the total output of the network. We use the Keras[1] deep learning framework with the TensorFlow [40] backend. We set the initial learning rate of the network to 0.01. When the total loss value does not change after 15 epochs, we change the learning rate to one half of its value. For all convolution layers, we set the $L_1$ regularizer with a weight of $1\mathrm{e}^{-4}$. Because the above four datasets have different numbers of samples, the batchsize used on the four datasets is selected as 128, 128, 128 and 16 respectively. We perform all our experiments on two NVIDIA GTX 2080 GPUs and with RAM of 128 G.

### 5.2 Ablation Study

In this subsection, we firstly use eight kinds of actions in cross-view on the NTU RGB+D dataset to examine the validity of our proposed Two-Stream TCNs (TS-TCNs). The accuracy of the original Res-TCN is 83.1%. By integrating the inter-frame and the intra-frame feature representations of the skeleton sequence and using our proposed Two-Stream TCNs, we achieve an improvement of 7.1%. The detailed confusion matrices are given in the supplementary files[2].

[1] https://github.com/fchollet/keras, Feb. 2020.

[2] Supplementary files. https://github.com/jingong/TS-TCN, Mar. 2020.

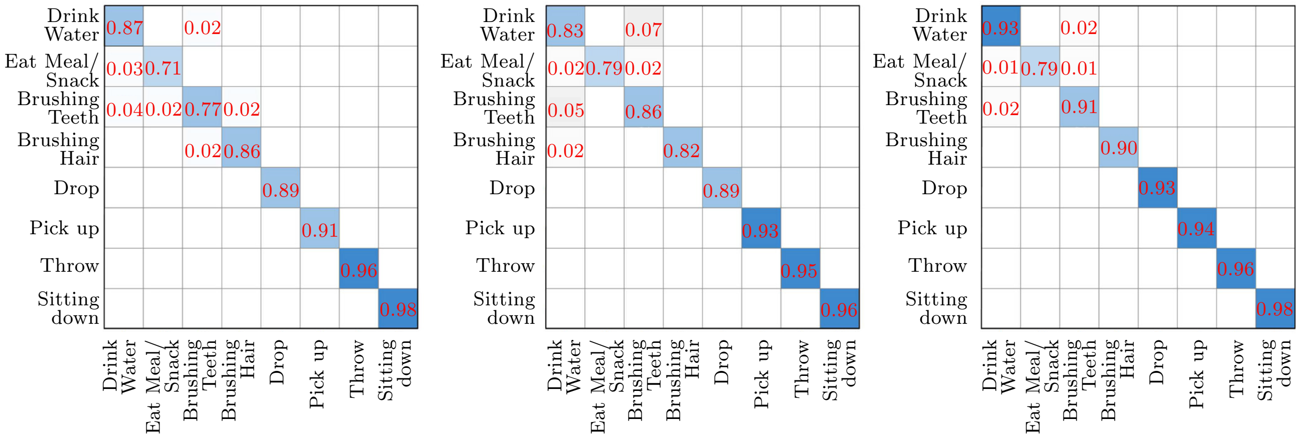
### *5.2.1 TS-TCNs Results*

To show the learning capabilities of our model more intuitively, we analyze the learning capabilities of each network branch and evaluate their accuracy on the testing set. Moreover, to analyze the proposed TS-TCNs in detail, we visualize the softmax outputs of the three parts containing the two branches' prediction and the ensemble average prediction on the NTU RGB+D dataset as depicted in Fig.5. The recognition accuracy of "drink water" in inter-frame stream (in Fig.5(a)) is higher than that in intra-frame stream (in Fig.5(b)), which shows that, in such action classes, the inter-frame feature representations have a greater impact on performance than the intra-frame feature representations. Similarly, "eat meal/snack" performs better on intra-frame stream (in Fig.5(b)) than on inter-frame stream (in Fig.5(a)), highlighting that intra-frame feature representations have a greater impact on performance than inter-frame feature representations. Obviously, inter-frame stream (in Fig.5(a)) will produce a lower misclassification in some actions than intra-frame stream (in Fig.5(b)), which makes it less likely for the model to overfit certain actions. For example, inter-frame has lower misclassification probabilities from the actions "drink water & brushing hair & throw & sitting down" to the actions "eat meal/snack & brushing teeth & pick up" than intra-frame, which demonstrates that inter-frame can compensate the weakness of intra-frame. Finally, as shown in Fig.5(c), all action classes perform better than any action in Fig.5(a) and Fig.5(b). Therefore, our Two-Stream TCN can use the complementary characteristics of inter-frame and intra-frame in different actions to improve recognition accuracy.

Section 3 mentions two different types of feature representations, one allowing the network to have a strong learning ability for changes in skeleton points over time, and one allowing the network to have a strong learning ability for the relative changes of the skeleton point in space. From Fig.5, we count the recognition accuracy of each branch and TS-TCN in action classes. By doing so, we can see the superiority of the proposed feature representations and the superiority of TS-TCN. These two critical feature representations in time and space are indispensable, playing a vital role in the improvement of network recognition capabilities. If either of the two branch networks is removed, the performance obtained by the network in each action class will not reach the current experimental results.

### *5.2.2 Comparison with Res-TCN*

In this subsection, we take Res-TCN [25] as the baseline. Firstly, we validate the learnability and rationality of inter-frame and intra-frame feature representations. As shown in "Two-Stream (Res-TCN)" in Table 1, we put the inter-frame and the intra-frame feature representations into Res-TCN and use our Two-Stream TCNs to improve the accuracy of CV from 83.1% to 87.4%. The bold numbers in all tables indicate the best experimental accuracy. From these results, we see that these two feature representations are more suitable for TCNs learning. Then, we validate the learning ability of the redesigned residual blocks. Each residual



(a) (b) (c)

Fig.5. Confusion matrices of our Two-Stream TCNs according to action set on NTU RGB+D. We select eight types of actions ("drink water", "eat meal/snack", "brushing teeth", "brushing hair", "drop", "pick up", "throw" and "sitting down") from the NTU RGB+D dataset to show the confusion matrix. The row and the column of each confusion matrix denote the ground truth and the prediction, respectively. (a) Inter-frame stream. (b) Intra-frame stream. (c) Two-stream. Because the value of the blank position in the confusion matrix is less than 0.01, we do not fill in the value.

block of Res-TCN contains a convolution layer, while our modified residual block contains two convolution layers. Increasing the number of convolution layers in residual blocks will further enhance the learning ability of the network. As shown in "Two-Stream (B1–B9)", the accuracy of CV can be improved from 83.1% to 88.1% by using our proposed residual blocks, which is a significant breakthrough in the learning ability of the TCNs. Finally, we verify the contribution of the last three residual blocks to the network. Similarly, as shown in "Two-Stream (B1–B12)", using our redesigned TCNs can further improve the accuracy of CV from 83.1% to 90.2%.

**Table 1**. Accuracy Comparison with Res-TCN on Dataset NTU RGB+D with CS and CV Benchmarks

| Method | CS (%) | CV (%) |
|---|---|---|
| Res-TCN [25] | 74.3 | 83.1 |
| Two-stream (Res-TCN) | 81.7 | 87.4 |
| Two-stream (B1–B9) | 82.0 | 88.1 |
| Two-stream (B1–B12) | **82.4** | **90.2** |

Note: (B1–B*k*) means residual block 1 to block *k*.

### *5.2.3 Two-Stream Networks Analysis*

The important improvement of recognition accuracy comes from the utilization of two kinds of feature representations encoding spatial and temporal information. These feature representations play a complementary role in the recognition of many action sequences. Here, we compare the performance of each type of individual input, as shown in the inter-frame stream and the intra-frame stream in Table 2. The implementation details are incorporated as described in Subsection 4.3. The results are shown in Table 2 as "two-stream". As can be seen, the two-stream method is superior to the inter-frame and the intra-frame methods with an accuracy improvement of 4%–5%.

**Table 2**. Accuracy Comparison with Different Streams on Dataset NTU RGB+D with CV Benchmark

| Method | Accuracy (%) |
|---|---|
| Inter-frame stream | 86.3 |
| Intra-frame stream | 85.2 |
| Two-stream | **90.2** |

### *5.2.4 Intra-Frame Feature Representations Analysis*

In this subsection, we validate the proposed intra-frame feature representations. Intra-frame features can be roughly divided into the angle [41], distance [32], vector, and other features. We use the angle of intra-frame joint formation given in [41], which is reported as intra-frame stream (angle). The distance between intra-frame joint points given in [32] is used to carry out the experiment, which is reported as intra-frame stream (distance). We experiment on the NTU RGB+D dataset and compare them with our intra-frame articulation point formation. Results in Table 3 show that our method is superior to these previous approaches [32, 41].

**Table 3**. Accuracy Comparison with Different Intra-Frame Feature Representations on Dataset NTU RGB+D with CS and CV Benchmarks

| Method | CS (%) | CV (%) |
|---|---|---|
| Inter-frame stream (angle) | 72.4 | 79.1 |
| Intra-frame stream (distance) | 73.6 | 82.2 |
| Our intra-frame stream (vector) | **78.0** | **85.2** |

## 5.3 Quantitative Comparison

*NTU RGB+D Dataset*. We perform experimental comparisons on the NTU RGB+D dataset. Our comparison focuses on existing methods based on CNN, RNN, and GCN. We examine the proposed two-stream network with inter-frame and intra-frame feature representations. Results show that the feature representations used in this paper achieve the best performance. In this dataset, we experiment with the accuracy of the algorithm with the cross-view (CV) and cross-subject (CS) benchmarks. Table 4 gives a comparison of the specific algorithm accuracy. Our method yields an accuracy of 7.1% higher than the latest TCN-based (Res-TCN [25]) approach and achieves better performance on inter-frame and intra-frame than Res-TCN. Furthermore, compared with the recent GCN-based (BGC-LSTM [42]) approach, our method is 1.2% higher with CV and 0.6% higher with CS.

**Table 4**. Accuracy Comparison with Skeleton-Based Action Recognition Methods on Dataset NTU RGB+D with CS and CV Benchmarks

| | Method | CS (%) | CV (%) |
|---|---|---|---|
| Previous methods | P-LSTM [11] | 62.9 | 70.7 |
| | TS-LSTM [17] | 74.6 | 81.3 |
| | Res-TCN [25] | 74.3 | 83.1 |
| | VA-LSTM [33] | 79.4 | 87.6 |
| | ST-GCN [20] | 81.5 | 88.3 |
| | ST-BBMIM [15] | 80.0 | 84.2 |
| | EleAtt-GRU [18] | 80.7 | 88.4 |
| | BGC-LSTM [42] | 81.8 | 89.0 |
| Ours | Inter-frame stream | 78.9 | 86.3 |
| | Intra-frame stream | 78.0 | 85.2 |
| | Two-stream | **82.4** | **90.2** |

*NTU RGB+D 120 Dataset.* On this dataset, the evaluation strategy is the same as the NTU RGB+D dataset. We implement two deep learning based approaches on NTU RGB+D 120, i.e., Res-TCN [25] and ST-GCN [20]. Table 5 gives a comparison of the specific algorithm accuracy. The accuracy of our method is 8.1% higher than the latest TCN-based (Res-TCN [25]) approach with the CV benchmark and achieves better performance on inter-frame and intra-frame than Res-TCN. Compared with the GCN-based approach, our performance is 1.9% higher than that of ST-GCN [20].

**Table 5.** Accuracy Comparison with Skeleton-Based Action Recognition Methods on Dataset NTU RGB+D 120 with CS and CV Benchmarks

| | Method | CS (%) | CV (%) |
|---|---|---|---|
| Previous | ST-LSTM [16] | 55.7 | 57.9 |
| methods | MTLN [34] | 58.4 | 57.9 |
| | Res-TCN [25] | 67.5 | 75.6 |
| | ST-GCN [20] | 71.4 | 81.6 |
| Ours | Inter-frame stream | 70.8 | 81.6 |
| | Intra-frame stream | 69.6 | 80.7 |
| | Two-stream | **71.9** | **83.5** |

*Northwestern-UCLA Dataset.* We adopt the training and testing strategies given in [27]. As shown in Table 6, the accuracy of our proposed method reaches 91.9%. Compared with the current best LSTM model, we achieve the best accuracy by improving 1.2%. We also implement the recently proposed and popular 2S-AGCN [21] method on this dataset and achieve the best accuracy of 86.9% through many experiments on its model. Although the 2S-AGCN method can achieve better performance on relatively large datasets, our experiments show that it cannot perform better on smaller datasets.

**Table 6.** Accuracy Comparison with Skeleton-Based Action Recognition Methods on Dataset Northwestern-UCLA

| | Method | Accuracy (%) |
|---|---|---|
| Previous | Actionlet ensemble [31] | 76.0 |
| methods | Ensemble TS-LSTM [17] | 89.2 |
| | EleAtt-GRU [18] | 90.7 |
| | TS+MSSFN [19] | 88.9 |
| | 2S-AGCN [21] | 86.9 |
| Ours | Inter-frame stream | 89.0 |
| | Intra-frame stream | 90.3 |
| | Two-stream | **91.9** |

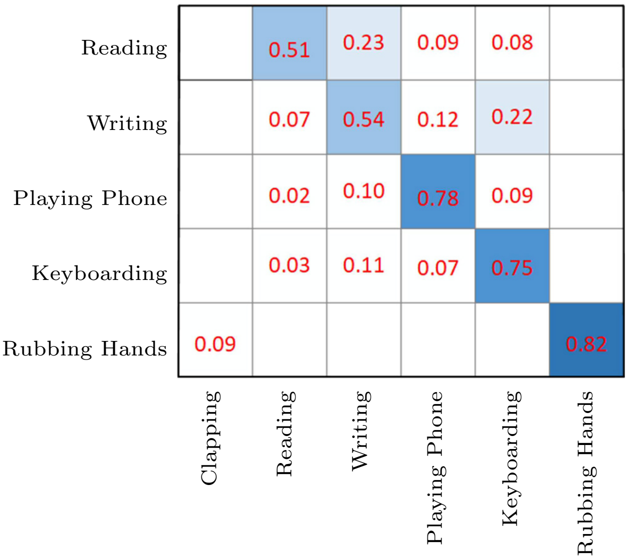
*UTKinect-Action Dataset.* We adopt the evaluation strategy employed in [43], with half of the subjects as the training set and the other half as the testing set. As shown in Table 7, our model obtains the best accuracy of 97.1%, an improvement over the previous accuracy of 97.0%.

**Table 7.** Accuracy Comparison with Skeleton-Based Action Recognition Methods on Dataset UTKinect-Action

| | Method | Accuracy (%) |
|---|---|---|
| Previous | Actionlet ensemble [31] | 90.9 |
| methods | ST-LSTM [16] | 97.0 |
| | Geometric features [32] | 96.0 |
| | Ensemble TS-LSTM [17] | 97.0 |
| | HKC+MKL-MLE [35] | 95.0 |
| Ours | Inter-frame stream | 90.2 |
| | Intra-frame stream | 92.2 |
| | Two-Stream | **97.1** |

### 5.4 Failure Cases

To analyze failure cases of our method, we visualize the confusion matrix of the result of misclassification. We find that the recognition accuracy of some classes in the cross-view benchmark of the NTU RGB+D dataset is relatively low. These action categories include: "reading & writing & playing phone & keyboarding & rubbing hands". From the confusion matrix in Fig.6, it can be seen that the action categories of misclassification have a great similarity to other categories. We can see that 23% of the samples



| | Clapping | Reading | Writing | Playing Phone | Keyboarding | Rubbing Hands |
|---|---|---|---|---|---|---|
| Reading | | 0.51 | 0.23 | 0.09 | 0.08 | |
| Writing | | 0.07 | 0.54 | 0.12 | 0.22 | |
| Playing Phone | | 0.02 | 0.10 | 0.78 | 0.09 | |
| Keyboarding | | 0.03 | 0.11 | 0.07 | 0.75 | |
| Rubbing Hands | 0.09 | | | | | 0.82 |

Fig.6. Comparison of the confusion matrix of failure cases on the NTU RGB+D dataset. The confusion matrix shows some action classes ("reading", "writing", "playing phone", "keyboarding" and "rubbing hands") whose accuracy is less than 82%. The row and the column of the confusion matrix denote the ground truth and the prediction, respectively.

in the "reading" class are mistakenly assigned to "writing" because of the high similarities between the two action classes in the skeleton sequence. Similarly, 22% of the samples in the "writing" category are mistakenly assigned to the "keyboarding" category. For the NTU RGB+D dataset, the recognition accuracy of these five categories is lower because they highly depend on the skeleton points of the hand and the dataset has only two points for this body part.

## 6 Conclusions

We presented a two-stream network based on TCN for human action recognition. Firstly, we designed the inter-frame vector feature representation between adjacent frames and the intra-frame vector feature representation within a single frame. These vector feature representations are extracted from the original human skeleton sequences and composed as the input of the network. We proposed a modified residual block for TCN, which significantly improves its performance.

In this work, we achieved a competitive recognition accuracy of 90.2% on the most widely used NTU RGB+D dataset. The experimental results proved the feasibility of our TS-TCN network. The integration of inter-frame and intra-frame vector feature representations is currently the most effective way to improve the accuracy of human action recognition; therefore the two-stream network has a great practical value. At the same time, some failure cases based on skeleton-based action recognition are also given in Subsection 5.4. This problem can be supplemented by RGB images in conjunction with skeleton data to make up for the lack of skeleton information, which can further improve action recognition accuracy and promote the application of this research in industry. In the future, we hope to combine the skeleton sequence with the RGB video sequence to improve the performance of action recognition.

## References

[1] Aggarwal J K, Xia L. Human activity recognition from 3D data: A review. *Pattern Recognition Letters*, 2014, 48: 70-80.

[2] Weinland D, Ronfard R, Boyer E. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, 2011, 115(2): 224-241.

[3] Han F, Reily B, Hoff W, Zhang H. Space-time representation of people based on 3D skeletal data: A review. *Computer Vision and Image Understanding*, 2017, 158: 85-105.

[4] Liu H, Liu B, Zhang H, Li L, Qin X, Zhang G. Crowd evacuation simulation approach based on navigation knowledge and two-layer control mechanism. *Information Sciences*, 2018, 436/437: 247-267.

[5] Turaga P, Chellappa R, Subrahmanian V S. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 2008, 18(11): 1473-1488.

[6] Herath S, Harandi M, Porikli F. Going deeper into action recognition: A survey. *Image and Vision Computing*, 2017, 60: 4-21.

[7] Hou J H, Chau L P, Thalmann N M, He Y. Compressing 3-D human motions via keyframe-based geometry videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 2014, 25(1): 51-62.

[8] Sermanet P, Lynch C, Hsu J, Levine S. Time-contrastive networks: Self-supervised learning from multi-view observation. In *Proc. the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, July 2017, pp.486-487.

[9] Shotton J, Sharp T, Kipman A, Fitzgibbon A, Finocchio M, Blake A, Cook M, Moore R. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 2011, 56(1): 116-124.

[10] Li S, Fang Z, Song W, Hao A, Qin H. Bidirectional optimization coupled lightweight networks for efficient and robust multi-person 2D pose estimation. *Journal of Computer Science and Technology*, 2019, 34(3): 522-536.

[11] Shahroudy A, Liu J, Ng T T, Gang W. NTU RGB+D: A large scale dataset for 3D human activity analysis. In *Proc. the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, June 2016, pp.1010-1019.

[12] Zhu F, Shao L, Xie J, Fang Y. From handcrafted to learned representations for human action recognition: A survey. *Image and Vision Computing*, 2016, 55: 42-52.

[13] Huang Z W, Wan C, Probst T, Van G L. Deep learning on lie groups for skeleton-based action recognition. In *Proc. the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, July 2017, pp.1243-1252.

[14] Ke Q, An S, Bennamoun M, Sohel F, Boussaid F. Skeleton-Net: Mining deep part features for 3-D action recognition. *IEEE Signal Processing Letters*, 2017, 24(6): 731-735.

[15] Weng J, Weng C, Yuan J, Liu Z. Discriminative spatio-temporal pattern discovery for 3D action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 2019, 29(4): 1077-1089.

[16] Liu J, Shahroudy A, Xu D, Kot A C, Wang G. Skeleton-based action recognition using spatiotemporal LSTM network with trust gates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018, 40(12): 3007-3021.

[17] Lee I, Kim D, Kang S, Lee S. Ensemble deep learning for skeleton-based action recognition using temporal sliding LSTM networks. In *Proc. the 2017 IEEE International Conference on Computer Vision*, October 2017, pp.1012-1020.

[18] Zhang P, Xue J, Lan C, Zeng W, Gao Z, Zheng N. Adding attentiveness to the neurons in recurrent neural networks. In *Proc. the 15th European Conference on Computer Vision*, September 2018, pp.136-152.

[19] Meng F, Liu H, Liang Y, Tu J, Liu M. Sample fusion network: An end-to-end data augmentation network for skeleton-based human action recognition. *IEEE Transactions on Image Processing*, 2019, 28(11): 5281-5295.

[20] Yan S, Xiong Y, Lin D. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proc. the 32nd AAAI Conference on Artificial Intelligence*, February 2018, pp.7444-7452.

[21] Shi L, Zhang Y, Cheng J, Lu H. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proc. the 2019 IEEE Conference on Computer Vision and Pattern Recognition*, June 2019, pp.12026-12035.

[22] Li M, Chen S, Chen X, Zhang Y, Wang Y, Tian Q. Actional-structural graph convolutional networks for skeleton-based action recognition. In *Proc. the 2019 IEEE Conference on Computer Vision and Pattern Recognition*, June 2019, pp.3595-3603.

[23] Si C, Chen W, Wang W, Wang L, Tan T. An attention enhanced graph convolutional LSTM network for skeleton-based action recognition. In *Proc. the 2019 IEEE Conference on Computer Vision and Pattern Recognition*, June 2019, pp.1227-1236.

[24] Lea C, Flynn M D, Vidal R, Reiter A, Hager G D. Temporal convolutional networks for action segmentation and detection. In *Proc. the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, July 2017, pp.1003-1012.

[25] Kim T S, Reiter A. Interpretable 3D human action analysis with temporal convolutional networks. In *Proc. the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, July 2017, pp.1623-1631.

[26] Liu J, Shahroudy A, Perez M, Wang G, Duan L Y, Kot A C. NTU RGB+D 120: A large-scale benchmark for 3D human activity understanding. arXiv:1905.04757, 2019. https://arxiv.org/pdf/1905.04757.pdf, Jan. 2020.

[27] Jiang W, Nie X, Xia Y, Wu Y, Zhu S C. Cross-view action modeling, learning and recognition. In *Proc. the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp.2649-2656.

[28] Xia L, Chen C C, Aggarwal J K. View invariant human action recognition using histograms of 3D joints. In *Proc. the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2012, pp.20-27.

[29] Liu Z, Zhang C, Tian Y. 3D-based deep convolutional neural network for action recognition with depth sequences. *Image and Vision Computing*, 2016, 55: 93-100.

[30] Wang P, Li W, Wan J, Ogunbona P, Liu X. Cooperative training of deep aggregation networks for RGB-D action recognition. In *Proc. the 32nd AAAI Conference on Artificial Intelligence*, February 2018, pp.7404-7411.

[31] Jiang W, Liu Z, Wu Y, Yuan J. Learning actionlet ensemble for 3D human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014, 36(5): 914-927.

[32] Zhang S, Liu X, Xiao J. On geometric features for skeleton-based action recognition using multilayer LSTM networks. In *Proc. the 2017 IEEE Winter Conference on Applications of Computer Vision*, March 2017, pp.148-157.

[33] Zhang P, Lan C, Xing J, Zeng W, Xue J, Zheng N. View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In *Proc. the 2017 IEEE International Conference on Computer Vision*, October 2017, pp.2136-2145.

[34] Ke Q, Bennamoun M, An S, Sohel F, Boussaïd F. A new representation of skeleton sequences for 3D action recognition. In *Proc. the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, July 2017, pp.4570-4579.

[35] Ghorbel E, Boonaert J, Boutteau R, Lecoeuche S, Savatier X. An extension of kernel learning methods using a modified Log-Euclidean distance for fast and accurate skeleton-based human action recognition. *Computer Vision and Image Understanding*, 2018, 175: 32-43.

[36] Yuan J, Liu Z, Wu Y. Discriminative subvolume search for efficient action detection. In *Proc. the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2009, pp.2442-2449.

[37] Liu M, Shi Y, Zheng L, Xu K, Huang H, Manocha D. Recurrent 3D attentional networks for end-to-end active object recognition. *Computational Visual Media*, 2019, 5(1): 91-104.

[38] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. the 32nd International Conference on Machine Learning*, July 2015, pp.448-456.

[39] He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: Surpassing human-level performance on imageNet classification. In *Proc. the 2015 IEEE International Conference on Computer Vision*, December 2015, pp.1026-1034.

[40] Girija S S. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. arXiv:1603.04467, 2016. https://arxiv.org/abs/1603.04467, Jan. 2020.

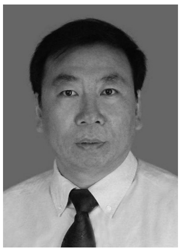
[41] Ofli F, Chaudhry R, Kurillo G, Vidal R, Bajcsy R. Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition. In *Proc. the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2012, pp.8-13.

[42] Zhao R, Wang K, Su H, Ji Q. Bayesian graph convolution LSTM for skeleton based action recognition. In *Proc. the 2019 IEEE Conference on International Conference on Computer Vision*, October 2019, pp.6881-6891.

[43] Yu Z, Chen W, Guo G. Fusing spatiotemporal features and joints for 3D action recognition. In *Proc. the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, June 2013, pp.486-491.

**Jin-Gong Jia** received his B.S. degree in software engineering at School of Information and Electrical Engineering, Ludong University, Yantai, in 2018. Currently, he is currently pursuing his Master's degree in software engineering at the School of Software, Shandong University, Jinan. His research interests include computer vision, human action recognition, and human pose estimation.

**Yuan-Feng Zhou** received his Master's and Ph.D. degrees in computer science and technology at the School of Computer Science and Technology, Shandong University, Jinan, in 2005 and 2009, respectively. He held a post-doctoral position with the Graphics Group, Department of Computer Science, The University of Hong Kong, Hong Kong, from 2009 to 2011. He is currently a professor with the School of Software, Shandong University, Jinan, where he is also a member of the Geometric Design and Information Visualization (GDIV) Laboratory. His current research interests include geometric modeling, information visualization, and image processing.



**Xing-Wei Hao** is a professor of Shandong University, Jinan. He received his Master's and Ph.D. degrees in computer science and technology at the School of Computer Science and Technology, Shandong University, Jinan, in 1990 and 2007, respectively. His current research interests include data mining, knowledge graph, and machine learning.

**Feng Li** is currently an associate professor in School of Software at Shandong University, Jinan. He received his Master's degree in computer science and technology at the School of Computer Science, Shandong University of Technology, Jinan, in 1998, and received his Ph.D. degree in computer science and technology at the School of Computer Science and Technology, Shandong University, Jinan, in 2019. His research interests cover wireless sensor networks, embedded systems and information management systems. He has authored a number of refereed papers in the related conferences and journals.

**Christian Desrosiers** received his Ph.D. degree in computer engineering from Polytechnique Montreal, Montreal, in 2008. He was a post-doctoral researcher at the University of Minnesota on the topic of machine learning. In 2009, he joined the Department of Software and IT Engineering, University of Quebec, Montreal, as a professor. He is also the co-director of the Laboratoire d'imagerie, de vision et d'intelligence artificielle (Laboratories LIVIA, Imaging Vision and Artificial Intelligence Laboratory). His main research interests focus on machine learning, image processing, computer vision, and medical imaging. He is a member of the REPARTI research network.

**Cai-Ming Zhang** is a professor and doctoral supervisor of the School of the Software at Shandong University, Jinan. He is now also the dean of the Digital Media Research Institute at Shandong University of Finance and Economics, Jinan. He received his B.S. and M.E. degrees in computer science at the School of Computer Science, Shandong University, Jinan, in 1982 and 1984, respectively, and his Dr. Eng. degree in computer science from the Tokyo Institute of Technology, Tokyo, in 1994. From 1997 to 2000, Dr. Zhang had held visiting position at the University of Kentucky, USA. His research interests include CAGD, CG, information visualization and medical image processing.